# Natural for z/OS

## Operations (Verwaltung/Betrieb)

Version 9.2.5

Oktober 2026

ADABAS & NATURAL

This document applies to Natural for z/OS Version 9.2.5 and all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 1979-2026 Software GmbH, Darmstadt, Germany and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software GmbH product names are either trademarks or registered trademarks of Software GmbH and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Detailed information on trademarks and patents owned by Software GmbH and/or its subsidiaries is located at https://softwareag.com/licenses.

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third-Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software GmbH Products / Copyright and Trademark Notices of Software GmbH Products". These documents are part of the product documentation, located at https://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

Use of this software is subject to adherence to Software GmbH's licensing conditions and terms. These terms are part of the product documentation, located at https://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

**Document ID: ATMF-OPERATIONS-925-20261013DE**

# Inhaltsverzeichnis


Vorwort ... ix
1 Über diese Dokumentation ... 1
Dokumentationskonventionen ... 2
Online-Informationen und Support ... 2
Datenschutz ... 3
I Natural konfigurieren ... 5
2 Natural-Objekte mit dem Natural-Nukleus verlinken ... 7
Vorteile der Verlinkung von Natural-Objekten mit dem Natural-Nukleus ... 8
Dienstprogramm ULDOBJ ... 9
ULDOBJ zur Generierung eines Objektmoduls benutzen ... 9
Zusätzliche Überlegungen zur Verlinkung von Subroutinen ... 11
Betriebssystemabhängigkeit der Objektmodulgenerierung ... 11
Beispiel für die Verlinkung eines Natural-Objekts mit dem Natural-Nukleus ... 11
3 Natural User Exits ... 15
NATUEX1 - User Exit für die Berechtigungskontrolle ... 16
NATSREX2 und NATSREX3 - User Exits für die Sortierverarbeitung ... 17
NATUSKnn - User Exit zur Berechnung von Sortierschlüsseln ... 18
NATPM - User Exit für invertierte Ausgabe ... 19
NREXPG - User Exit für NATRJE ... 20
USR0070P - User Exit für Editor-Profile ... 21
USR2002P - User Exit für Textstrings im Hilfefenster ... 21
USR2003P - User Exit für Hauptmenü ... 22
4 Natural User Access Method für Druck- und Arbeitsdateien ... 23
Beschreibung des Moduls NATAMUSR ... 24
Installation des Moduls NATAMUSR ... 24
Aufrufen des Drittanbieterprodukts ... 25
5 Natural-Systemdateien ... 27
Natural-Scratch-Pad-Datei ... 28
6 Natural-Text-Module und Makros ... 31
Funktion und Verwendung von Textmodulen ... 32
NATTEXT - Natural-Schlüsselwort-Definitionen ... 32
NATTXT2 - Ausgabetext, Schlüsselwörter und Benutzerabbruchmeldungen (in Groß- un Kleinschreibung) ... 33
NATTXT2U - Ausgabetext, Schlüsselwörter und Benutzerabbruchmeldungen (in Großbuchstaben) ... 36
NATTXT3 - Textfragmente für Platzhalter in Natural-Fehlermeldungen ... 36
NTERMSG - Natural-Beendigungsmeldungen und Rückgabecodes ... 37
7 Natural-Konfigurationstabellen ... 39
NATCONFG - Natural-Konfigurationstabellen ... 40
Allgemeine Übersicht über die in NATCONFG verwendeten Makros ... 40
NTDVCE - Terminal-Gerätespezifikationstabelle ... 41

NTMSG - Definitionen der Meldungsprotokolltabelle ... 42
NTSTAT - Definition der mit dem Natural-Nukleus verlinkten Objekte ... 43
NTCPAGE - Definitionen der Codepages ... 43
Unterstützung von Codepages ... 45
Unterstützte Ausgabegeräte ... 45
Umsetzungstabellen ... 46
Umsetzung von Klein- in Großbuchstaben ... 50
CMULT-Eintrag ... 51
Umwandlung der Ausgabe ... 51
Umwandlung von Eingaben ... 52
Code-Umsetzung von DBCS-Daten ... 52
NTTZ - Zeitzonendefinitionen ... 53
8 Natural-Speicherverwaltung ... 59
Thread- und Nicht-Thread-Umgebungen ... 60
Puffertypen ... 60
Feste Puffer ... 61
Variable Puffer ... 61
Anpassung der Puffereigenschaften ... 62
II Profilparameter anwenden ... 65
9 Natural-Parameter-Hierarchie ... 67
Übersicht über die Natural-Parameterhierarchie ... 68
Allgemeine Regeln für die Verwendung von Parametern ... 68
Natural-Parametermodul ... 69
Vordefinierte dynamische Parameter-Sets ... 70
Vordefinierte Benutzer-Parameter-Profile ... 70
Dynamische Parametereingabe ... 70
Natural-Security-Definitionen ... 71
Profilparametereinstellungen während der Sitzung zeigen/ändern ... 71
Einstellungen auf Programm-/Statement-Ebene ... 72
Einstellungen der Entwicklungsumgebung ... 72
Beispiele für die Parameterauswertung ... 72
10 Zuweisung von Parameterwerten ... 75
Quellen für die Parameterwertzuweisung ... 76
Statische Zuweisung von Parameterwerten ... 77
Dynamische Zuweisung von Parameterwerten ... 78
Session-Parameter zur Zuweisung von Parameterwerten zur Laufzeit ... 79
11 Generierung eines Natural-Parametermoduls ... 81
Parameter-Makro NTPRM ... 82
Zusätzliche Makros im Natural-Parametermodul ... 83
Beispiel für Makros im Natural-Parametermodul ... 85
III z/OS-Umgebung ... 89
12 Natural unter z/OS ... 91
Natural-Subsystem ... 92
TP-Monitor-Schnittstellen ... 92
Datenbankmanagementsystem-Schnittstellen ... 92

Natural im Batch-Modus unter z/OS ... 93
Natural als Server unter z/OS ... 93
13 Authorized Services Manager unter z/OS ... 95
ASM-Übersicht ... 96
ASM-Systemanforderungen ... 97
ASM starten ... 99
ASM-Operator-Kommandos ... 104
Struktur der Coupling Facility für ASM zurücksetzen ... 105
ASM-Meldungen, Condition Codes und Abend Codes ... 105
14 Natural Roll Server-Funktionen ... 107
Natural Roll Server - Überblick ... 108
Roll-Server in einem einzelnen z/OS-System ... 109
Roll Server in einer OS-Parallel-Sysplex-Umgebung ... 110
Roll File und LRB ... 112
15 Betrieb des Natural Roll Server ... 115
Roll Server-Systemanforderungen ... 116
Roll File formatieren ... 118
Natural Roll Server starten ... 122
Roll Server-Meldungen, Condition Codes und Abend Codes ... 130
Return Codes und Reason Codes der Roll Server-Anforderung ... 131
Roll Server-Betriebsfunktionen ... 131
Struktur der Coupling Facility zurücksetzen ... 133
Roll Server-Leistung optimieren ... 133
Roll Server User Exits ... 135
IV Natural im Batch-Modus ... 137
16 Natural im Batch-Modus - Allgemeines ... 139
Adabas-Datasets ... 140
Sort-Datasets ... 140
Subtasking-Session-Unterstützung bei Batch-Modus-Umgebungen ... 140
17 Natural im Batch-Modus unter z/OS ... 145
Natural z/OS-Batch-Schnittstelle ... 146
Treiberparameter für z/OS Batch ... 146
Von Natural im z/OS-Batch-Modus verwendete Datasets ... 146
V Natural Buffer Pools ... 153
18 Natural Buffer Pool - Allgemeines ... 155
Natural Buffer Pool - Funktionsprinzip ... 156
Buffer Pool-Überwachung und -Verwaltung ... 162
Globaler Natural Buffer Pool ... 165
19 Natural Global Buffer Pool under z/OS ... 167
Verwendung eines Natural Global Buffer Pool ... 168
Voraussetzungen ... 168
Betrieb eines Natural Global Buffer Pool (GBP) ... 168
GBP-Manager-Parametermodul ... 170
GBP-Betriebsfunktionen ... 171
Funktionsparameter des globalen Buffer Pools ... 173

Beispiele für NATBUFFER-Angaben ... 181
Beispiele für NATGBPvr-Ausführungsjobs ... 181
Lokalisierung ... 183
Meldungen ... 184
VI Message Buffer Pool verwenden ... 185
20 Message Buffer Pool verwenden ... 187
Zweck ... 188
Voraussetzungen ... 188
Betrieb des Message Buffer Pool ... 189
Beispiele für NATMBPvr-Ausführungsjobs ... 190
Message Buffer Pool-Betriebsfunktionen ... 191
Funktionsparameter ... 192
Meldungen ... 194
VII System-Spool-Zugang verwalten ... 195
21 System-Spool-Zugang verwalten ... 197
Zweck ... 198
Voraussetzung ... 198
Verwendung der Write-to-Spool-Funktion ... 198
VIII Natural 3GL CALLNAT-Schnittstelle verwenden ... 203
22 Natural 3GL CALLNAT-Schnittstelle - Zweck, Voraussetzungen, Einschränkungen ... 205
Zweck der 3GL CALLNAT-Schnittstelle ... 206
Voraussetzungen ... 206
Einschränkungen ... 208
23 Natural 3GL CALLNAT-Schnittstelle - Verwendung, Beispiele ... 211
Verwendung ... 212
Beispielumgebungen ... 216
IX Betrieb des Software AG Editor ... 219
24 Editor-Arbeitsdatei ... 221
Struktur der Editor-Arbeitsdatei ... 222
Editor-Arbeitsdatei unter z/OS ... 223
Verwendung der Software AG Editor Work File Formatting Utility ... 224
Formatierung während der Initialisierung ... 224
Verwaltung der Editor-Arbeitsdatei ... 225
Editor-Arbeitsdatei unter Com-plete/SMARTS ... 226
25 Editor Buffer Pool ... 227
Zweck des Editor Buffer Pool ... 228
Freie Blöcke erhalten ... 229
Initialisierung des Editor Puffer Pool ... 229
Neustart des Editor Buffer Pool ... 230
Editor Buffer Pool-Parameter ... 230
Buffer Pool-Initialisierung für Mehrbenutzerumgebungen ... 230
X Natural als Server ... 233
26 Natural als Server unter z/OS ... 235
Funktionalität unter z/OS ... 236

Natural-Nukleus-Installation in einer Server-Umgebung ..... 237
Behandlung von Druck- und Arbeitsdateien mit externen Datasets in einer Serverumgebung ..... 237
27 Natural als Server unter CICS ..... 239
Funktionalität unter CICS ..... 240
Installation der Natural CICS-Schnittstelle in einer Server-Umgebung ..... 240
Einschränkungen ..... 241
XI Natural-Ausführung - Verschiedene Themen ..... 243
28 Natural 31-Bit Mode-Unterstützung ..... 245
29 Unterstützung und Nutzung von Natural- und Nicht-Natural-Objekten ..... 247
Unterstützung für Natural-Objekte aus früheren Natural-Versionen ..... 248
Konventionen für den Aufruf von Backend-Programmen ..... 248
LE-Subprogramme ..... 250
Externe Sortierprogramme ..... 253
30 Eingabe-/Ausgabegeräte ..... 255
Terminal-Unterstützung ..... 256
Unterstützung von Lichtstiften (Light Pens) ..... 256
Drucker-Unterstützung ..... 258
31 Doppel-Byte-Zeichensätze ..... 261
Natural-Profilparameter SOSI ..... 262
Angabe des Ausgabeformats ..... 262
Parameterdefinitionen für DBCS-Unterstützung ..... 262
Editor-Profil-Optionen ..... 263
Prüfung der Eingabedaten ..... 264
Anpassung der Ausgabedaten ..... 264
Natural Stack-Daten ..... 265
Anwendungsprogrammierschnittstellen für die DBCS-Behandlung ..... 265
Alternatives Textmodul NATTXT2U ..... 266
32 Asynchrone Verarbeitung ..... 267
Kennzeichnung asynchroner Natural-Sitzungen ..... 268
Behandlung der Ausgabe einer asynchronen Natural-Sitzung ..... 268
Behandlung unerwarteter oder unerwünschter Eingaben ..... 269
Weitere Überlegungen zu Profilparametern ..... 269

# Vorwort

Diese Dokumentation enthält Informationen zur Verwaltung und zum Betrieb von Natural in einer Großrechner-Umgebung unter dem Betriebssystem z/OS.

Sie ist in die folgenden Hauptteile gegliedert:

| | |
|---|---|
| **Natural konfigurieren** | Beschreibt, wie Sie Natural-Objekte mit dem Natural-Nukleus verlinken. Enthält Informationen über Natural-User-Exits, Natural-Benutzer-Zugriffsmethoden auf Druck- und Arbeitsdateien, Natural-Systemdateien, Natural-Textmodule, Natural-Konfigurationstabellen und die Natural-Speicherverwaltung. |
| **Profilparameter anwenden** | Gibt einen Überblick über die hierarchische Struktur der verschiedenen Ebenen, auf denen Natural-Parameter gesetzt werden können. Erläutert, wie Profilparametern statisch, dynamisch und zur Laufzeit Werte zugewiesen werden können. Beschreibt, wie Sie ein Natural-Parametermodul erstellen. |
| **z/OS-Umgebung verwalten** | Enthält eine Zusammenfassung spezieller Aspekte, die gelten, wenn Sie Natural unter z/OS online oder im Batch-Modus ausführen. Beschreibt die Funktionen und den Betrieb des Authorized Services Manager (ASM). Erläutert die Funktionen des Natural Roll Servers. Enthält Informationen zum Roll Server: Systemanforderungen, Betrieb, Leistungsoptimierung und Neustartfähigkeit. |
| **Natural im Batch-Modus betreiben** | Enthält allgemeine Überlegungen zum Betrieb von Natural im Batch-Modus (Adabas-Datasets, Sort-Datasets, Unterstützung von Subtasking-Sitzungen für Batch-Umgebungen) und speziell zum Betrieb von Natural im Batch-Modus unter z/OS. |
| **Natural Buffer Pools verwalten** | Enthält Informationen zu den verschiedenen Speicherverwaltungsfunktionen, die Ihnen als Natural-Systemverwalter unter z/OS zur Verfügung stehen. |
| **Message Buffer Pool verwenden** | Enthält Informationen zum Starten und Betreiben des Message Text Buffer Pool. |
| **System-Spool-Zugang verwalten** | Mit der Write-to-Spool-Funktionalität können Natural-Benutzer Reports direkt in den System-Spool schreiben. Sie kann in jeder Natural-Umgebung (Com-plete, TSO, CICS, IMS TM, Batch usw.) eingesetzt werden und verwendet die View WRITE-SPOOL des Entire System Server. Diese Ausgabe kann dann von jeder Software verarbeitet werden, die eine Ausgabe in JES oder POWER-Spool erwartet (z. B. Entire Output Management). |
| **Natural 3GL CALLNAT-Schnittstelle verwenden** | Enthält Informationen über die Natural 3GL `CALLNAT`-Schnittstelle, über die Natural 3GL-Programme Natural-Subprogramme aufrufen und ausführen kann. |
| **Betrieb des Software AG-Editors** | Enthält Informationen zum Betrieb des Software AG-Editors. |
| **Natural als Server verwenden** | Beschreibt die Verwendung von Natural als Server und den Natural Server Monitor. |

| | |
|---|---|
| **Natural-Ausführung - Verschiedene Themen** | Enthält Informationen zur Unterstützung des Natural-31-Bit-Modus, zur Unterstützung und Verwendung von Natural- und Nicht-Natural-Objekten, zu Ein- und Ausgabegeräten, Doppelbyte-Zeichensätzen und asynchroner Verarbeitung. |

### Notation *vrs* oder *vr*

In dieser Dokumentation steht die Schreibweise *vrs* oder *vr* für die jeweilige Produktversion (siehe auch *Version* im *Glossar*).

# 1 Über diese Dokumentation


- Dokumentationskonventionen ..... 2
- Online-Informationen und Support ..... 2
- Datenschutz ..... 3

## Dokumentationskonventionen

| Konvention | Beschreibung |
|---|---|
| **Fettschrift** | >Kennzeichnet Elemente auf einem Bildschirm. |
| `Nichtproportionale Schrift` | Kennzeichnet Namen und Orte von Diensten im Format *`Ordner.Unterordner.Dienst`*, Programmierschnittstellen (APIs), Namen von Klassen, Methoden und Properties in Java. |
| *`Kursivschrift`* | Kennzeichnet:<br><br>Variablen, für die Sie situations- oder umgebungsspezifische Werte angeben müssen.<br>Neue Begriffe, wenn sie erstmals im Text auftreten.<br>Verweise auf andere Dokumentationsquellen. |
| `Nichtproportionale Schrift` | Kennzeichnet:<br><br>Text, den Sie eingeben müssen.<br>Meldungen, die vom System angezeigt werden.<br>Programmcode. |
| { } | Zeigt eine Reihe von Auswahlmöglichkeiten an, von denen Sie eine auswählen müssen. Geben Sie nur die innerhalb der geschweiften Klammern vorhandenen Informationen ein. Geben Sie nicht die Klammersymbole { } ein. |
| `\|` | Trennt zwei sich gegenseitig ausschließende Auswahlmöglichkeiten in einer Syntaxzeile voneinander ab. Geben Sie eine der Auswahlmöglichkeiten ein. Geben Sie nicht das Symbol `\|` ein. |
| [ ] | Zeigt eine oder mehrere Optionen an. Geben Sie nur die innerhalb der eckigen Klammern vorhandenen Informationen ein. Geben Sie nicht die Klammersymbole [ ] ein. |
| ... | Zeigt an, dass Sie mehrere Auswahlmöglichkeiten desselben Typs eingeben können. Geben Sie nur die Informationen ein. Geben Sie nicht die drei Auslassungspunkte (...) ein. |

## Online-Informationen und Support

### Produktdokumentation

Sie finden die Produktdokumentation auf unserer Dokumentationswebsite unter **https://documentation.softwareag.com**.

Zusätzlich können Sie auch über **https://www.softwareag.cloud** auf die Dokumentation für die Cloud-Produkte zugreifen. Navigieren Sie zum gewünschten Produkt und gehen Sie dann, je nach Produkt, zu „Developer Center“, „User Center“ oder „Documentation“.

### Produktschulungen

Sie finden hilfreiches Produktschulungsmaterial auf unserem Lernportal unter **https://knowledge.softwareag.com**.

### Tech Community

Auf der Website unserer Tech Community unter **https://techcommunity.softwareag.com** können Sie mit Experten der Software AG zusammenarbeiten. Von hier aus können Sie zum Beispiel:

- Unsere umfangreiche Wissensdatenbank durchsuchen.
- In unseren Diskussionsforen Fragen stellen und Antworten finden.
- Die neuesten Nachrichten und Ankündigungen der Software AG lesen.
- Unsere Communities erkunden.
- Unsere öffentlichen Repositories auf GitHub and Docker unter **https://github.com/softwareag** und **https://hub.docker.com/publishers/softwareag** besuchen und weitere Ressourcen der Software AG entdecken.

### Produktsupport

Support für die Produkte der Software AG steht lizenzierten Kunden über unser Empower-Portal unter **https://empower.softwareag.com** zur Verfügung. Für viele Dienstleistungen auf diesem Portal benötigen Sie ein Konto. Wenn Sie noch keines haben, dann können Sie es unter **https://empower.softwareag.com/register** beantragen. Sobald Sie ein Konto haben, können Sie zum Beispiel:

- Produkte, Aktualisierungen und Programmkorrekturen herunterladen.
- Das Knowledge Center nach technischen Informationen und Tipps durchsuchen.
- Frühwarnungen und kritische Alarme abonnieren.
- Supportfälle öffnen und aktualisieren.
- Anfragen für neue Produktmerkmale einreichen.

## Datenschutz

Die Produkte der Software AG stellen Funktionen zur Verarbeitung von personenbezogenen Daten gemäß der Datenschutz-Grundverordnung (DSGVO) der Europäischen Union zur Verfügung. Gegebenenfalls sind in der betreffenden Systemverwaltungsdokumentation entsprechende Schritte dokumentiert.

# I Natural konfigurieren

Dieser Teil der Dokumentation beschreibt, wie Sie Natural anpassen können.

**Natural-Objekte mit dem Natural-Nukleus verlinken**
**Natural User Exits**
**Natural User Access Method für Druck- und Arbeitsdateien**
**Natural-Systemdateien**
**Natural-Text-Module und Makros**
**Natural-Konfigurationstabellen**
**Natural-Speicherverwaltung**

Siehe auch folgende Dokumente in der *Debugger und Dienstprogramme (Utilities)*-Dokumentation:

- *SYSCP Utility - Codepage-Verwaltung*
- *SYSEXT Utility - Natural-Anwendungsprogrammierschnittstellen*
- *SYSAPI Utility - APIs von Natural Add-on-Produkten*

# 2 Natural-Objekte mit dem Natural-Nukleus verlinken


- Vorteile der Verlinkung von Natural-Objekten mit dem Natural-Nukleus ..... 8
- Dienstprogramm ULDOBJ ..... 9
- ULDOBJ zur Generierung eines Objektmoduls benutzen ..... 9
- Zusätzliche Überlegungen zur Verlinkung von Subroutinen ..... 11
- Betriebssystemabhängigkeit der Objektmodulgenerierung ..... 11
- Beispiel für die Verlinkung eines Natural-Objekts mit dem Natural-Nukleus ..... 11

Der Natural-Nukleus ist eine Sammlung von Dienstprogrammen wie Speicherverwaltung, String-Handling, Betriebssystemschnittstellen, Compiler und Laufzeitumgebung, die den Kern von Natural bilden. Er ist unabhängig von Betriebssystem und TP-System.

Der Natural-Nukleus besteht aus dem umgebungsunabhängigen und dem umgebungsabhängigen Nukleus. Der umgebungsunabhängige Nukleus kann von mehreren Großrechner-Betriebs- und TP-Systemen verwendet werden. Der umgebungsabhängige Nukleus enthält Komponenten, die von den Betriebs- und TP-Systemen abhängen. Weitere Informationen finden Sie unter *Komponenten des Natural-Nukleus* in der *Installation*-Dokumentation.

Dieses Kapitel beschreibt die Vorteile der Verlinkung von Natural-Objekten mit dem Natural-Nukleus und gibt Informationen über die Vorgehensweise. Folgenden Themen werden behandelt:

## Vorteile der Verlinkung von Natural-Objekten mit dem Natural-Nukleus

Die Verlinkung von Natural-Objekten mit dem Natural-Nukleus bietet folgende Vorteile:

- **Bessere Performance**
  Die Objekte werden aus dem Nukleus und nicht aus dem **Natural Buffer Pool** ausgeführt. Das spart Platz im Buffer Pool und führt auch zu weniger Datenbankaufrufen. (Wenn katalogisierte Natural-Objekte nicht mit dem Natural-Nukleus verlinkt sind, werden sie in einer Datenbankdatei gespeichert, z.B. in Adabas, und der eigentliche Code muss aus dieser Datei in den Buffer Pool geladen werden, bevor er ausgeführt werden kann).
- **Konsistenz**
  Da ein Objekt, das mit dem Natural-Nukleus verlinkt ist, immer vom Nukleus ausgeführt wird, hat es keine Auswirkungen, wenn das katalogisierte Objekt, von dem es abgeleitet wurde, in der Natural-Systemdatei gelöscht oder geändert wird. Somit bleibt der Status des Objekts während jeder TP-Monitor-Sitzung unverändert. Eine neue Version eines Objekts, das mit dem Nukleus verlinkt ist, erhalten Sie, indem Sie es mit dem Dienstprogramm **ULDOBJ** entladen, die neue Version erneut mit dem Natural-Nukleus verlinken und das Natural-Modul aktualisieren. („aktualisieren“ bedeutet, dass eine neue Kopie eines Moduls in die TP-Monitor-Region geladen wird).
- **Globale Fehlerbehandlung**
  Wenn ein katalogisiertes Objekt ein anderes Programm zur Fehlerbehandlung aufruft (z. B. über die Natural-Systemvariable `*ERROR-TA`) und das Fehlerbehandlungsprogramm nicht in den Buffer Pool geladen werden kann, kann der ursprüngliche Fehler möglicherweise nicht erkannt werden und jeder nachfolgende Fehler kann den ersten Fehler maskieren und zu Verwirrung führen. Um diese Situation zu vermeiden, können Sie ein vom Benutzer geschriebenes globales Fehlerbehandlungsprogramm mit dem Nukleus verlinken.

## Dienstprogramm ULDOBJ

Mit dem Dienstprogramm ULDOBJ können Sie katalogisierte Natural-Objekte mit dem Natural-Nukleus verlinken. Mit dem Dienstprogramm ULDOBJ erzeugen Sie ein Objektmodul aus einem katalogisierten Natural-Objekt und schreiben es in eine Natural-Arbeitsdatei. Der generierte Objektbaustein wird dann vom Linkage Editor verarbeitet und mit dem Natural-Nukleus verlinkt.

Wenn ein umgebungsunabhängiger Natural-Nukleus verwendet wird, muss das generierte Objektmodul mit dem umgebungs*un*abhängigen Teil des Nukleus verlinkt werden.

## ULDOBJ zur Generierung eines Objektmoduls benutzen

**Um das ULDOBJ-Dienstprogramm aufzurufen:**

1 Melden Sie sich in der Library `SYSMISC` an und geben Sie das Kommando `ULDOBJ` ein.

```
10:08:14            ***** NATURAL OBJECT MAINTENANCE *****          2023-05-31
User: XYZ                 - NATURAL ULDOBJ UTILITY -          Library: SYSMISC
                                                              Opsys .. z/OS


                    Specify parameters below ....

                              Object ...... ________   (Enter '.' to exit)
                              Library ..... SYSMISC_
                              OP System ... z/OS____

                                                                            ↩
 ↩
```

2 Geben Sie die folgenden Parameter an und bestätigen Sie sie:

| | |
|---|---|
| `Object` | Der Name des katalogisierten Objekts, das verarbeitet werden soll. Das Objekt kann ein Programm, ein Subprogramm, eine Subroutine, eine Helproutine oder eine Map (Maske) sein.<br><br>Sie können einen Punkt (.) eingeben, um die Utility-Ausführung zu beenden (siehe **unten**). |
| `Library` | Der Name der Library, die das katalogisierte Objekt enthält. |
| `OP System` | Der Name des Betriebssystems, für das der Objektbaustein generiert werden soll. Der Name des Betriebssystems muss z/OS sein. |

Für jedes verarbeitete Objekt zeigt das Dienstprogramm ULDOBJ einen Report an, der die folgenden Informationen enthält:

- den Objekttyp (Programm, Subprogramm, Subroutine, Helproutine, Map, Adapter); siehe *Objekte zum Erstellen und Pflegen von Natural-Anwendungen* im *Leitfaden zur Programmierung*,
- den Namen des bearbeiteten katalogisierten Objekts,
- den Programmiermodus (S = Structured Mode, R = Reporting Mode),
- den Namen der Library, die das katalogisierte Objekt enthält,
- den Namen des Betriebssystems, für das das Objekt erzeugt wurde,
- die Größe des katalogisierten Objekts und des optimierten Codes (falls zutreffend),
- die Natural-Version des katalogisierten Objekts (siehe *Version* im *Glossar*),
- Statistiken über die letzte Katalogisierung des Objekts, einschließlich Benutzer- und Terminalkennungen (IDs).

ULDOBJ fordert die Eingabe eines weiteren Objekts und einer weiteren Library an, nachdem die Daten der ersten Eingabe verarbeitet wurden. Das Betriebssystem wird nicht abgefragt, da es nicht sinnvoll ist, für dieselbe Natural-Arbeitsdatei Objektmodule für mehr als ein Betriebssystem zu generieren.

**Um das Dienstprogramm ULDOBJ zu beenden:**

- Nachdem das letzte katalogisierte Objekt bearbeitet wurde, geben Sie im ersten Eingabefeld (`Object`) einen Punkt (.) ein und drücken Sie ENTER.

  Das generierte Objektmodul entspricht dem Format des angegebenen Betriebssystems. Er hat ein verschiebbares Format (Relocation) mit nicht ausführbarem Code und besteht aus:

  - einem externen Symbolverzeichnis (ESD),
  - einem Relocation Dictionary (RLD),
  - Text mit den Anweisungen und Daten, die dem Programm entsprechen,
  - einem `END`-Statement (End-of-Module-Indikator für das Lademodul).

  Das generierte Objektmodul wird in eine Natural-Arbeitsdatei geschrieben, die als Eingabe für einen Linkage Editor verwendet wird.

  Das generierte Objektmodul muss vom Linkage Editor des Betriebssystems verarbeitet werden, bevor der Code als Load-Modul ausführbar ist (siehe **Beispiel** weiter unten). Jedes Load-Modul ist gültig, sobald es mit dem Natural-Nukleus verlinkt und durch eine `NTSTAT`-Eintragsdefinition im Natural-Konfigurationsmodul `NATCONFG` definiert ist (siehe ***Natural-Konfigurationstabellen***).

## Zusätzliche Überlegungen zur Verlinkung von Subroutinen

Nachdem ein katalogisiertes Objekt vom Dienstprogramm ULDOBJ entladen und mit dem Natural-Nukleus verlinkt wurde, kann das katalogisierte Objekt aus der Natural-Systemdatei gelöscht werden.

Dies gilt jedoch nicht für ein Objekt des Typs Subroutine. Eine Subroutine hat zwei Namen:

- den in den Statements `PERFORM` und `DEFINE SUBROUTINE` angegebenen Namen und
- den Namen des Objekts, das das Statement `DEFINE SUBROUTINE` enthält.

Natural verlinkt intern diese beiden Namen, was jedoch nur möglich ist, wenn das katalogisierte Objekt noch in der Natural-Systemdatei vorhanden ist. Würde das katalogisierte Objekt gelöscht, ginge diese Zuordnung verloren und die mit dem Nukleus verlinkte Subroutine wäre nicht ausführbar.

## Betriebssystemabhängigkeit der Objektmodulgenerierung

Eine `NAME`-Steueranweisung wird als letzte Karte des Objektmoduls generiert. Sie spezifiziert die Ersetzungsfunktion. Zum Beispiel:

```
NAME TEST (R)
```

`TEST` ist der Name des katalogisierten Objekts.

## Beispiel für die Verlinkung eines Natural-Objekts mit dem Natural-Nukleus

Wenn zum Beispiel die Objekte `LOGPROG` und `EDITPROG` in der Library `SYSLIB` mit dem Natural-Nukleus verlinkt werden sollen, könnten folgende Schritte durchgeführt werden:

1. Identifizieren Sie die katalogisierten Objekte, die verlinkt werden sollen.

```
Object         Library
-------------- --------------
LOGPROG        SYSLIB
EDITPROG       SYSLIB
```

2. Erstellen Sie den Batch Natural Job Stream, indem Sie die folgenden Karten verwenden:

```
//CMWKF01  DD DSN=ULD.NAT.PGMS,UNIT=SYSDA,DISP=(,KEEP),
//            SPACE=(CYL,(3,1),,RLSE),VOL=SER=VVVVVV,
//            DCB=(RECFM=FB,BLKSIZE=800,LRECL=80)
//CMSYNIN DD *
LOGON SYSMISC
ULDOBJ LOGPROG,SYSLIB,OS
EDITPROG,SYSLIB
.
FIN
/*
```

3. Erstellen Sie den Jobfluss für den Linkage Editor.

```
//JOBCARD JOB (ACCTING),CLASS=A,MSGCLASS=X
//*
//* GENERATE OS LOAD MODULE FROM ULDOBJ UTILITY
//*
//LINK1    EXEC PGM=IEWL,PARM='LIST,LET,XREF,NCAL,RENT,REUS'
//SYSLMOD  DD DSN=NATURAL.USER.LOAD,DISP=SHR
//SYSUT1   DD UNIT=SYSDA,SPACE=(1024,(200,20))
//SYSPRINT DD SYSOUT=X
//SYSLIN   DD DSN=NAT.ULD.PGMS,DISP=OLD,UNIT=SYSDA,VOL=SER=VVVVVV
/*
```

In diesem Schritt werden die Lademodule `LOGPROG` und `EDITPROG` in den Dataset `NATURAL.USER.LOAD` gestellt.

Mit einem zusätzlichen Link-Edit-Job können diese Module als ein einziges Lademodul miteinander verlinkt werden, bevor sie in Schritt 5 mit dem Nukleus verlinkt werden.

```
//JOBCARD JOB (ACCTING),CLASS=A,MSGCLASS=X
//*
//* OPTIONAL JOB TO LINK CATALOGED OBJECTS TOGETHER
//*
//LINK2    EXEC PGM=IEWL,PARM='LIST,LET,XREF,NCAL,RENT,REUS'
//SYSLMOD  DD DSN=NATURAL.USER.LOAD,DISP=SHR
//SYSUT1   DD UNIT=SYSDA,SPACE=(1024,(200,20))
//SYSPRINT DD SYSOUT=X
//SYSLIN   DD *
INCLUDE SYSLMOD(LOGPROG)  LOGON NATURAL PGM
INCLUDE SYSLMOD(EDITPROG) EDITOR NATURAL PGM
NAME XXXXXX(R)
/*
```

4. Definieren Sie die statisch verlinkten Natural-Programme im Quellcode-Modul `NATCONFG` in der Tabelle `NSTATIC` für verlinkte Natural-Programme:

```
NTSTAT INPL,TYPE=W
NTSTAT INPLLIB,TYPE=W
NTSTAT AERROR,TYPE=W
NTSTAT LOGPROG          <==== Ihre Einträge
NTSTAT EDITPROG         <====
```

`TYPE=W` bedeutet, dass eine „schwache“ externe Referenz auf das angegebene Programm anstelle einer normalen Referenz erzeugt wird.

5. Überprüfen Sie den Jobfluss des Linkage Editor für den Natural-Nukleus und fügen Sie Folgendes hinzu:

```
//*
//* INCLUDE DDNAME AND DSN OF DATASET WHERE OBJECTS RESIDE
//*
//SYSLMOD DD DSN=NATURAL.USER.LOAD,DISP=SHR
//NATLIB DD DSN=NATURAL.USER.LOAD,DISP=SHR//*
//SYSLIN DD*
...
...                                 INCLUDE MODULES FOR NUCLEUS
...
INCLUDE NATLIB(nat-parm-module)     NATURAL PARAMETER MODULE
INCLUDE SYSLMOD(LOGPROG)            LOGON NATURAL PGM
INCLUDE SYSLMOD(EDITPROG)           EDITOR NATURAL PGM
...
...                                 INCLUDE ENTRY AND NAME CARDS
...
/*
```

*`nat-parm-module`* steht für den Namen des **Natural-Parametermoduls**.

Wenn die katalogisierten Objekte miteinander verlinkt wurden (wie in Schritt 3 optional geschehen), fügen Sie dieses Lademodul anstelle der einzelnen Lademodule in den Link des Nukleus ein.

# 3 Natural User Exits


- NATUEX1 - User Exit für die Berechtigungskontrolle ..... 16
- NATSREX2 und NATSREX3 - User Exits für die Sortierverarbeitung ..... 17
- NATUSKnn - User Exit zur Berechnung von Sortierschlüsseln ..... 18
- NATPM - User Exit für invertierte Ausgabe ..... 19
- NREXPG - User Exit für NATRJE ..... 20
- USR0070P - User Exit für Editor-Profile ..... 21
- USR2002P - User Exit für Textstrings im Hilfefenster ..... 21
- USR2003P - User Exit für Hauptmenü ..... 22

Ein Natural User Exit ist ein Objekt, das von Natural, einer Unterkomponente oder einem Unterprodukt aufgerufen wird. In der Regel wird ein Beispiel-User Exit in Quellcodeform geliefert. Die im User Exit enthaltenen Anweisungen müssen vom Benutzer selbst geschrieben oder angepasst werden. Der Zweck eines User Exit besteht darin, Daten zu beeinflussen oder Entscheidungen zu treffen. Die meisten User Exits nutzen die Vorteile der Programmiersprache Natural. Eine kleine Teilmenge muss in Assembler-Sprache geschrieben werden.

Weitere Natural User Exits und Anwendungsprogrammierschnittstellen werden an den entsprechenden Stellen in der Dokumentation der Natural-Unterkomponenten oder Unterprodukte beschrieben (*Natural RPC*, *TP-Monitor-Schnittstellen*, Dienstprogramme (Utilities), Add-on-Produkte usw.).

## NATUEX1 - User Exit für die Berechtigungskontrolle

Der User Exit NATUEX1 wird immer dann aufgerufen, wenn eine Benutzersitzung aktiviert wird. Er kann verwendet werden, um festzustellen, ob der Benutzer berechtigt ist, Natural zu benutzen oder nicht. Die Security-Daten, mit denen dies festgestellt wird, können aus dem verwendeten Security-System (z.B. RACF oder ACF2) abgerufen werden.

`NATUEX1` wird unter Verwendung von Standardaufrufkonventionen aufgerufen:

| Register | Inhalt |
|---|---|
| 15 | Einsprungadresse von NATUEX1 |
| 14 | Rücksprungadresse von Natural |
| 13 | Adresse eines Speicherbereichs von 18 Worten |
| 1 | Adresse einer Parameterliste |

Die Parameterliste enthält fünf Adressen:

| Adresse | Zeigt auf ein 8-Byte-Feld, das den Wert enthält, der zum Füllen folgender Natural-Systemvariablen verwendet wird: |
|---|---|
| 1 | `*INIT-USER` |
| 2 | `*ETID` |
| 3 | `*INIT-ID` |
| 4 | `*INIT-PROGRAM` |
| 5 | `*USER` (Beachten Sie, dass diese Systemvariable bei einer Natural Security-Anmeldung überschrieben wird.) |

Diese fünf Werte können durch den User Exit geändert werden.

Für eine normale Beendigung muss der User Exit die Kontrolle mit dem auf `0` gesetzten Register 15 zurückgeben. Wenn Register 15 nicht `0` enthält, wird die Natural-Sitzung mit dem Condition Code beendet, der dem Wert in Register 15 entspricht.

NATUEX1 kann mit dem umgebungsunabhängigen Nukleus oder mit einem umgebungsabhängigen Nukleus verlinkt werden. Es ist auch möglich, ihn mit einem alternativen Natural-Parametermodul oder als separates Modul zu verlinken, wenn Sie mit dem Profilparameter `RCA` arbeiten.

Ein Beispiel für den User Exit ist als Member `XNATUEX1` in der Natural Source Library verfügbar.

**Für CICS:** Siehe auch *NCIUIDEX - Benutzerkennung-Exit-Schnittstelle* in der Natural-*TP-Monitor-Schnittstellen*-Dokumentation.

## NATSREX2 und NATSREX3 - User Exits für die Sortierverarbeitung

Natural bietet zwei User Exits für die Sortierverarbeitung: `NATSREX2` und `NATSREX3`.

Die beiden User Exits können sowohl mit dem Natural-eigenen Sortierprogramm als auch mit einem externen Sortierprogramm verwendet werden. Die Exits werden automatisch aktiviert, wenn sie mit dem Nukleus verlinkt sind, so dass ihre Adressen aufgelöst werden. Da viele externe SORT-Programme bereits mehrere Exit-Funktionen bereitstellen, können die Exits `NATSREX2` und `NATSREX3` vor allem mit dem internen Sortierprogramm von Natural verwendet werden.

`NATSREX2` wird immer dann aufgerufen, wenn Natural einen Datensatz an das Sortierprogramm übergibt. `NATSREX3` wird aufgerufen, wenn das Sortierprogramm nach Beendigung des Sortierlaufs einen Datensatz an Natural übergibt. Das ausgelieferte Beispiel zeigt, wie Sie Ihre eigene Sortierreihenfolge für einen `SORT`-Lauf festlegen können.

Bei der Aktivierung der User Exits müssen die folgenden Registerkonventionen eingehalten werden:

| Register | Inhalt |
|---|---|
| 15 | Einsprungadressen `NATSREX2` und `NATSREX3` |
| 14 | Rücksprungadresse von Natural |
| 13 | Adresse des 18-Wort-Speicherbereichs |
| 1 | Adresse des Sortiersatzes |
| 3 | Länge des Sortiersatzes |

Die User Exits müssen die Natural-Register sichern und sie bei der Rückgabe der Kontrolle an Natural wiederherstellen.

Da das Sortier-Exit-Modul mit dem Modul `NAT2SORT` verlinkt ist, muss die Programmierung reentrant sein. Das Format und die Struktur der Sortiersätze dürfen nicht verändert werden.

# NATUSKnn - User Exit zur Berechnung von Sortierschlüsseln

Einige Landessprachen enthalten Zeichen, die von einem Sortierprogramm oder Datenbanksystem nicht in der richtigen alphabetischen Reihenfolge sortiert werden. Mit der Systemfunktion SORTKEY können Sie solche „falsch sortierten“ Zeichen in andere Zeichen umwandeln, die alphabetisch „richtig sortiert“ werden.

Wenn Sie die Funktion SORTKEY in einem Natural-Programm verwenden, wird der User Exit NATUSK*nn* aufgerufen, wobei *nn* der aktuelle Sprachcode ist (d.h. der aktuelle Wert der Systemvariablen *LANGUAGE).

Ein User Exit NATUSK*nn* kann in jeder Programmiersprache geschrieben werden, die eine Standard-CALL-Schnittstelle bietet. Die mit SORTKEY angegebene Zeichenkette wird an den User Exit übergeben. Der User Exit muss so programmiert werden, dass er „falsch sortierte“ Zeichen in dieser Zeichenkette in entsprechende „richtig sortierte“ Zeichen umwandelt. Die konvertierte Zeichenkette wird dann im Natural-Programm zur Weiterverarbeitung verwendet.

Für die Konvertierung kann NATUSK*nn* die Umsetzungstabelle NTUTAB1 des Konfigurationsmoduls NATCONFG verwenden, d.h. NTUTAB1 muss ggf. entsprechend angepasst werden.

NATUSK*nn* wird mit den Standard-Aufrufkonventionen aufgerufen:

| Register | Inhalt |
|---|---|
| 15 | Einsprungadresse von NATUSK*nn* |
| 14 | Rücksprungadresse von Natural |
| 13 | Adresse eines Speicherbereichs von 18 Vollwörtern |
| 1 | Adresse einer Parameterliste |

Die Parameterliste enthält die folgenden Adressen:

| Offset | Adresse von |
|---|---|
| +0 | Die von Natural übergebene Zeichenkette. |
| +4 | Die Länge der Zeichenkette (Vollwort). |
| +8 | Die aus der Umwandlung resultierende Zeichenkette. |
| +12 | Die Länge der Ergebniszeichenkette (Vollwort). |
| +16 | Die Umsetzungstabelle NTUTAB1. |

NATUSK*nn* muss alle Register mit Ausnahme von 14 und 15 sichern und sie bei der Rückgabe der Kontrolle an Natural wiederherstellen.

Für eine normale Beendigung muss der User Exit die Kontrolle mit Register 15 auf Return Code 0 zurückgeben. Wenn Register 15 nicht `0` enthält, wird ein entsprechender Natural-Fehler ausgegeben.

**Beispiele für User Exit-Programme**

Die folgenden Beispiele für User Exits liegen im Quellcode vor:

| Programm | Funktion |
|---|---|
| `NATUSK01` | Gilt für Englisch und wandelt alle englischen Kleinbuchstaben in der Zeichenkette in Großbuchstaben um. |
| `NATUSK02` | Gilt für Deutsch und wandelt die deutschen Umlaute ä, ö, ü und ß in die entsprechenden Ersatzzeichen ae, oe, ue, ss um, um eine andere Sortierreihenfolge zu erreichen. |

`NATUSK`*nn* kann mit dem umgebungsunabhängigen Nukleus oder mit einem umgebungsabhängigen Nukleus verlinkt werden. Es ist auch möglich, ihn mit einem alternativen Natural-Parametermodul zu verlinken, oder als separates Modul, wenn Sie mit dem Profilparameter `RCA=NATUSK`*nn* arbeiten.

Weitere Informationen zu Verlinkungs- und Ladekonventionen sind beim `CALL`-Statement in der Natural-*Statements*-Dokumentation vorhanden.

## NATPM - User Exit für invertierte Ausgabe

Das Modul `NATPM` wird zur Unterstützung von Terminals mit umgekehrter Schreibrichtung verwendet. Es enthält die User Exit-Routine für die Feld- und Zeilenumsetzung, die von Natural bei Terminal-Ein-/Ausgaben aufgerufen wird, wenn für einige Felder der Druckmodus (Profilparameter `PM` auf `I` gesetzt wurde.

`PM=I` steht für inverse Richtung und wird verwendet, um Sprachen zu unterstützen, die von rechts nach links schreiben (z.B. bidirektionale Sprachen). Siehe auch die Beschreibung des Profilparameters `PM`.

Das Modul `NATPM` wird als Quellcode-Modul ausgeliefert und kann bei Bedarf geändert werden.

### Invertierungslogik

Natural stellt eine User Exit-Routine zur Verfügung, die für jedes Feld, dessen Ergebnisattribut `PM=I` ist, und für jede Zeile, die über Hardcopy, zusätzlichen Report und Primär-Batch-Ausgabe gedruckt werden soll, aufgerufen wird. Dieser Exit wird mit drei Parametern aufgerufen:

- das Quellfeld, das invertiert werden soll,
- das Zielfeld, das die invertierten Daten erhalten soll,
- ein Längenfeld, das die Länge der Quell- und Zielfelder angibt.

Da diese User Exit-Routine als Quellcode für alle Benutzer verfügbar ist, kann sie als expliziter Feld-Exit verwendet werden, der durch das Attribut `PM=I` ausgelöst wird. Der Benutzer kann dann Zeilen- oder Feldinhalte prüfen und ändern.

### Feld-User Exit

Der User Exit in `NATPM` wird für jedes Feld aufgerufen, für das das Attribut `PM=I` gesetzt ist.

Dieses Attribut kann vom Natural-Programmierer gesetzt werden oder wird für numerische Felder automatisch gesetzt, wenn der globale Druckmodus auf `PM=I` eingestellt ist. Dabei spielt es keine Rolle, ob die Ausgabe für das Terminal, für den Ausdruck, für zusätzliche Reports oder für die Primärausgabe im Batch erzeugt wird.

Bei Druckgeräten erwartet Natural keine automatische Invertierung durch die Hardware, sondern ruft `NATPM` erneut für die gesamte Zeile auf. Diese Funktion kann in Ländern verwendet werden, in denen die Feldinvertierung nicht erforderlich ist, um eine Schnittstellenlogik mit Natural auf der Grundlage eines Feldattributs einzurichten.

## NREXPG - User Exit für NATRJE

`NREXPG` ist ein User Exit für Natural Remote Job Entry (NATRJE). Nach Beendigung des Jobs wird jede JCL-Karte an den Exit übergeben, bevor sie an das Betriebssystem übermittelt wird. Dem Exit stehen die folgenden Daten zur Verfügung:

- die zu übergebende JCL-Karte,
- ein Rückgabecode-Feld,
- der Name des Natural-Programms, das gerade ausgeführt wird,
- die Natural-Benutzerkennung,
- ein 240 Byte großer Arbeitsbereich.

Nach jedem Aufruf übergibt der Exit einen Rückgabecode an NATRJE, der eines der folgenden Ereignisse anzeigt:

| Code | Erläuterung |
|---|---|
| 0 | Übergabe: Die Karte wird übermittelt; der Exit kann die Karte vor der Übergabe ändern. |
| 4 | Beendigung: Die Karte wird übermittelt; der Exit ist für weitere Karten des aktuellen Jobs gesperrt. |
| 8 | Einfügung: Die Karte wird übersprungen, da davon ausgegangen wird, dass sie nur ein Einfügezeichen, z. B. das Prozentzeichen (%), enthält; weitere angegebene Karten werden übermittelt. |
| 10 | Löschung: Die Karte wird nicht übermittelt. |
| 12 | Der aktuelle Job wird geleert (flush). |

Ein Beispiel für den User Exit mit der Bezeichnung NREXPG ist als Member XNATRJE in der Natural Source Library verfügbar. Der Exit kann nach den Regeln der als CSTATIC angegebenen Programme assembliert und verlinkt werden. Ein CSTATIC-Eintrag für NREXPG ist jedoch nicht erforderlich.

## USR0070P - User Exit für Editor-Profile

Mit der User Exit-Routine USR0070P können Sie die Parametereinstellungen für den Natural-Programmeditor oder den Natural-Datenbereichseditor (Data Area Editor) im Standardprofil SYSTEM ändern.

Weitere Informationen zum Editor-Profil finden Sie unter *Editoren - Allgemeine Informationen* in der *Editoren*-Dokumentation.

USR0070P liefert eine Liste aller Parameter, die eine Standard-Einstellung erhalten sollen.

Mit diesem User Exit können Sie auch festlegen, ob Editor-Profile in der Systemdatei FNAT, in der Systemdatei FUSER oder in der Scratch-Pad-Datei abgelegt werden sollen.

Außerdem berücksichtigt USR0070P die DBCS-Unterstützung und setzt die Editor-Profil-Optionen **Editieren in Kleinbuchstaben** und **Dynamische Umwandlung von Kleinbuchstaben** entsprechend.

Ein Beispiel für diese User Exit-Routine ist in der Library SYSEXT in der Systemdatei FNAT sowohl in Objekt- als auch in Quellcodeform verfügbar. Informationen zu ihrer Verwendung sind im Textobjekt USR0070T enthalten.

## USR2002P - User Exit für Textstrings im Hilfefenster

Die User Exit-Routine USR2002P kann verwendet werden, um die Textstrings für das Fenster **Current Natural Message** anzupassen, das durch Drücken der Hilfetaste aufgerufen wird, während sich der Cursor in der Nachrichtenzeile befindet.

Das Objekt USR2002P selbst enthält die Texte, die im Fenster **Current Natural Message** verwendet werden, z.B. den Fenstertitel und die beschreibenden Texte, wie die Feldnamen Sh (Kurzmeldung), Tx (Langmeldung), Ex (Erklärung) und Ac (Aktion).

Ein Beispiel für diese User Exit-Routine ist in der Library SYSEXT in der Systemdatei FNAT sowohl in Objekt- als auch in Quellcodeform verfügbar. Informationen zu ihrer Verwendung sind im Textobjekt USR2002T enthalten.

## USR2003P - User Exit für Hauptmenü

Mit der User Exit-Routine USR2003P können die folgenden Einstellungen für das Natural-Hauptmenü (Main Menu) und die untergeordneten Menüs angepasst werden:

- Position und Farbe der Nachrichtenzeile,
- Position und Farbe der PF-Tastenzeilen.

Ein Beispiel für diese User Exit-Routine ist in der Library SYSEXT in der Systemdatei FNAT sowohl in Objekt- als auch in Quellcodeform verfügbar. Informationen zu ihrer Verwendung sind im Textobjekt USR2003T enthalten.

# 4 Natural User Access Method für Druck- und Arbeitsdateien


- Beschreibung des Moduls NATAMUSR ..... 24
- Installation des Moduls NATAMUSR ..... 24
- Aufrufen des Drittanbieterprodukts ..... 25

Dieses Kapitel beschreibt die Natural User Access Method, die eine Schnittstelle für Produkte von Drittanbietern zur Unterstützung von Natural-Druck- und/oder-Arbeitsdateien darstellt.

## Beschreibung des Moduls NATAMUSR

Das Modul `NATAMUSR` bietet eine Exit-Schnittstelle für Software-Hersteller (Einstiegspunkt `NATAM9EX`) für die Behandlung von Natural-Druck- und -Arbeitsdateien. Es besteht eigentlich aus zwei Teilen:

- dem mit Natural ausgelieferten Natural User Access Method Stub `NATAMUSR` und
- dem Natural User Access Method-Exit `NATAM9EX`, der von einem Software-Hersteller geliefert wird.

## Installation des Moduls NATAMUSR

Das Modul `NATAMUSR` (mit dem Access Method Exit) kann auf eine der folgenden Arten installiert werden:

- verlinkt mit dem umgebungsunabhängigen Nukleus,
- verlinkt mit dem umgebungsabhängigen Nukleus,
- verlinkt mit einem alternativen Natural-Parametermodul (wie über den Profilparameter `PARM` geladen),
- verlinkt als separates Modul. In diesem Fall sind die folgenden Natural-Profilparameter erforderlich:

  ```
  RCA=(NATAM09),RCALIAS=(NATAM09,xxx),
  ```

  wobei *xxx* der Name des separaten Moduls in der Load Library ist.

Der umgebungsunabhängige Nukleus und der umgebungsabhängige Nukleus sind beschrieben im Kapitel *Komponenten des Natural-Nukleus* in der *Installation*-Dokumentation.

## Aufrufen des Drittanbieterprodukts

**Um das Drittanbieterprodukt für die Natural-Druck- und/oder -Arbeitsdateiverarbeitung aufzurufen:**

- Geben Sie `AM=USER` für die entsprechenden Dateien an (siehe Makros `NTPRINT` und `NTWORK`).

Einzelheiten zur Installation der Natural User Access Method und weitere Informationen über den Exit-Handler des Drittanbieters finden Sie in der Dokumentation des jeweiligen Software-Herstellers.

# 5 Natural-Systemdateien


- Natural-Scratch-Pad-Datei ..... 28

In der folgenden Tabelle sind die Natural-Systemdateien aufgeführt und beschrieben, die normalerweise in einer Natural-Umgebung verfügbar sind. Die Verfügbarkeit der Systemdateien und der in den Dateien enthaltenen Daten hängt von den Produkten ab, die zusätzlich zum Basisprodukt Natural installiert sind.

Die Einstellungen für die Systemdateien werden mit gleichnamigen Natural-Profilparametern definiert (Ausnahme: **Scratch Pad File**). Sie können den Hyperlinks in der folgenden Tabelle folgen, um Details zu diesen Parametern in der Parameter-Referenz-Dokumentation zu lesen.

| Systemdatei bzw. Parameter | Verfügbar mit | Datei-Inhalt |
|---|---|---|
| FNAT | Basis-Natural | Alle für Natural-Systemanwendungen benötigten Objekte. |
| FUSER | Basis-Natural | Für Benutzeranwendungen benötigte benutzerspezifische Objekte. |
| FDIC | Basis-Natural | Natural-DDMs (Datendefinitionsmodule).<br><br>Wenn Predict installiert ist, enthält FDIC außerdem noch Daten für das Predict-Datendiktionärsystem.<br><br>Wenn der Natural Development Server installiert ist, enthält FDIC außerdem noch Anwendungsdaten und Informationen über die Sperrung von Objekten. |
| FSEC | Natural Security | Für Sicherheitseinstellungen benötigte Steuerungs-/Überwachungsinformationen. |
| FSPOOL | Natural Advanced Facilities | Steuerungs- und Spool-Informationen, die zum Ausgeben eines Reports auf dem Bildschirm oder einem Drucker und zum Erstellen von statistischen Druckinformationen benötigt werden. |
| Scratch-Pad | Basis-Natural | Daten die nicht explizit als Natural-Objekte in einer anderen Systemdatei gespeichert sind. |
| FPROF | Basis-Natural | Parameterprofile, die mit dem Profilparameter PROFILE angegeben werden, vorausgesetzt dass keine Datenbankinformationen als Subparameter von PROFILE geliefert werden. |
| FREG | Basis-Natural | Registry-Daten, die nicht explizit in einer anderen Systemdatei gespeichert werden. |

## Natural-Scratch-Pad-Datei

Die Natural-Scratch-Pad-Datei dient zum Speichern von Aufzeichnungen und Bildschirmabzügen, die nicht explizit als Natural-Objekt in der Natural-Systemdatei FNAT oder FUSER gespeichert werden können.

Im Gegensatz zu FNAT und FUSER ist eine Scratch-Pad-Datei in einer Natural-Sitzung nicht zwingend erforderlich. Sie müssen jedoch eine Scratch-Pad-Datei definieren, wenn Sie mit

schreibgeschütztem Zugriff auf Systemdateien arbeiten (Profilparameter `ROSY=ON`). Andernfalls können die Aufzeichnungen und Bildschirmabzüge nicht gespeichert werden und es wird eine entsprechende Fehlermeldung (NAT0106) ausgegeben. Die Scratch-Pad-Datei ist vom schreibgeschützten Zugriff ausgeschlossen.

Eine vernünftige Schätzung des damit verbundenen Speicherbedarfs ist kaum möglich, da der von der Recording Utility und vom Dienstprogramm `NATPAGE` (für Bildschirmabzüge) verwendete Speicherplatz nicht im Voraus berechnet werden kann. Die Größe der Scratch-Pad-Datei, die an Ihrem Standort benötigt wird, kann jedoch geschätzt werden, wenn Sie die Arten von Datensätzen, die darauf gespeichert werden, besser kennen. Der Inhalt der Scratch-Pad-Datei wird im folgenden Abschnitt beschrieben:

- Recordings - Aufzeichnungen
- Bildschirmaufnahmen - NATPAGE
- Dateiverwaltung

**Verwandtes Thema:**

*Scratch-Pad-Datei definieren* in der *Installation*-Dokumentation.

## Recordings - Aufzeichnungen

Die Recording Utility wird über Terminalkommandos aktiviert. Übersicht siehe *Bildschirme aufzeichnen* in der *Terminalkommandos*-Dokumentation. Die mit dieser Utility erfolgenden Aufzeichnungen von Abläufen oder Mitschnitte von Eingaben werden wie Natural-Quellcodeprogramme (oder andere Objekttypen) gespeichert. Die Größe einer Aufzeichnung hängt davon ab, wie viele Bildschirmeingaben während einer Aufzeichnungssitzung getätigt wurden. Aufzeichnungen sind wie Programme, die mit einer Library verbunden sind.

Derzeit ist es nicht möglich, Aufzeichnungen in der Scratch-Pad-Datei mit dem Natural-Systemkommando `LIST` aufzulisten. Das Dienstprogramm SYSMAIN kann jedoch verwendet werden, um die in der Scratch-Pad-Datei gespeicherten Aufnahmen aufzulisten und zu verwalten. Um die Aufnahmen in der FNAT/FUSER-Datei statt in der Scratch-Pad-Datei zu speichern, müssen Sie den Profilparameter `RFILE` setzen.

Aufzeichnungen, die in der Systemdatei FNAT oder FUSER gespeichert sind, werden durch Transaction Backouts (BTs), die in den Anwendungsprogrammen des Benutzers ausgegeben werden, beeinflusst (unterbrochen). Dies ist ein sehr häufig auftretendes Problem für Benutzer der Aufzeichnungsfunktion, das durch die Verwendung der Scratch-Pad-Datei vermieden werden kann.

### Bildschirmaufnahmen - NATPAGE

Das Dienstprogramm NATPAGE kann verwendet werden, um Bildschirmbilder (in chronologischer Reihenfolge ihres Auftretens) in der Scratch-Pad-Datei zu speichern. NATPAGE kann mit dem Terminalkommando %P aktiviert werden. Ab dem Zeitpunkt, an dem %P abgesetzt wird, werden alle Bildschirme, die dem Endbenutzer präsentiert werden, in der Scratch-Pad-Datei gespeichert (sofern diese für Ihre Sitzung definiert wurde), bis das Terminalkommando %O eingegeben wird. Die erfassten Bildschirme können mit dem Terminalkommando %E angezeigt werden.

Für jedes Bildschirmbild wird der aktuelle Inhalt des Seitenpuffers und des Seitenattributpuffers gespeichert. Das bedeutet, dass die Menge der gespeicherten Daten von den Einstellungen der Profilparameter PS/LS für die Sitzung und natürlich von der Anzahl der Bildschirmbilder abhängt. Die Anzahl der möglichen Bilder pro Benutzersitzung hängt vom Profilparameter PD ab (Voreinstellung ist 50, gültige Werte sind 0 - 255).

Die Größe des Seitenpuffers kann wie folgt berechnet werden:

```
PS * LS
```

Die Größe des Seitenattributpuffers wird dynamisch ermittelt.

### Dateiverwaltung

Die Scratch-Pad-Datei muss nicht gewartet werden, sofern sie ausreichend groß ist.

- Aufzeichnungen in der Scratch-Pad-Datei können mit Hilfe des Dienstprogramms SYSMAIN gelöscht, kopiert, verschoben und aufgelistet werden.
- Erfasste Bildschirme können mit dem Terminalkommando %E gelöscht werden.
- Gespeicherte Bildschirmbilder können jedoch in Natural nicht verwaltet werden.

Der Speicherplatz in der Scratch-Pad-Datei kann wiedergewonnen werden, indem sie in Zeiten der Inaktivität mit Adabas-Dienstprogrammen aktualisiert wird, ohne dass dies Auswirkungen auf nachfolgende Natural-Sitzungen hat, die die Scratch-Pad-Datei verwenden.

# 6 Natural-Text-Module und Makros


- Funktion und Verwendung von Textmodulen ..... 32
- NATTEXT - Natural-Schlüsselwort-Definitionen ..... 32
- NATTXT2 - Ausgabetext, Schlüsselwörter und Benutzerabbruchmeldungen (in Groß- un Kleinschreibung) ..... 33
- NATTXT2U - Ausgabetext, Schlüsselwörter und Benutzerabbruchmeldungen (in Großbuchstaben) ..... 36
- NATTXT3 - Textfragmente für Platzhalter in Natural-Fehlermeldungen ..... 36
- NTERMSG - Natural-Beendigungsmeldungen und Rückgabecodes ..... 37

In diesem Kapitel enthält Beschreibungen der Natural-Textmodule NATTEXT, NATTXT2, NATTXT2U, NATTXT3 und das Natural-Makro NTERMSG.

## Funktion und Verwendung von Textmodulen

Alle Natural-Schlüsselwörter, alternative Schlüsselwörter und Standard-Ausgabetexte sind in den Modulen NATTEXT und NATTXT2 enthalten. Natural-Systemkommandos und alternative Systemkommandos sind ebenfalls als Schlüsselwörter und alternative Schlüsselwörter in diesen Modulen enthalten. Ersetzungstextfragmente für Natural-Fehlermeldungen sind im Modul NATTXT3 enthalten. Die Module sind als Quellcode in der Natural Source Library und als Lademodul in der Natural Load Library enthalten.

Bei Bedarf können Sie Natural-Schlüsselwörter, alternative Schlüsselwörter und den in diesen Modulen enthaltenen Text ändern. Zum Beispiel können Natural-Sitzungsabbruchmeldungen von Englisch in eine andere Sprache geändert werden, Natural-Schlüsselwörter können deaktiviert oder Synonyme hinzugefügt werden.

Wenn Änderungen an einem NATTEXT-, NATTXT2- oder NATTXT3-Modul vorgenommen werden, muss jedes geänderte Modul assembliert, verlinkt und in das ausführbare Natural-Modul eingebunden werden. Siehe die entsprechende Natural-*Installation*-Dokumentation.

## NATTEXT - Natural-Schlüsselwort-Definitionen

Das Modul NATTEXT enthält die Makros NTKEY, NTALT und NTSYN für jedes Schlüsselwort und alternative Schlüsselwort, das von Natural erkannt werden soll.

### NATTEXT ändern

**Vorsicht:** Es wird empfohlen, das Modul NATTEXT nur aus sehr wichtigen Gründen zu modifizieren, da es nach einer Änderung nicht mehr ordnungsgemäß vom Support gewartet werden kann.

Es gelten die folgenden Regeln:

- Ein Schlüsselwortwert für ein NTKEY- oder NTALT-Makro kann geändert werden, indem der aktuelle Schlüsselwortwert durch den gewünschten Wert ersetzt wird.
- Ein Schlüsselwort oder alternatives Schlüsselwort kann deaktiviert werden, indem der Schlüsselwortwert durch das Prozentzeichen (%) ersetzt wird.
- Die Position jedes NTKEY- und NTALT-Makros innerhalb des Moduls ist fest und darf nicht verschoben werden. Zusätzliche NTKEY- und NTALT-Makros dürfen nicht eingefügt werden.

- Mit dem Makro NTSYN können für jedes Schlüsselwort oder alternative Schlüsselwort Synonyme zugewiesen werden. Ein oder mehrere NTSYN-Makros können nach einem NTKEY- oder NTALT-Makro eingefügt werden. Das Makro NTSYN enthält einen Parameter, nämlich den Wert, der als Synonym verwendet werden soll. Wenn das Synonym eingebettete Leerzeichen enthält, muss der gesamte Wert in Hochkommata eingeschlossen werden.

### Beispiel für das Ändern des NATTEXT-Moduls

Das folgende Beispiel veranschaulicht, wie ein NATTEXT-Modul geändert wird. In diesem Beispiel

- soll das Synonym RECHERCHE für das Schlüsselwort FIND verwendet werden,
- das Synonym LISEZ soll für das alternative Schlüsselwort BROWSE verwendet werden,
- die Schlüsselwörter GET und HISTOGRAM sollen deaktiviert werden.

NATTEXT **vor** Änderung:

```
STATNAM NTKEY FIND
        NTALT BROWSE
        NTALT GET
        NTALT ACCEPT
        NTALT REJECT
        NTALT HISTOGRAM
```

NATTEXT **nach** Änderung:

```
STATNAM NTKEY FIND
        NTSYN RECHERCHE
        NTALT BROWSE
        NTSYN LISEZ
        NTALT %
        NTALT ACCEPT
        NTALT REJECT
        NTALT %
```

## NATTXT2 - Ausgabetext, Schlüsselwörter und Benutzerabbruchmeldungen (in Groß- un Kleinschreibung)

Das Modul NATTXT2 enthält die Makros NTKEYT, NTALTT und NTSYNT, die Folgendes definieren:

- Standard-Natural-Ausgabetexte
- Schlüsselwörter und alternative Schlüsselwörter für Natural Systemkommandos und Dienstprogramme (Utilities)

- Benutzerdefinierte Abbruchmeldungen

### Standard-Natural-Ausgabetexte

Das Modul NATTXT2 enthält die folgenden Standard-Natural-Ausgabetexte, die jeweils auch in einer anderen Sprache ausgegeben werden können, wenn der Sprachcode entsprechend gesetzt ist (siehe auch unten):

- das Literal Page (Seite), das im Seitenkopf der Standardausgabe verwendet wird,
- der Name jedes Monats, wie er in der Natural-Systemvariablen *DATG (gregorianisches Datum) und in den Datums-Eingabemasken (L) verwendet wird, und der Name jedes Tages, wie er in den Datums-Editiermasken (N) verwendet wird,
- die Meldung ENTER INPUT DATA und die Skelett-Fehlermeldungen für die Fehlernummern 1104, 1105 und 1106 (die bei der Online-Eingabeverarbeitung verwendet werden),
- die Fehlermeldung, die verwendet wird, wenn eine Systemdatei nicht geöffnet werden konnte (die nicht aus der Systemdatei abgerufen werden kann); eine Fehlernummer der Form NAT8*xxx* (wobei *xxx* der dezimale Adabas-Antwortcode ist) wird dieser Fehlermeldung durch Natural hinzugefügt,
- die Konstanten More, Top und Bottom, die in Fenstern für die Anzeige von Positionsinformationen in Textform verwendet werden,
- die Tabelle zur Definition von Reports und die Report-Behandlung von Reports > 33.

Alle in NATTXT2 enthaltenen Werte können geändert werden, indem der aktuelle Text durch den gewünschten Text ersetzt wird. Wenn ein Monatsnamen-Synonym länger als neun Zeichen ist, werden nur die ersten neun Stellen von der Systemvariablen *DATG verwendet.

NTSYNT-Makro-Statements können wie für das Modul NATTEXT beschrieben hinzugefügt werden. Mit NATTXT2 kann jedoch ein zweiter Parameter angegeben werden. Dieser Parameter ist optional und steht für das Sprachkennzeichen, das für das Synonym verwendet werden soll. Die Angabe des Sprachkennzeichens bewirkt, dass Natural Meldungen ausgibt, die aus der Verwendung dieses Synonyms in der entsprechenden Sprache resultieren. Wenn Fehlermeldungstexte in der Natural-Systemdatei unter Verwendung eines anderen Sprachkennzeichens als 1 (dem Standardwert für Englisch) gespeichert wurden, werden Fehlermeldungen in der entsprechenden Sprache ausgegeben. Informationen darüber, welcher Sprachcode für welche Sprache steht, finden Sie beim Profilparameter ULANG.

### Schlüsselwörter und alternative Schlüsselwörter für Natural Systemkommandos und Dienstprogramme (Utilities)

Das Modul NATTXT2 enthält NTKEYT- und NTALTT-Makros für jedes Schlüsselwort und jedes alternative Schlüsselwort, das von Natural für die folgenden Natural-Systemkommandos und -Dienstprogramme erkannt werden soll, sowie für die Parameter der Kommandos und gegebenenfalls deren Werte. Jedes dieser Makros kann auch in einer anderen Sprache verwendet werden, wenn der Sprachcode entsprechend eingestellt ist (siehe auch unten):

- alle Natural-Systemkommandos im Allgemeinen,
- für das Systemkommando GLOBALS die Parameter und ggf. deren Werte,
- für das Systemkommando COMPOPT die Parameter und ggf. deren Werte,
- allgemein zugängliche Systemkommandos (diese Systemkommandos sind permanent gültig und können weder durch Natural Security noch durch den Natural-Profilparameter NC gesperrt werden,)
- Natural-Dienstprogramme (Utilities).

Die Statements der Makros NTKEYT und NTALTT können ähnlich wie die Statements der Makros NTKEY und NTALT verwendet werden, wie für das Modul NATTEXT beschrieben.

Die Makro-Statements NTSYNT können wie unter ***Standard-Natural-Ausgabetexte*** beschrieben verwendet werden.

### Benutzerdefinierte Abbruchmeldungen

Mit dem Makro NTERMSG können benutzerdefinierte Abbruchmeldungen für alle Rückgabewerte (1 - 255) hinzugefügt werden, die mit einem TERMINATE-Statement ausgegeben werden können und die normalerweise zu der Natural-Abbruchmeldung NAT9987 führen.

Mit dem ersten Parameter wird der Text der Abbruchmeldung und mit dem zweiten Parameter der entsprechende Rückgabecode angegeben.

**Beispiel:**

```
NTERMSG 'USR0077 THIS IS A SAMPLE USER MESSAGE FOR RETURN CODE 77',77
```

Ein Statement TERMINATE 77 in einer Natural-Anwendung führt zu der folgenden Abbruchmeldung: USR0077 THIS IS A SAMPLE USER MESSAGE FOR RETURN CODE 77.

## NATTXT2U - Ausgabetext, Schlüsselwörter und Benutzerabbruchmeldungen (in Großbuchstaben)

Das Modul NATTXT2U enthält die gleichen Elemente wie das Modul NATTXT2. Der Unterschied besteht darin, dass bestimmte Schlüsselwörter für die englische Sprache in NATTXT2 in gemischter Großschreibung enthalten sind, während sie in NATTXT2U in Großbuchstaben geschrieben sind. Dies betrifft die Schlüsselwörter MORE, TOP, BOTTOM, PAGE und alle Monats- und Wochentagsnamen.

NATTXT2U sollte in Umgebungen, in denen die Kleinbuchstaben-Codepunkte H'81' bis H'A9' zur Darstellung nationaler Zeichen verwendet werden, anstelle von NATTXT2 mit dem Natural-Nukleus verlinkt werden, z. B. wenn die Codepage 930 mit Katakana-Zeichen halber Breite verwendet wird.

## NATTXT3 - Textfragmente für Platzhalter in Natural-Fehlermeldungen

Das Modul NATTXT3 enthält die Makros zur Definition von Textfragmenten, die in Natural-Fehlermeldungen den Platzhalter :*n*: ersetzen sollen.

Jedes Textfragment kann in verschiedenen Sprachen definiert werden. Welcher Sprachcode für welche Sprache steht, können Sie der Beschreibung des Profilparameters ULANG entnehmen.

Die Textfragmente werden in EBCDIC- und Unicode-Schreibweise erzeugt.

**Anmerkung:** Um das Modul NATTXT3 zu assemblieren, muss ein High-Level-Assembler verwendet werden, der die Makrofunktion UPPER und die Definition von Unicode-Zeichen (DC CU'*unicode text*') unterstützt.

Beispiel:

Der Text für den Natural-Fehler NAT0082 (beim Versuch, ein nicht vorhandenes Programm auszuführen) sieht wie folgt aus:

```
Invalid command, or :1: :2: does not exist in library.
```

Der Versuch, das Objekt NOTEXIST auszuführen, führt zu folgendem Ergebnis:

```
NAT0082 Invalid command, or Program NOTEXIST does not exist in library.
```

`:2:` wurde durch den Objektnamen (`NOTEXIST`) ersetzt.

`:1:` wurde durch das Textfragment `Program` ersetzt.

Das Textfragment wurde im Modul `NATTXT3` wie folgt deklariert:

```
*================================================================
*              PROGRAM                  0002
*================================================================
      MSGSDEF  &LC_PGM
         SPACE
*----------------------------------------------------------------
      MSGSLAN  01,Program           1   ENGLISH
      MSGSLAN  02,Programm          2   GERMAN
      MSGSLAN  03,programme         3   FRENCH
      MSGSLAN  04,programa          4   SPANISH
         SPACE
*----------------------------------------------------------------
      MSGSGEN
```

Textfragmentwerte für zusätzliche Sprachen können durch Hinzufügen weiterer `MSGSLAN`-Makros eingegeben werden.

## NTERMSG - Natural-Beendigungsmeldungen und Rückgabecodes

Natural verfügt über eine Reihe von Standard-Session-Termination-Meldungen (NAT99...), die im Makro `NTERMSG` ausgeliefert werden und dort geändert werden können (z.B. um sie in eine andere Sprache zu übersetzen). Die Gesamtlänge von ID und Text kann bis zu 72 Zeichen betragen. Nachdem das Makro `NTERMSG` geändert wurde, muss das Natural-Parametermodul neu assembliert und verlinkt werden.

Neben der Meldungskennung und dem Text enthält jede Standardbeendigungsmeldung auch einen der folgenden Natural-System-Rückgabecodes, die ebenfalls im Makro `NTERMSG` definiert sind:

| Code | Erläuterung |
|---|---|
| 0 | Normale Beendigung. |
| 4 | Fehler bei der Ausführung/Kompilierung (nur Batch-Modus). |
| 8 | Beendigung aufgrund eines schweren Laufzeitfehlers. |
| 12 | Sitzungsinitialisierungsfehler. |
| 16 | Abnormale Beendigung aufgrund eines Abbruchs oder eines schweren Umgebungsfehlers. |

Wenn der Profilparameter `TS` auf `ON` gesetzt ist, werden die Abbruchmeldungen mit Hilfe der Großbuchstabentabelle `NTUTAB1`, die im Modul `NATCONFG` bereitgestellt wird, in Großbuchstaben umgesetzt, bevor sie angezeigt werden.

Zusätzlich zu `TS=ON` werden weitere Parameter für die Umsetzung von Meldungen in Großbuchstaben von mehreren Natural-Komponenten bereitgestellt. Weitere Informationen finden Sie unter *Weitere Parameter für die Umsetzung in Großbuchstaben* in der Beschreibung des Profilparameters `TS`.

# 7 Natural-Konfigurationstabellen


- NATCONFG - Natural-Konfigurationstabellen ..... 40
- Allgemeine Übersicht über die in NATCONFG verwendeten Makros ..... 40
- NTDVCE - Terminal-Gerätespezifikationstabelle ..... 41
- NTMSG - Definitionen der Meldungsprotokolltabelle ..... 42
- NTSTAT - Definition der mit dem Natural-Nukleus verlinkten Objekte ..... 43
- NTCPAGE - Definitionen der Codepages ..... 43
- Unterstützung von Codepages ..... 45
- Unterstützte Ausgabegeräte ..... 45
- Umsetzungstabellen ..... 46
- Umsetzung von Klein- in Großbuchstaben ..... 50
- CMULT-Eintrag ..... 51
- Umwandlung der Ausgabe ..... 51
- Umwandlung von Eingaben ..... 52
- Code-Umsetzung von DBCS-Daten ..... 52
- NTTZ - Zeitzonendefinitionen ..... 53

Dieses Kapitel enthält allgemeine Informationen über die Natural-Konfigurationstabellen, die im Modul NATCONFG enthalten sind.

Siehe auch ***Unterstützte Eingabe- und Ausgabegeräte***.

## NATCONFG - Natural-Konfigurationstabellen

Das Modul NATCONFG enthält die Natural-Konfigurationstabellen.

**Vorsicht:** Die Standardvorgaben in NATCONFG müssen und sollten generell nicht verändert werden. Insbesondere sollten Sie *keine* der in der folgenden Liste mit einem Stern (*) gekennzeichneten Tabellen ohne vorherige Rücksprache mit dem Support *ändern*.

Für die meisten der Tabellen gibt es entsprechende Makros im **Natural-Parametermodul** sowie dynamische Profilparameter. Wenn Sie eine NATCONFG-Tabelle ändern müssen, verwenden Sie das entsprechende Makro im Parametermodul oder den dynamischen Profilparameter, um die Tabelle zu überschreiben. (Würden Sie die Änderungen in den NATCONFG-Tabellen selbst vornehmen, müssten Sie NATCONFG mit späteren Natural-Releases erneut ändern und neu assemblieren).

Das Modul NATCONFG enthält Makros für die Definition der folgenden Natural-Standardkonfigurationstabellen.

Darüber hinaus enthält es die folgenden Tabellen:

- Die Tabelle der Standard-Attention-Identifier. Sie definiert die physischen Terminalabruftasten für Natural (*).
- Verschiedene andere Tabellen (*).

## Allgemeine Übersicht über die in NATCONFG verwendeten Makros

Die folgende Tabelle gibt einen allgemeinen Überblick über die Makros, die im Modul NATCONFG für die Definition der Natural-Standardkonfigurationstabellen verwendet werden:

| Makro | Zweck |
|---|---|
| NTDVCE * | Tabelle der Terminaltypen. Dient dazu, den zu verwendenden Terminaltreiber anzugeben, Einzelheiten siehe Beschreibung unten.<br>**Wichtig:** Ändern Sie nicht ein bestehendes NTDVCE-Makro, sondern erstellen Sie ein neues. |
| NTMSG | Meldungsprotokolltabelle. Natural-Meldungen, die in das Job-Meldungsprotokoll geschrieben oder auf der Bedienerkonsole angezeigt werden sollen. |

| Makro | Zweck |
|---|---|
| NTSTAT | Definition von Natural-Objekten, die mit dem Natural-Nukleus verlinkt sind. |
| NTCPAGE | Definitionen von Codepages. |
| NTTAB | Primäre Ausgabe-Umsetzungstabelle. |
| NTTAB1 NTTAB2 | Sekundäre Ausgabe/Eingabe-Umsetzungstabellen. |
| NTUTAB1 NTUTAB2 | Tabellen für die Umsetzung von Klein- und Großbuchstaben. Diese Tabellen müssen z.B. für den deutschen Zeichensatz angepasst werden. |
| NTTABA1 NTTABA2 | Tabellen für die Umsetzung von EBCDIC-Zeichen in ASCII-Zeichen und umgekehrt. Diese Tabellen werden vom Natural-Object Handler verwendet. |
| NTTABL | SYS*-Umsetzungstabelle. Setzt die Ausgabe von Programmen um, die in Natural SYS... Libraries enthalten sind. |
| NTLANG * | Sprachumsetzungstabelle. Enthält eine Liste aller verfügbaren Sprachcodes, die für Natural definiert sind. |
| NTSCTAB | Scanner-Zeichentypentabelle. Legt fest, welche Zeichen Kleinbuchstaben, Großbuchstaben, numerische Zeichen und Sonderzeichen sind (gilt für dynamische Profilparameter, MASK- und SCAN-Optionen). |
| NTTZ | Zeitzonendefinitionen. Das Makro NTTZ ermöglicht die Angabe von Zeitzonen und die automatische Umstellung auf die Sommerzeit und umgekehrt. |
| NTBUFID | Mit den Parametern MIN und MAX dieses Makros können die Puffergrößengrenzen für variable Puffer geändert werden, siehe ***Anpassung der Puffereigenschaften***.<br>**Wichtig:** Die Standardwerte der anderen Parameter in diesem Makro sollten nicht geändert werden, da die Ergebnisse unvorhersehbar sein können.<br>**Wichtig:** |

* Ändern Sie ohne vorherige Rücksprache mit dem Support *keine* der in dieser Liste mit einem Stern (*) gekennzeichneten Tabellen.

Weitere Einzelheiten finden Sie unter ***Umsetzungstabellen***.

## NTDVCE - Terminal-Gerätespezifikationstabelle

Für jeden von Natural unterstützten Terminaltyp wird eine Terminal-Konvertierungsroutine bereitgestellt. Die entsprechenden Terminaltreiber sind für die eigentlichen Terminal-Ein-/Ausgaben zuständig. Sie bauen den physischen Datenstrom aus dem Bildschirmpuffer und dem Bildschirmattributpuffer auf und stellen ihn in den Terminal-E/A-Puffer.

Zusätzlich wird ein Telex-Treiber für Con-nect bereitgestellt, um eine schnellere Telex-, Telefax- und Teletext-Kommunikation vom und zum Topcall Messaging Server zu ermöglichen. Dieser Treiber unterstützt das Topcall-Ganzseitenprotokoll.

Mit dem NTDVCE-Makro ist es möglich, neue Terminaltreiber zu Natural hinzuzufügen, um Änderungen der terminalspezifischen Ein-/Ausgabe- oder Groß-/Kleinschreibungstabellen anzugeben. Weitere Informationen, die angegeben werden können, sind das Rahmenzeichen, die Position der Nachrichtenzeile, ob die Bildschirmoptimierung ein- oder ausgeschaltet werden soll, sowie verschiedene Flags in der IOCB. Darüber hinaus kann die Terminalspezifikation an einen vorhandenen Treiber weitergeleitet werden, indem andere Umsetzungstabellen verwendet werden, oder sie kann in eine Treiberroutine eingehängt werden.

Das Makro NTDVCE wird entweder durch das Terminalkommando %T= in der Natural-Kommandozeile oder durch das Statement SET CONTROL 'T=...' in einem Natural-Programm aufgerufen. Zu Beginn einer Natural-Sitzung werden die Umsetzungstabellen NTTAB, NTTAB1, NTTAB2, NTUTAB1 und NTUTAB2 aus dem Modul NATCONFG in den Benutzerbereich kopiert, wo sie von NTDVCE geändert werden.

Beachten Sie, dass die Umsetzungstabellen von denselben Makros dynamisch oder innerhalb des Natural-Parametermoduls geändert werden können.

## NTMSG - Definitionen der Meldungsprotokolltabelle

Das Makro NTMSG wird verwendet, um Natural-Meldungen zu definieren, die auf der Bedienerkonsole ausgegeben oder in das Job-Meldungsprotokoll (falls vorhanden) geschrieben werden sollen. Eine definierte Meldung wird zusätzlich geschrieben, d.h. die übliche Natural-Verarbeitung bleibt unverändert. Die Tabelle zur Definition von Protokollmeldungen finden Sie unter dem Label NATMSGT in NATCONFG. Dort können Sie Ihre NTMSG-Definitionen auf der Basis einer Meldung pro Zeile hinzufügen.

### NTMSG-Makro-Syntax

Die Syntax des Makros NTMSG lautet wie folgt:

```
        NTMSG  NATnnnn,logid
```

### NTMSG-Makro-Parameter

| Parameter | Beschreibung |
|---|---|
| NAT*nnnn* | *nnnn* ist die Nummer der Natural-Meldung (erforderlich). |
| *logid* | Gibt das Protokollziel an, d.h. die Bedienerkonsole oder das Job-Meldungsprotokoll oder beides.<br><br>Mögliche Werte: WTO, WTL oder WTO+WTL |

## NTSTAT - Definition der mit dem Natural-Nukleus verlinkten Objekte

Jedes Objekt, das mit dem Natural-Nukleus verlinkt werden soll, muss mit einem NTSTAT-Makro angegeben werden. Bei der Suche nach einem Objekt durchsucht Natural immer zuerst diese Liste, unabhängig von der angegebenen Library. Wie Sie Natural-Objekte mit dem Natural-Nukleus verlinken, erfahren Sie im Abschnitt ***Natural-Objekte mit dem Natural-Nukleus verlinken*** beim Dienstprogramm ULDOBJ.

### NTSTAT-Makro-Syntax

Die Syntax des NTSTAT-Makros lautet wie folgt:

```
        NTSTAT object-name[,TYPE=W]
```

### NTSTAT-Makro-Parameter

| Parameter | Beschreibung |
|---|---|
| *object-name* | Gibt den Namen des Objekts an, das mit dem Natural-Nukleus verlinkt ist. |
| TYPE=W | Bedeutet, dass der Einstiegspunkt des verlinkten Objekts als "weak external" für den Natural-Nukleus definiert ist. Dadurch wird eine Fehlermeldung des Linkage Editors vermieden, falls das Objekt nicht mit dem Natural-Nukleus verlinkt ist. |

## NTCPAGE - Definitionen der Codepages

Alle Codepages, die während einer Natural-Sitzung verwendet werden sollen, müssen im Quellcode-Modul NATCONFG vordefiniert werden. Für jede zu definierende Codepage muss ein spezielles NTCPAGE-Makro eingegeben werden. Während der Sitzungsinitialisierung wird die durch die Profilparameter CP, CPOBJIN, CPSYNIN, CPPRINT und die CP und den CP-Schlüsselwort-Subparameter des Profilparameters PRINT oder des Parametermakros NTPRINT überprüft. Wenn diese Codepage nicht in NATCONFG definiert ist, wird eine Fehlermeldung ausgegeben.

### NTCPAGE-Makro-Syntax

Die Syntax des NTCPAGE-Makros lautet wie folgt:

```
        NTCPAGE IANA=value,                                          *
              CCSID=value,                                           *
              CCSN=value,                                            *
              ALIAS=value,                                           *
              PHC=value,                                             *
              MULTI=value,                                           *
              ECS=value
```

## NTCPAGE-Makro-Parameter

| Parameter | Beschreibung | |
|---|---|---|
| IANA | Dieser Parameter gibt den Standardnamen der Codepage an. Maximale Länge: 64 Zeichen. | |
| CCSID | Dieser Parameter ist erforderlich. Er gibt die kodierte Zeichensatzkennung an, d. h. einen numerischen Wert mit bis zu 5 Ziffern.<br><br>**Beispiele:**<br>1141 Deutsche EBCDIC-Codepage<br>62243 Hebräisch/Lateinisch (ISO 8859) Codepage | |
| ALIAS | Dieser Parameter ist optional. Er gibt den Alias-Namen der Codepage an. Maximale Länge: 32 Zeichen. | |
| PHC | Dieser Parameter ist optional. Er gibt das Platzhalterzeichen an. Länge: 2 Bytes hexadezimal. | |
| ECS | Dieser Parameter ist optional. Er gibt die Schlüsselnummer der Codepage im Entire Conversion Service (ADAECS) an, der von Adabas verwendet wird. | |
| MULTI | Dieser Parameter ist optional. Er gibt an, ob es sich bei der Codepage um eine Ein-Byte-Codepage oder eine Multi-Byte- oder ASCII-Codepage handelt. Mögliche Werte: | |
| | ON | Die Codepage ist eine Ein-Byte-Codepage, z. B. IBM01140. Die Codepage kann als Natural Sitzungs-Codepage verwendet werden. Die Sitzungs-Codepage wird durch den Natural-Profilparameter CP definiert. |
| | OFF | Die Codepage ist eine Multi-Byte-Codepage oder ASCII-Codepage. Sie kann nicht als Session-Codepage für Natural verwendet werden. Jeder Versuch, diese Codepage zu verwenden, führt zur Initialisierungsmeldung NAT7019.<br><br>Dies ist die Standardeinstellung. |
| | VALID | Die Codepage ist eine Multi-Byte-Codepage, kann aber als Natural-Session-Codepage verwendet werden. IBM-939 ist zum Beispiel eine japanische EBCDIC-Codepage, die DBCS-Zeichen enthält. |

**Beispiele:**

```
        NTCPAGE IANA=IBM819,                                          *
              CCSID=819,                                              *
              ALIAS='ISO-8859-1',                                     *
              PHC=003F
```

```
        NTCPAGE IANA='IBM-939',                                       *
              CCSID=939,                                              *
              ECS=3035,                                               *
              ALIAS='ibm-939_P120-1999',                              *
              PHC=3013,                                               *
              MULTI=VALID
```

Siehe auch *Unicode-/Codepage-Umgebung konfigurieren und verwalten*.

## Unterstützung von Codepages

Mit dem Makro `NTDVCE` können verschiedene Codepages definiert und mit einem bestimmten Terminaltyp und -namen verlinkt werden. Wenn Natural dann mit `PM=C` gestartet wird, werden alle Terminal-E/A bei der Eingabe übersetzt und bei der Ausgabe neu übersetzt. Solange die Codepages kompatibel sind, kann so eine gemeinsame Datendarstellung beibehalten werden.

Siehe auch *SYSCP Utility* in der *Debugger und Dienstprogramme (Utilities)*-Dokumentation.

## Unterstützte Ausgabegeräte

Attributkontrollvariablen und -formate definieren Attribute, um eine bestimmte Darstellung auf dem Ausgabegerät zu erzeugen. Natural bietet eine breite Palette möglicher Attribute, um dem Endanwender die bestmögliche Nutzung bei der Gestaltung von Masken (Maps) und Reports auf dem Terminal zu ermöglichen.

Leider unterstützen nicht alle Terminals alle bei Natural verfügbaren Funktionen. Diese Eigenschaften werden auf solchen Geräten meist ignoriert oder durch andere Techniken simuliert. Grundsätzlich gibt es in einer IBM-Umgebung zwei Datenstromdefinitionen, den Standard-Datenstrom und den erweiterten Datenstrom.

Die folgenden Ausgabegeräte werden unterstützt:

- Sequenzielle Ausgabegeräte für Batch, zusätzliche Reports
- Zeilenorientierte Online-Terminals

- Blockmodus-orientierte Online-Terminals

### Sequenzielle Ausgabegeräte für Batch, zusätzliche Reports

Die Ausgabedaten enthalten Standard-ASA-Steuerzeichen, die den Zeilenvorschub und die Seitenauswurf-Funktion des jeweiligen Druckers steuern. Dieser Drucker kann entweder der zentrale Drucker im Rechenzentrum sein, der durch das Online- oder Batch-Spooling-System unterstützt wird, oder der SCS-Drucker, der als Online-Terminaldrucker verwendet wird.

Die folgenden Geräte können zum Drucken der in dieser Form erstellten Reports verwendet werden:

| Gerät | Typ |
|---|---|
| Anschlagdrucker | Zentrale Standard-Druckerhardware |
| Laserdrucker | Hochgeschwindigkeitsdrucker, Terminaldrucker |
| Typenraddrucker | Terminaldrucker |
| Tintenstrahl | Terminaldrucker |

### Zeilenorientierte Online-Terminals

| Terminal-Marke | Beschreibung |
|---|---|
| TTY | Daten, die an TTY-Geräte gesendet werden, werden mit den Standardzeichen Formfeed, Linefeed usw. generiert. |

### Blockmodus-orientierte Online-Terminals

| Terminal-Marke | Beschreibung |
|---|---|
| IBM | Alle Modelle und Größen, die den Standard-Datenstrom und/oder den erweiterten Datenstrom unterstützen. |
| PC | Alle Modelle und Größen, die den Standard-Datenstrom und/oder den erweiterten Datenstrom unterstützen. |

## Umsetzungstabellen

Alle Daten, die von Natural-Programmen gedruckt, angezeigt oder geschrieben werden, werden von Natural umgesetzt. Dadurch wird sichergestellt, dass keine unzulässigen Steuerzeichen zu Terminal-E/A-Fehlern führen oder Datenmüll auf dem Terminal erscheinen kann.

Ein weiteres Merkmal ist die Umsetzung in und aus Zeichensätzen, die von der lateinischen Definition abweichen, insbesondere arabische, kyrillische, griechische und hebräische Zeichen.

In diesem Abschnitt werden alle Merkmale und Funktionen beschrieben, die die Umsetzung von Feldern betreffen, wenn Daten auf externe Geräte wie CRT (Bildschirmterminals) oder Online- und Batch-Spooling-Systeme geschrieben werden.

Die Statements `INPUT`, `DISPLAY`, `PRINT` und `WRITE` schreiben Daten auf externe Geräte wie CRT, TTY oder in sequenzielle Dateien oder lesen Daten von ihnen. Alle diese Statements verwenden Parameter wie Konstanten, Variablen, Editiermasken, Attributkontrollvariablen und Formate zur Steuerung des Ausgabebildes und der Eingabedarstellung. Konstanten und Variablen werden durch Verwendung ihrer jeweiligen Werte im Ausgabebild erzeugt. Die Darstellung dieser Werte wird dann durch die Attributkontrollvariablen, Formate, Editiermasken und Umsetzungstabellen gesteuert.

Natural verwendet mehrere Umsetzungstabellen und bietet auch die Verwendung alternativer Umsetzungstabellen an, die alle in NATCONFG enthalten sind.

Die folgenden Tabellen werden bereitgestellt:

| Makro | Tabelle |
|---|---|
| `NATSCTU` | Scanner-Tabelle, die für Unicode-Zeichen erforderlich ist. Sie bildet die Eigenschaften von Unicode-Zeichen der Unicode-Spezifikation ab (wie von der gelieferten ICU-Version unterstützt), die vom Natural-Nukleus verwendet werden sollen.<br><br>**Wichtig:** Diese Tabelle darf niemals geändert werden. |
| `NATCPTAB` | Optionale Tabellen zur Beschleunigung der Konvertierung von *Einzelbyte*-Codepages.<br><br>Wenn die Tabelle vorhanden ist, erfolgt die Konvertierung von einer Codepage in eine andere Codepage schneller, da sie über diese Tabelle und nicht durch den Aufruf von ICU-Funktionen erfolgt.<br><br>Die folgenden Codepages werden von der mitgelieferten `NATCPTAB` unterstützt:<br><br>`IBM01140`<br>`IBM01141`<br>`IBM01145`<br>`IBM01146`<br>`IBM01147`<br>`ASCII`<br><br>Es ist möglich, mit dem Makro `NTCPCNV` neue Einträge hinzuzufügen. Für jede Konvertierungsrichtung ist ein Eintrag erforderlich, der den IANA-Namen der Quell-Codepage, den IANA-Namen der Ziel-Codepage und optional ein Leerzeichen, ein Ersetzungszeichen und ein Platzhalterzeichen enthält, gefolgt von einer vollständigen Liste der Zeichenzuordnungen. |
| `NTSCTAB` | Tabelle, die die Eigenschaften von Zeichen definiert,<br><br>■ die in Maskendefinitionen für die Option `MASK` verwendet werden,<br>■ die als Begrenzungszeichen in den Statements `EXAMINE` und `SEPARATE` erkannt werden. |

| Makro | Tabelle |
|---|---|
| | Diese Tabelle kann zur Definition von Großbuchstaben-Attributen, Kleinbuchstaben-Attributen, Sonderzeichen, hexadezimalen Zeichen und numerischen Zeichen verwendet werden.<br><br>Um diese Tabelle zu ändern, können Sie das Makro `NTSCTAB` im Natural-Parametermodul oder den entsprechenden dynamischen Profilparameter `SCTAB` verwenden.<br><br>Wenn der Profilparameter `CP` auf einen anderen Wert als `OFF` gesetzt ist, wird die Änderung ignoriert und die Tabelle wird entsprechend der für die Natural-Sitzung verwendeten Codepage angepasst. Siehe auch *Umsetzungstabellen* in der *Unicode- und Codepage-Unterstützung*-Dokumentation. |
| `NTTAB` | Die standardmäßige (primäre) Ausgabe-Umsetzungstabelle, die für die Bildschirm- oder Druckerausgabe verwendet wird.<br><br>Grundsätzlich werden mit dieser Tabelle alle Zeichen unterhalb von `X'40'`, d.h. vom Leerzeichen bis zum Fragezeichen, umgesetzt (`X'00'` wird nicht umgesetzt). Damit ist gewährleistet, dass alle Terminal-Steuerzeichen vor der Ausgabe umgesetzt werden und keine Control Escape-Sequenzen die Bildschirmausgabe beeinflussen können. Sonderzeichen (`X'FE'` und `X'FF'`), die die Bildschirmausgabe beeinflussen könnten, werden in das Fragezeichen (`?`) umgesetzt.<br><br>Wenn nichts anderes angegeben ist, werden alle Natural-Ausgabedaten mit `NTTAB` umgesetzt.<br><br>Um diese Tabelle zu ändern, können Sie das Makro `NTTAB` im Natural-Parametermodul oder den entsprechenden dynamischen Profilparameter `TAB` verwenden.<br><br>Die Änderung wird ignoriert, wenn mit dem Profilparameter `CP` (`CP=ON`, `CP=AUTO` oder `CP=`*`codepage`*) eine Codepage angegeben ist, und die Tabelle wird von ICU entsprechend der beim Sitzungsstart verwendeten Codepage angepasst.<br><br>Außerdem werden alle Zeichen unterhalb von `X'40'` wie oben beschrieben in das Fragezeichen (`?`) umgesetzt. Ein Zeichen wird von dieser Umsetzung ausgenommen, wenn eine der folgenden Bedingungen zutrifft:<br><br>■ Das Zeichen wird explizit in dasselbe Zeichen umgesetzt.<br>■ Das Zeichen ist eines der logischen Shift-Out/Shift-In-Zeichen, die mit dem Profilparameter `SOSI` angegeben wurden (siehe *Parameter-Referenz*-Dokumentation), und die angegebene Codepage ist keine MBCS-Codepage. |
| `NTTAB1` | Die alternative (sekundäre) Ausgabe-Umsetzungstabelle für den sekundären Zeichensatz, die verwendet wird, wenn der Natural-Parameter `PM` auf `C` gesetzt ist.<br><br>Wichtig ist dabei die Umsetzung aller möglichen Terminal-Steuerzeichen. Wenn `PM=C` angegeben ist, werden alle Natural-Ausgabedaten mit `NTTAB1` übersetzt. Eine mögliche Anwendung von `NTTAB1` besteht darin, die Umsetzung von Escape-Sequenzen zur Druckersteuerung zu vermeiden.<br><br>Um diese Tabelle zu ändern, können Sie das Makro `NTTAB1` im Natural-Parametermodul oder den entsprechenden dynamischen Profilparameter `TAB1` verwenden. |

| Makro | Tabelle |
|---|---|
| | Die Änderung wird ignoriert, wenn eine Codepage über den Profilparameter CP (CP=ON, CP=AUTO or CP=*codepage*) angegeben ist und die Tabelle nicht verwendet wird. |
| NTTAB2 | Die sekundäre Eingabeumrechnungstabelle, die verwendet wird, wenn der Natural-Parameter PM auf C eingestellt ist. Wenn PM=C angegeben ist, werden alle Natural-Eingabedaten mit NTTAB2 umgesetzt. Die Umsetzung zwischen verschiedenen Sprachen oder Codepages kann mit dieser Tabelle zusammen mit NTTAB1 durchgeführt werden.<br><br>Um diese Tabelle zu ändern, können Sie das Makro NTTAB2 im Natural-Parametermodul oder den entsprechenden dynamischen Profilparameter TAB2 verwenden.<br><br>Die Änderung wird ignoriert, wenn eine Codepage über den Profilparameter CP (CP=ON, CP=AUTO oder CP=*codepage*) angegeben ist und die Tabelle nicht verwendet wird. |
| NTTABS | In dieser Tabelle sind alle gültigen Zeichen definiert, die in Natural-Variablennamen verwendet werden können. Sie wird für den Natural-Syntaxprozessor verwendet.<br><br>Außerdem sind darin alle gültigen Zeichen definiert, die an der ersten Position eines Natural-Variablennamens verwendet werden können.<br><br>Darüber hinaus wird festgelegt, ob es sich bei der Variablen um eine globale Variable, eine Nicht-Datenbank-Variable oder eine Quellcode-Variable handelt.<br><br>Wenn mit dem Profilparameter CP (CP=ON, CP=AUTO oder CP=*codepage*) eine Codepage angegeben wird, wird die Tabelle von ICU entsprechend der beim Sitzungsstart verwendeten Codepage angepasst. |
| NTUTAB1 | Die benutzerspezifische Beispieltabelle für die Umsetzung der Eingabe von Klein- in Großbuchstaben.<br><br>Darüber hinaus führt diese Tabelle die mit dem Statement EXAMINE TRANSLATE INTO UPPER CASE angegebene Umsetzung durch.<br><br>Um diese Tabelle zu ändern, können Sie das Makro NTUTAB1 im Natural-Parametermodul oder den entsprechenden dynamischen Profilparameter UTAB1 verwenden.<br><br>Die Änderung wird ignoriert, wenn eine Codepage mit dem Profilparameter CP (CP=ON, CP=AUTO oder CP=*codepage*) angegeben ist und die Tabelle nicht verwendet wird. |
| NTUTAB2 | Beispiel für eine benutzerspezifische Umsetzungstabelle, die die mit dem Statement EXAMINE TRANSLATE INTO LOWER CASE angegebene Umsetzung durchführt.<br><br>Um diese Tabelle zu ändern, können Sie das Makro NTUTAB2 im Natural-Parametermodul oder den entsprechenden Profilparameter UTAB2 verwenden.<br><br>Die Änderung wird ignoriert, wenn eine Codepage mit dem Profilparameter CP (CP=ON, CP=AUTO or CP=*codepage*) angegeben ist und die Tabelle nicht verwendet wird. |
| NTLANG | Sprachencode-Tabelle, die festlegt, welche Sprachnummer welchem Sprachencode in der Systemvariablen *LANGUAGE zugeordnet ist. |
| NTTABL | Die SYS*-Ausgabe-Umsetzungstabelle, die durch den Natural-Profilparameter TS gesteuert wird. Bei TS=ON wird diese Tabelle verwendet, um die von Programmen in Natural |

| Makro | Tabelle |
|---|---|
| | SYS*-Libraries erzeugten Ausgaben (außer änderbaren Feldern) von lateinischer Kleinschreibung in Großschreibung umzusetzen.<br><br>Diese Tabelle ermöglicht die Verwendung aller Groß- und Kleinbuchstaben in lateinisch geprägten Ländern, erlaubt aber auch die Verwendung dieser Anwendungen in Ländern, in denen die Kleinbuchstaben durch ein eigenes Alphabet ersetzt wurden.<br><br>Um diese Tabelle zu ändern, können Sie das Makro NTTABL im Natural-Parametermodul oder den entsprechenden dynamischen Profilparameter TABL verwenden.<br><br>Wenn Natural mit einer MBCS-Codepage (z. B. CP='IBM-939') ausgeführt wird, wird die Tabelle nicht verwendet, sondern die Umsetzung erfolgt über ICU gemäß den aktuellen Gebietsschema-Einstellungen. |
| WRDFCUC1<br>WRDFCUC2<br>WRDFCSP2 | Die DBCS-Umsetzungstabellen, die für die Umsetzung von Doppelbyte-Zeichen in lateinische Zeichen und umgekehrt verwendet werden.<br><br>**Wichtig:** Diese Tabellen müssen explizit aktiviert werden, z.B. für fernöstliche Länder. |

## Umsetzung von Klein- in Großbuchstaben

Für änderbare Felder und Eingabefelder kann die Umsetzung in Groß- und Kleinbuchstaben festgelegt werden. Generell bedeutet die Umsetzung in Kleinbuchstaben, dass die Daten so übernommen werden, wie sie eintreffen. Es wird keine Umsetzung vorgenommen. Damit ist es z.B. auch im Batchbetrieb möglich, hexadezimale Daten ohne Umsetzung einzulesen.

Es gibt mehrere Möglichkeiten, die Groß-/Kleinschreibung anzugeben:

| | |
|---|---|
| LC=OFF | Die Umsetzung von Kleinbuchstaben (Lower to Upper Case Translation) ist ausgeschaltet, d.h. es erfolgt eine globale Großbuchstabenumsetzung.<br><br>Dieser Profilparameter kann im Natural-Parametermodul oder als dynamischer Parameter angegeben werden.<br><br>Achtung: Der *Session*-Parameter LC hat eine völlig andere Funktion (Leading Characters). |
| %U | Die Großbuchstabenumsetzung ist global aktiv.<br><br>Auf Feldebene kann das Attribut AD=T oder AD=W angegeben werden. Diese Attribute werden nur wirksam, wenn die globale Großbuchstabenumsetzung ausgeschaltet ist (LC=ON, %L). Dann ist es möglich, die Umsetzung auf Feldebene aus einem Natural-Programm heraus zu steuern. |

| | |
|---|---|
| EXAMINE TRANSLATE | Die Umsetzung von Groß-/Kleinschreibung kann auch mit dem Statement EXAMINE TRANSLATE durchgeführt werden.<br><br>EXAMINE TRANSLATE führt standardmäßig eine Großbuchstabenumsetzung unter Verwendung der Umsetzungstabelle NTUTAB1 und eine Kleinbuchstabenumsetzung mittels der Umsetzungstabelle NTUTAB2 durch. |

## CMULT-Eintrag

Die Verwendung des CMULT-Eintrags wird nicht mehr empfohlen. Verwenden Sie stattdessen das EXAMINE TRANSLATE-Statement (siehe oben).

## Umwandlung der Ausgabe

Alle Felder werden nach der Formatierung durch eventuelle Editiermasken, AL- oder NL-Parameterwerte, Füllzeichen usw. anhand einer Umsetzungstabelle umgewandelt. Dadurch wird sichergestellt, dass keine Daten mit eingebetteten Steuerinformationen, die nicht explizit von Natural erzeugt wurden, an das Front-End-Druckgerät gesendet werden können. Das bedeutet, dass Felder an ein Anzeigegerät gesendet werden können, auch wenn sie hexadezimale Informationen enthalten, die mit internen Attributen identisch sind. Diese Attribute werden vor einer Ausgabeoperation umgewandelt, so dass Natural das in der Ausgabeanweisung definierte Bildschirmlayout garantiert.

Es sind mehrere Umsetzungstabellen verfügbar. Wenn nichts explizit definiert ist, wird die primäre Umsetzungstabelle NTTAB verwendet.

Wenn PM angegeben ist, wird die sekundäre Umsetzungstabelle NTTAB1 verwendet. Für änderbare Felder bedeutet PM=C auch, dass die eingehenden Daten erneut umgesetzt werden, d.h. für die Ausgabe umgesetzt und für die Eingabe neu umgesetzt werden.

Mit dieser Umsetzungstabellenlogik ist es z.B. möglich, arabische Ziffern in lateinische Ziffern umzuwandeln. Arabische Ziffern haben eine andere hexadezimale Darstellung als die normalen lateinischen Ziffern auf der Terminal-Hardware. So können bei der Ausgabe die lateinischen Ziffern in das arabische Äquivalent umgewandelt werden und bei der Eingabe können die arabischen Ziffern wieder in lateinische zurückgewandelt werden.

Besondere Überlegungen sind bei Natural-System-Anwendungen erforderlich, die lateinische Klein- und Großbuchstaben verwenden. Vor allem auf Terminals, die Arabisch, Griechisch, Kyrillisch usw. unterstützen, kann die Hardware so eingestellt werden, dass keine lateinischen Kleinbuchstaben, sondern die einheimischen Zeichen angezeigt werden.

Leider sind lateinische Kleinbuchstaben krakelig, wenn sie z. B. in kyrillischer Schrift angezeigt werden. Daher kann Natural mit dem Parameter TS=ON (Systemausgabe umsetzen) verwendet werden. TS=ON bewirkt, dass SYS*-Libraries (ohne die Library SYSTEM) und alle Natural-System-

kommandos mit Hilfe einer dritten Umsetzungstabelle namens NTTABL umgesetzt werden. Standardmäßig führt diese Umsetzungstabelle eine Umsetzung in Großbuchstaben für alle lateinischen Kleinbuchstaben durch. Allerdings werden nur Ausgabedaten auf diese Weise behandelt. Dies ermöglicht die Eingabe von Daten im nativen Zeichensatz auch in Natural-Editoren oder Systemanwendungen.

Werden jedoch Natural-Dienstprogramme (Utilities) verwendet, um Daten anzuzeigen, die im nativen Zeichensatz eingegeben wurden, führt dies zu einer Großbuchstabenumsetzung auch für Daten in z. B. kyrillischer Darstellung. Das Ergebnis wäre wiederum unleserlich. Daher können alle Dienstprogramme (Utilities) des Natural-Systems das Format PM=C für Felder verwenden, die Daten enthalten, die im nativen Zeichensatz eingegeben wurden. In diesem Fall wird weder die Umsetzungstabelle NTTABL noch die sekundäre Umsetzungstabelle NTTAB1 verwendet. Die Daten werden einfach mit der primären Umsetzungstabelle NTTAB umgesetzt.

Weitere Informationen finden Sie unter den Profilparametern PM und TS in der *Parameter-Referenz*-Dokumentation.

## Umwandlung von Eingaben

Die Umsetzungsstabelle NTUTAB1 ist vorhanden, um die Umsetzung von Klein- in Großbuchstaben zu steuern. Dies kann in Ländern, in denen Sonderzeichen verwendet werden, die nicht nach der einfachen Logik aufgebaut sind, dass nur ein Bit den Status dieses Buchstabens steuert, zu Problemen führen. Dies betrifft insbesondere deutsche Umlaute oder dänische Sonderzeichen. In solchen Fällen kann die Umsetzung nur durch eine Anpassung der NTUTAB1-Tabelle erreicht werden, in der für jedes Zeichen der entsprechende Klein-/Großbuchstabe angegeben werden kann.

Wenn Großbuchstabenumsetzung (%U) und PM=C angegeben ist, erfolgt zuerst die Großbuchstabenumsetzung (mit NTUTAB1) und dann die sekundäre Eingabeumsetzung (mit NTTAB2).

## Code-Umsetzung von DBCS-Daten

Damit Doppelbyte-Zeichensätze (DBCS) verarbeitet werden können, gibt es die Anwendungsprogrammierschnittstelle **USR4213N**, die Doppelbyte-Zeichen in lateinische Zeichen umsetzt, siehe ***Doppelbyte-Zeichensätze (DBCS)***.

# NTTZ - Zeitzonendefinitionen

Das Makro `NTTZ` wird verwendet, um eine Zeitzone und eine automatische Umstellung auf und von der Sommerzeit festzulegen.

**Anmerkung:** Zeitdefinitionen werden vom Systemadministrator festgelegt, und der Benutzer kann diese Definitionen mit dem Natural-Profilparameter `TD=zonename` referenzieren. Mit diesem Parameter können Benutzer aus verschiedenen Ländern und Zeitzonen ihre eigene Ortszeit wählen.

Das Makro `NTTZ` kann auf minimaler Basis verwendet werden, um eine Zeitdifferenz für eine Zeitzone zu definieren. Außerdem kann ein automatischer Wechsel zur und von der Sommerzeit angegeben werden, entweder als festes Datum oder in einer flexibleren Definition wie „erster Sonntag im April". Die automatische Umstellung auf die Sommerzeit erfolgt während einer laufenden Natural-Sitzung, ohne dass eine Benutzerinteraktion erforderlich ist. Vordefinierte Beispiele für `NTTZ`-Makrodefinitionen sind in dem ausgelieferten Modul `NATCONFG` verfügbar.

Bezugspunkt für die automatische Sommerzeitumstellung ist die aktuelle Maschinenzeit, also die UTC (GMT)-Zeit. Je nachdem, in welchem Zeitraum sich die aktuelle Maschinenzeit befindet, wird die aktuelle Ortszeit ermittelt. Die Unterstützung der automatischen Sommerzeitumstellung gilt derzeit für den Zeitraum von 2002 bis 2041.

Die folgenden Themen werden in diesem Abschnitt behandelt:


- NTTZ-Makro - Überlegungen und Beschränkungen
- Syntax des Makros NTTZ
- NTTZ-Makro-Parameter
- Beispiel für ein NTTZ-Makro


## NTTZ-Makro - Überlegungen und Beschränkungen

Folgende Überlegungen und Einschränkungen sind zu beachten:

1. **Zeitformat**

   Das grundlegende Zeitformat ist:

   ```
   +hh:mm:ss
   ```

   oder:

```
-hh:mm:ss
```

Gilt von 00:00:00 bis 23:59:59. Auch Abkürzungen sind zulässig, zum Beispiel: *hh*:*mm*. Das Pluszeichen (+) wird standardmäßig angenommen, das Minuszeichen (-) kann bei den Parametern TDON oder TDOFF erforderlich sein.

2. **UTC versus Ortszeit**

   Um einen eindeutigen Bezugspunkt für die Zeitumstellung zu haben, werden die NTTZ-Makroparameter SWTON und SWTOFF in UTC-Zeit angegeben, während die Wochentagsnamen und die Tageszahlen in den NTTZ-Makroparametern DSTON und DSTOFF in Ortszeit angegeben werden.

3. **Gleichzeitige Verwendung der Natural-Profilparameter DD, YD und TD**

   Die Natural-Profil-Parameter DD und YD haben keinen Einfluss auf die automatische Sommerzeitumstellung, weil die Umstellung auf Basis der aktuellen Maschinenzeit erfolgt.

   Es wird empfohlen, die gleichzeitige Verwendung von DD oder YD und dem Profilparameter TD=*zonename* zu vermeiden.

4. **Gleichzeitige Verwendung des Natural-Profilparameters TD und des User Exit CMCOTIME**

   Die gleichzeitige Verwendung des Profilparameters TD=*zonename* und des User Exit CMCOTIME (Maschinenzeit überschreiben) wird nicht empfohlen, da eine Änderung der Maschinenzeit (TOD Clock) zu unvorhersehbaren Ergebnissen bei der automatischen Umschaltung führen kann, die mit TD=*zonename* aufgerufen wird.

### Syntax des Makros NTTZ

Die Syntax des Makros NTTZ lautet wie folgt:

```
        NTTZ ZONE=value;                                          *
             TDON=value,                                          *
             TDOFF=value,                                         *
             SWTON=value,                                         *
             SWTOFF=value,                                        *
             DSTONvalue,                                          *
             DSTOFF=value
```

## NTTZ-Makro-Parameter

ZONE | TDON | TDOFF | SWTON | SWTOFF | DSTON | DSTOFF

### ZONE - Time Zone Name

ZONE=*value* gibt den Namen der Zeitzone an, auf die mit dem Parameter TD verwiesen werden kann. Die erste Vorkommen eines Namens wird ausgewählt.

| Wert | Erläuterung |
|---|---|
| 32 Zeichen. | Die maximale Länge eines Zeitzonennamens beträgt 32 Zeichen, um beschreibende benutzerdefinierte Zonennamen zu ermöglichen, z. B. den Namen der Hauptstadt eines Landes. |

### TDON - Differenz der lokalen Sommerzeit zur UTC-Zeit

TDON=*value* gibt die Differenz der lokalen Sommerzeit zur UTC-Zeit (früher GMT) an.

| Wert | Erläuterung |
|---|---|
| +*hh*:*mm*:*ss*<br>or<br>-*hh*:*mm*:*ss* | Siehe auch **Zeitformat**. |

**Anmerkungen:**

1. Wenn nur der Parameter TDON definiert ist, erhält der Benutzer die Anzeige der Ortszeit als seine Zonenzeit, ohne automatische Umschaltung auf und von Sommerzeit.
2. Der Parameter TDON entspricht dem Parameter SWTON.

### TDOFF - Differenz der lokalen Zonenzeit zur UTC-Zeit

TDOFF=*value* gibt die Differenz der lokalen Zonenzeit zur UTC-Zeit (früher GMT) an.

| Wert | Erläuterung |
|---|---|
| +*hh*:*mm*:*ss*<br>or<br>-*hh*:*mm*:*ss* | Siehe auch **Zeitformat**. |

**Anmerkung:** Dieser Parameter entspricht dem Parameter SWTOFF.

### SWTON - Zeitpunkt des Beginns der Sommerzeit

SWTON=*value* gibt den UTC-Zeitpunkt an, an dem die Sommerzeit eingeschaltet wird.

| Wert | Erläuterung |
|---|---|
| *hh:mm:ss* | Siehe auch **Zeitformat**. |

### SWTOFF - Zeitpunkt des Endes der Sommerzeit

`SWTOFF=value` gibt den UTC-Zeitpunkt an, an dem die Sommerzeit ausgeschaltet wird.

| Wert | Erläuterung |
|---|---|
| *hh:mm:ss* | Siehe auch **Zeitformat**. |

### DSTON - Datum des Beginns der Sommerzeit

`DSTON=(value1,value2,value3,value4,day-number)` bezeichnet den Tag, an dem die Sommerzeit eingeschaltet wird.

| Wert | Mögliche Einstellungen |
|---|---|
| *value1* | FIRST, SECOND, THIRD, FOURTH oder LAST. |
| *value2* | MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY oder SUNDAY. |
| *value3* | AFTER, BEFORE oder IN. |
| *value4* | JANUARY ... DECEMBER. |
| *day-number* | Eine gültige Tageszahl für den jeweiligen Monat.<br><br>Der Standardwert ist 1. |

**Anmerkungen:**

1. Das Schlüsselwort LAST erfordert das Schlüsselwort BEFORE oder IN.
2. Wenn das Schlüsselwort IN angegeben wird, darf keine Tageszahl (*day number*) angegeben werden.

### DSTOFF - Datum, an dem die Sommerzeit endet

`DSTOFF=(value1,value2,value3,value4,day-number)` bezeichnet den Tag, an dem die Sommerzeit abgeschaltet wird.

| Wert | Mögliche Einstellungen |
|---|---|
| *value1* | FIRST, SECOND, THIRD, FOURTH oder LAST. |
| *value2* | MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY oder SUNDAY. |
| *value3* | AFTER, BEFORE oder IN. |
| *value4* | JANUARY ... DECEMBER. |
| *day-number* | Eine gültige Tageszahl für den jeweiligen Monat.<br><br>Der Standardwert ist 1. |

**Anmerkungen:**

1. Das Schlüsselwort `LAST` erfordert das Schlüsselwort `BEFORE` oder `IN`.
2. Wenn das Schlüsselwort `IN` angegeben wird, darf keine Tageszahl (*`day number`*) angegeben werden.

### Beispiel für ein NTTZ-Makro

Für die Sommerzeitumstellung in Westeuropa:

```
        NTTZ ZONE=MEZ,                                                *
              TDON=2,                                                 *
              TDOFF=+01:00:00,                                        *
              SWTON=01:00:00,                                         *
              SWTOFF=01:00:00,                                        *
              DSTON=(LAST,SUNDAY,IN,MARCH),                           *
              DSTOFF=(LAST,SUNDAY,IN,OCTOBER)
```

Zusätzliche Beispiele für andere Zeitzonen (Nord- und Südamerika, Asien usw.) sind in dem ausgelieferten Modul `NATCONFG` enthalten.

# 8 Natural-Speicherverwaltung


- Thread- und Nicht-Thread-Umgebungen ..... 60
- Puffertypen ..... 60
- Feste Puffer ..... 61
- Variable Puffer ..... 61
- Anpassung der Puffereigenschaften ..... 62

Dieses Dokument beschreibt, wie Natural Hauptspeicher zuordnet und verwendet. Ein von einer Natural-Nukleuskomponente angeforderter Speicherbereich wird als Buffer (*EN*) oder Puffer (*DE*) bezeichnet.

## Thread- und Nicht-Thread-Umgebungen

Es gibt zwei verschiedene Arten von Speicherumgebungen:

- Thread-Speicherumgebung (typisch für Mehrbenutzerumgebungen, z. B. CICS)
- Nicht-Thread-Speicherumgebung (typisch für Einzelbenutzerumgebungen, z. B. Batch)

In einer Thread-Umgebung wird ein großer, als „Thread“ bezeichneter Speicherbereich für eine Sitzung vorab zugewiesen. Die Größe des Threads muss vom Systemadministrator vordefiniert werden. Während einer Sitzung wird jede Pufferzuweisungsanforderung (GETMAIN) innerhalb ihres Threads von Natural selbst durchgeführt. Freier Speicherplatz aufgrund von Pufferfreigabeanforderungen (FREEMAIN) kann wiederverwendet werden.

Bei bestimmten Ereignissen (Terminal-Ein-/Ausgaben und langen Wartezeiten) kann der Thread-Speicher komprimiert und in einen externen Speicher (Roll File/Auslagerungsdatei) ausgelagert werden. Der freigegebene Thread kann von anderen Natural-Sitzungen wiederverwendet werden. Wenn eine unterbrochene Sitzung wieder aufgenommen werden soll, wird sie aus dem externen Speicher wieder in einen freien Thread verlegt.

Der Speicherplatz in dem Roll File, auf dem der komprimierte Thread-Speicher abgelegt wird, wird als Slot bezeichnet. Die Slot-Größe hat eine feste Länge und wird vom Systemadministrator festgelegt. Sie muss groß genug sein, um den größten komprimierten Thread-Speicher aufzunehmen. Im schlimmsten Fall kann sie gleich der Thread-Größe sein.

In einer Nicht-Thread-Umgebung werden alle Speicheranforderungen direkt an das Betriebs(sub)system weitergeleitet. Es wird keine Auslagerung/Einlagerung (Rollout/Rollin) durchgeführt, d. h. die Puffer für eine Sitzung werden bis zur Beendigung der Sitzung aufbewahrt, es sei denn, sie wurden vorher ausdrücklich freigegeben.

## Puffertypen

Es gibt drei verschiedene Arten von Puffern:

- Feste Puffer
- Variable Puffer
- Physische Puffer

**Feste Puffer und variable Puffer** haben ein 32-Byte-Präfix mit einem gemeinsamen Layout für alle Umgebungen. Das Präfix beginnt mit dem Puffernamen, gefolgt von 5 Pufferlängenfeldern (total, used low-end, max. used, used high-end, max. used high-end). Die verwendeten Längenfelder werden von den Komponenten gepflegt, denen die Puffer zugehören, und für die Thread-Komprimierung verwendet. Jeder Puffer hat eine eindeutige ID-Nummer (1-255) und kann nur einmal vorhanden sein. Einige Puffer werden während der Sitzungsinitialisierung zugewiesen, andere werden bei Bedarf zugewiesen. Das Systemkommando `BUS` kann benutzt werden, um Informationen über alle zurzeit zugewiesenen festen und variablen Puffer anzuzeigen. Die Eigenschaften der Puffer sind im Quellcode-Modul `NATCONFG` definiert, das in Ausnahmefällen angepasst werden kann (siehe ***Anpassung der Puffereigenschaften*** unten). Die Größe einiger Puffer kann durch einen Profilparameter festgelegt werden. Eine vollständige Liste solcher Puffer finden Sie unter dem Profilparameter `DS`.

**Physische Puffer** werden außerhalb des Threads zugewiesen. Sie haben kein Pufferpräfix und sind nicht eindeutig. Sie werden nur in Ausnahmefällen und nur vorübergehend verwendet. Physische Puffer werden bei der nächsten Terminal-E/A automatisch freigegeben. Es ist möglich, über den Profilparameter WPSIZE Workpools für physische Puffer zu definieren.

## Feste Puffer

In einer Thread-Umgebung werden feste Puffer nur vom unteren Ende des Threads aus zugeordnet. Im Gegensatz zu variablen Puffern können feste Puffer nicht relativ zum Thread verschoben werden und ihre Größe kann nicht erhöht oder verringert werden.

## Variable Puffer

In einer Thread-Umgebung werden variable Puffer vom oberen Ende des Threads aus zugeordnet. Wenn innerhalb des Threads kein Platz mehr ist, werden variable Puffer vorübergehend außerhalb des Threads zugeordnet. Bei der Thread-Komprimierung werden alle verwendeten Pufferteile in den Thread komprimiert. Wenn sie nicht in den Thread passen, wird die Sitzung abnormal beendet. Dies kann insbesondere bei der Verwendung großer dynamischer Variablen vorkommen.

Nach der Dekomprimierung des Threads können die variablen Puffer an eine andere Stelle innerhalb oder außerhalb des Threads verschoben worden sein. Die Größe der variablen Puffer kann auf Anforderung durch die sie verfügende Komponente erhöht oder verringert werden. Einige variable Puffer sind so definiert, dass sie während der Thread-Komprimierung automatisch verkleinert oder freigegeben werden.

Die Gesamtmenge des außerhalb des Threads zugeordneten Speichers kann durch den Profilparameter `OVSIZE` begrenzt werden.

## Anpassung der Puffereigenschaften

Alle Puffer werden im Quellcode-Modul NATCONFG durch NTBUFID-Makro-Definitionen definiert.

**Vorsicht:** Ändern Sie keine Puffereigenschaften außer den unten erläuterten MIN-, MAX- und CMPR-Parametereinstellungen, da die Ergebnisse unvorhersehbar sein können.

Es ist möglich, die Grenzen der Puffergröße durch die Parameter MIN und MAX des Makros NTBUFID zu ändern. Dies ist nur bei variablen Puffern (TYPE=VAR) sinnvoll. Die Grenzen für alle Puffer sind entweder durch die Standardeinstellung (0 - 2097151 KB) oder durch die Grenzen der entsprechenden Profilparameter definiert. Weitere Informationen finden Sie in der Beschreibung des Profilparameters DS. Die Grenzen der Puffergrößen-Profilparameter im Natural-Parametermodul werden von den MIN- und MAX-Parametern von NTBUFID nicht beeinflusst, aber die Grenzen für die dynamischen Profilpuffergrößen-Parameter werden von MIN und MAX überschrieben.

Das Setzen des MAX-Parameters auf einen Wert in KB bedeutet, das die Größe dieses Puffers während der Ausführung der Sitzung dieses Maximum nicht überschreiten darf. Dies kann zu Laufzeitfehlern führen, wenn mehr Pufferspeicher für den gewünschten Puffer angefordert wird.

Wenn Sie den Parameter MIN auf einen Wert in KB setzen, bedeutet dies, dass die Größe dieses Puffers während der Ausführung der Sitzung nicht unter diesem Wert liegen darf. Im Falle der 3GL-CALLNAT-Schnittstelle (NAT3GCAN) ist die Einstellung eines Puffer-Minimalwertes beispielsweise für die folgenden Puffer sinnvoll, da die Größe dieser Puffer auf einer niedrigeren Natural-Programmebene, die von einem 3GL-Programm aufgerufen wird, nicht erhöht werden darf.

| Puffer | Zweck |
|---|---|
| DATSIZE | Datenbereiche |
| GLBTOOL | Dienstprogramm-GDA |
| GLBUSER | Benutzer-GDA |
| GLBSYS | System-GDA |
| AIVDAT | AIV-Bereich |
| CONTEXT | Kontextvariablen |

Der Parameter CMPR des Makros NTBUFID definiert den Kompressionsoptimierungsalgorithmus für den Puffer. Er entspricht dem Profilparameter CMPR, der den Standardwert festlegt. Weitere Informationen über die möglichen Parameterwerte finden Sie unter *CMPR - Standard-Algorithmus zur Komprimierungsoptimierung* in der *Parameter-Referenz*-Dokumentation.

Beispiel für eine Definition von Puffereigenschaften:

```
DATSIZE NTBUFID ID=GETMDATA,TYPE=VAR+INI,CMPR=OPT2,MAX=512
```

Weitere Informationen zu Profilparametern, die sich auf die Puffergrößen auswirken, finden Sie unter *Speicherverwaltung* in der *Parameter-Referenz*-Dokumentation.

# II Profilparameter anwenden

Dieser Teil beschreibt die Grundlagen und Regeln, die bei der Anwendung von Natural-Profilparametern in einer Großrechnerumgebung gelten.

| | |
|---|---|
| **Natural-Parameter-Hierarchie** | Überblick über die hierarchische Struktur der verschiedenen Ebenen, auf denen Natural-Parameter gesetzt werden können. Anhand von Beispielen werden die verschiedenen Szenarien veranschaulicht. |
| **Zuweisung von Parameterwerten** | Zuweisung von Werten zu Profilparametern: statisch, dynamisch und zur Laufzeit. |
| **Generierung eines Natural-Parametermoduls** | Generierung eines Natural-Parametermoduls unter Verwendung des Makros `NTPRM` und anderer Parametermakros. |

**Verwandte Themen:**

- Ausführliche Informationen zu einzelnen Profilparametern finden Sie in der *Parameter-Referenz*-Dokumentation.
- Eine Übersicht über die nach Kategorien gruppierten Profilparameter finden Sie unter *Profilparameter sortiert nach Kategorien* in der *Parameter-Referenz*-Dokumentation.

# 9 Natural-Parameter-Hierarchie


- Übersicht über die Natural-Parameterhierarchie ..... 68
- Allgemeine Regeln für die Verwendung von Parametern ..... 68
- Natural-Parametermodul ..... 69
- Vordefinierte dynamische Parameter-Sets ..... 70
- Vordefinierte Benutzer-Parameter-Profile ..... 70
- Dynamische Parametereingabe ..... 70
- Natural-Security-Definitionen ..... 71
- Profilparametereinstellungen während der Sitzung zeigen/ändern ..... 71
- Einstellungen auf Programm-/Statement-Ebene ..... 72
- Einstellungen der Entwicklungsumgebung ..... 72
- Beispiele für die Parameterauswertung ..... 72

Dieses Dokument beschreibt die hierarchische Struktur der verschiedenen Ebenen, auf denen Natural-Profilparameter gesetzt werden können. Zur Veranschaulichung des Szenarios werden verschiedene Beispiele angeführt.

Ausführliche Informationen zu einzelnen Profilparametern finden Sie in der *Parameter-Referenz*-Dokumentation.

## Übersicht über die Natural-Parameterhierarchie

Natural-Profilparameter beeinflussen das Erscheinungsbild und die Reaktion der Arbeitsumgebung eines Natural-Benutzers. Diese Parameter werden auf verschiedenen hierarchisch organisierten Ebenen eingestellt, wie in der folgenden Tabelle dargestellt (Priorität von hoch bis niedrig).

| Ebene | Kurzbeschreibung/Verweise auf detaillierte Beschreibungen |
|---|---|
| Während der Sitzung | ■ Einstellungen der Entwicklungsumgebung<br>■ Einstellungen auf Programm-/Statement-Ebene<br>■ Profilparametereinstellungen während der Sitzung<br>■ Natural-Security-Definitionen |
| Dynamisch beim Sitzungsstart | ■ Dynamische Parametereingabe<br>■ Vordefinierte Benutzer-Parameter-Profile<br>■ Vordefinierte dynamische Parameter-Sets<br>■ Alternatives Natural-Parametermodul |
| Statisch | Natural-Parametermodul |

Die hierarchisch organisierten Ebenen werden in den referenzierten Abschnitten behandelt, beginnend mit der niedrigsten und endend mit der höchsten Priorität.

## Allgemeine Regeln für die Verwendung von Parametern

Es gelten die folgenden allgemeinen Regeln:

- Ein Parameterwert, der auf einer höheren Ebene festgelegt wird, überschreibt den auf einer niedrigeren Ebene definierten Wert (Ausnahmen: `PROFILE`, `SYS`, `DYNPARM` und einige andere Parameter, die durch Hinzufügen von Werten funktionieren).
- Dynamische Parameter beim Sitzungsstart haben Reihenfolgepriorität, d.h. sie werden von links nach rechts ausgewertet.

- **Beispiel:**

```
ESIZE=20,DATSIZE=60,ESIZE=100
```

  Der resultierende Wert ist `ESIZE=100`.

- Nicht alle Parameter, die auf einer niedrigeren Ebene verfügbar sind, können auch auf einer höheren Ebene definiert werden.

## Natural-Parametermodul

Ein Natural-Parametermodul enthält eine Reihe von Profilparametern, die für die Konfiguration Ihrer Natural-Umgebung erforderlich sind.

Ein Natural-Parametermodul wird aus dem `NTPRM`**-Makro** und zusätzlichen Makros während des Installationsvorgangs erstellt, wie in ***Generierung eines Natural-Parametermoduls*** beschrieben.

Je nach Ihren persönlichen Präferenzen können Sie mehr als ein Natural-Parametermodul verwenden, z. B. ein Modul für Natural-Batch- und eines für Natural-Online-Sitzungen.

Das Natural-Parametermodul bildet die unterste Ebene der Natural-Parameterhierarchie.

Neben dem Natural-Parametermodul benötigen Sie möglicherweise ein zusätzliches Parametermodul für ein Natural-Zusatzprodukt, das Sie in Ihrer Umgebung einsetzen, z. B. die Natural-CICS-Schnittstelle.

- Alternatives Natural-Parametermodul

### Alternatives Natural-Parametermodul

Zusätzlich zu einem Natural-Parametermodul, das statisch mit dem Nukleus verlinkt ist, können Sie alternative Natural-Parametermodule definieren, die in einer TP- oder Betriebssystem-Library gespeichert sind. Sie können sie dazu verwenden, die Parameterdefinitionen des statischen Natural-Parametermoduls für eine Natural-Sitzung zu überschreiben, indem Sie den Profilparameter `PARM` angeben, wie in der *Parameter-Referenz*-Dokumentation beschrieben. Ausnahme: `CSTATIC`-Parameterdefinitionen werden nicht überschrieben.

**Wichtig:** Der Profilparameter `PARM` sollte als erster Parameter in einem dynamischen Parameterstring erscheinen, da sonst das alternative Natural-Parametermodul alle zuvor im dynamischen Parameterstring eingetragenen Parametereinstellungen überschreibt.

**Verwendungsbeschränkungen**

Sie können die Verwendung eines alternativen Natural-Parametermoduls auf einen bestimmten Benutzer oder auf mehrere Benutzer beschränken, indem Sie das Makro `NTUSER` verwenden.

Definieren Sie in diesem Makro die Kennungen (IDs) der Benutzer, die berechtigt sind, dieses Parametermodul zu verwenden. Nur diese Benutzer dürfen dann den Namen des Parametermoduls mit dem Profilparameter `PARM` angeben.

## Vordefinierte dynamische Parameter-Sets

Mit dem Assembler-Makro `NTSYS` können Parameter-Sets vordefiniert werden, die in einem Natural-Parametermodul benannt werden. Diese Sets können beim Aufruf von Natural unter ihrem Namen angesprochen werden, sofern das entsprechende Parametermodul aktiv ist.

Beim Aufruf reagieren die vordefinierten Parameter-Sets genauso wie dynamisch an dieser Stelle eingegebene Parameter.

Siehe auch Profilparameter `SYS`.

## Vordefinierte Benutzer-Parameter-Profile

Mit dem Natural-Dienstprogramm SYSPARM können Sie individuelle Profile erstellen, die in einer Systemdatei gespeichert werden. Jedes Profil erhält einen eindeutigen Zeichennamen. In einem solchen Profil können Sie Werte für beliebige dynamische Natural-Parameter festlegen.

Die mit dem Dienstprogramm `SYSPARM` erstellten Profile werden beim Aufruf von Natural über den Parameter `PROFILE` aktiviert.

Mit dem Profilparameter `USER` können Sie die Verwendung eines Profils auf einen bestimmten Benutzer oder auf mehrere Benutzer einschränken.

Beim Aufrufen verhalten sich die vordefinierten Parameterprofile genauso wie dynamisch eingegebene Parameter an dieser Stelle.

## Dynamische Parametereingabe

Fast alle Parameter können dynamisch überschrieben werden, wenn Natural gestartet wird. Dynamische Parameter werden strikt sequentiell ausgewertet.

Diese allgemeine Überschreibmöglichkeit kann jedoch durch den Profilparameter DYNPARM generell oder für bestimmte Parameter eingeschränkt werden (nur dynamisch, z.B. in einem Profil).

Mit dem Makro `NTDYNP` im Natural-Parametermodul können Sie analoge Einstellungen vornehmen. Dadurch wird jedoch die Verwendung des Profilparameters `DYNPARM` unterbunden.

Sie können das Dataset `CMPRMIN` verwenden, um dynamische Parameter im Batch-Modus unter z/OS oder in Batch-ähnlichen Systemen wie TSO oder BMP-Umgebungen unter IMS TM zu definieren.

Der Vorteil dieser Methode ist, dass Sie nicht die JCL ändern müssen, wenn Sie Natural-Einstellungen ändern wollen. Außerdem wird die Längenbeschränkung der Parameter-Zeichenketten (z.B. 100 Zeichen unter z/OS) vermieden.

## Natural-Security-Definitionen

Neben dem Schutz der Libraries, Dateien und Kommandos ermöglicht Natural Security auch die Einstellung bestimmter sitzungsrelevanter Profilparameter. Die Definitionen gelten für die aktuelle Library des Benutzers.

Die Benutzer können auch Einstellungen für ihre privaten oder Standard-Libraries festlegen.

Die aktuellen Security-Einstellungen (Session-Parameter) können mit dem Natural-Systemkommando `PROFILE` angezeigt werden.

Die Natural Security-Parameterdefinitionen werden nach den regulären Profilparametern ausgewertet, d. h. sie können diese überschreiben.

## Profilparametereinstellungen während der Sitzung zeigen/ändern

Mit dem Natural-Systemkommando `GLOBALS` oder dem Natural-Statement `SET GLOBALS` können Sie innerhalb und für die Dauer einer Natural-Sitzung bestimmte sitzungsrelevante Profilparameter anzeigen und setzen (ändern).

Diese Definitionen gelten für den Kommandomodus und für alle Programme, die während der aktuellen Sitzung ausgeführt werden.

Siehe auch ***Session Parameter zur Zuweisung von Parameterwerten zur Laufzeit*** oder Statement `SET GLOBALS`.

## Einstellungen auf Programm-/Statement-Ebene

Das Natural-Statement `FORMAT` kann in einem Programm verwendet werden, um Parameterwerte zu setzen, die für dieses Programm spezifisch gültig sind.

Darüber hinaus ist es möglich, bestimmte Parameter auf Statement-Ebene durch ein Terminalkommando zu setzen.

## Einstellungen der Entwicklungsumgebung

Im **Natural-Hauptmenü** (Main Menu) können Sie über die Option **Einstellungen der Entwicklungsumgebung** (Development Environment Settings) ein Untermenü aufrufen, das die Auswahl der Werkzeuge ermöglicht, die zur Überwachung und Einrichtung der Natural-Entwicklungsumgebung zur Verfügung stehen.

## Beispiele für die Parameterauswertung

Die nachfolgenden Beispiele basieren auf den folgenden Parametereinstellungen:

| Parameter | Parametermodul | Alternatives Parametermodul<br>`ALTPARM` | Benutzerprofil<br>`MYPROF` |
|---|---|---|---|
| `DATSIZE` | `40` | `50` | `60` |
| `DSIZE` | `6` | `2` (Standardwert) | Keine Angabe |
| `ESIZE` | `28` (Standardwert)<br>`NTSYS A: 40`<br>`NTSYS B: 50` | `NTSYS A: 60` | `80` |

Die folgenden Beispiele zeigen die Ergebnisse bei verschiedenen dynamischen Parameter-Zeichenketten.

### Beispiel 1: Keine dynamischen Parameter

| Resultierende Werte | Herkunft |
|---|---|
| DATSIZE 40 | **Parametermodul** |
| DSIZE 6 | Parametermodul |
| ESIZE 28 | Parametermodul |
| Sonstige: Standardwert | Parametermodul |

### Beispiel 2: PARM=ALTPARM

| Resultierende Werte | Herkunft |
|---|---|
| DATSIZE 50 | ALTPARM |
| Sonstige: Standardwert | ALTPARM |

### Beispiel 3: SYS=A

| Resultierende Werte | Herkunft |
|---|---|
| DATSIZE 40 | Parametermodul |
| DSIZE 6 | Parametermodul |
| ESIZE 40 | NTSYS-Makro im Parametermodul |

### Beispiel 4: PARM=ALTPARM,SYS=A

| Resultierende Werte | Herkunft |
|---|---|
| DATSIZE 50 | ALTPARM |
| DSIZE 2 | ALTPARM |
| ESIZE 60 | NTSYS-Makro in ALTPARM |

### Beispiel 5: PARM=ALTPARM,SYS=B

| Resultierende Werte | Herkunft |
|---|---|
| Fehler | ALTPARM enthält keine NTSYS B-Angabe |

**Beispiel 6:** `SYS=A,PROFILE=MYPROF`

| Resultierende Werte | Herkunft |
|---|---|
| DATSIZE 60 | MYPROF |
| DSIZE 6 | Parametermodul |
| ESIZE 80 | MYPROF |

**Beispiel 7:** `SYS=A,PROFILE=MYPROF,ESIZE=100`

| Resultierende Werte | Herkunft |
|---|---|
| DATSIZE 60 | MYPROF |
| DSIZE 6 | Parametermodul |
| ESIZE 100 | Dynamischer Parameter |

**Beispiel 8:** `PROFILE=MYPROF,SYS=A`

| Resultierende Werte | Herkunft |
|---|---|
| DATSIZE 60 | MYPROF |
| DSIZE 6 | Parametermodul |
| ESIZE 40 | NTSYS-Makro im Parametermodul |

**Beispiel 9:** `DSIZE=8,SYS=A,PROFILE=MYPROF,PARM=ALTPARM`

| Resultierende Werte | Herkunft |
|---|---|
| DATSIZE 50 | ALTPARM |
| Sonstige: Standardwert | ALTPARM |

# 10 Zuweisung von Parameterwerten


- Quellen für die Parameterwertzuweisung ..... 76
- Statische Zuweisung von Parameterwerten ..... 77
- Dynamische Zuweisung von Parameterwerten ..... 78
- Session-Parameter zur Zuweisung von Parameterwerten zur Laufzeit ..... 79

Dieses Kapitel informiert darüber, wie Profilparameter statisch, dynamisch und zur Laufzeit mit Werten belegt werden.

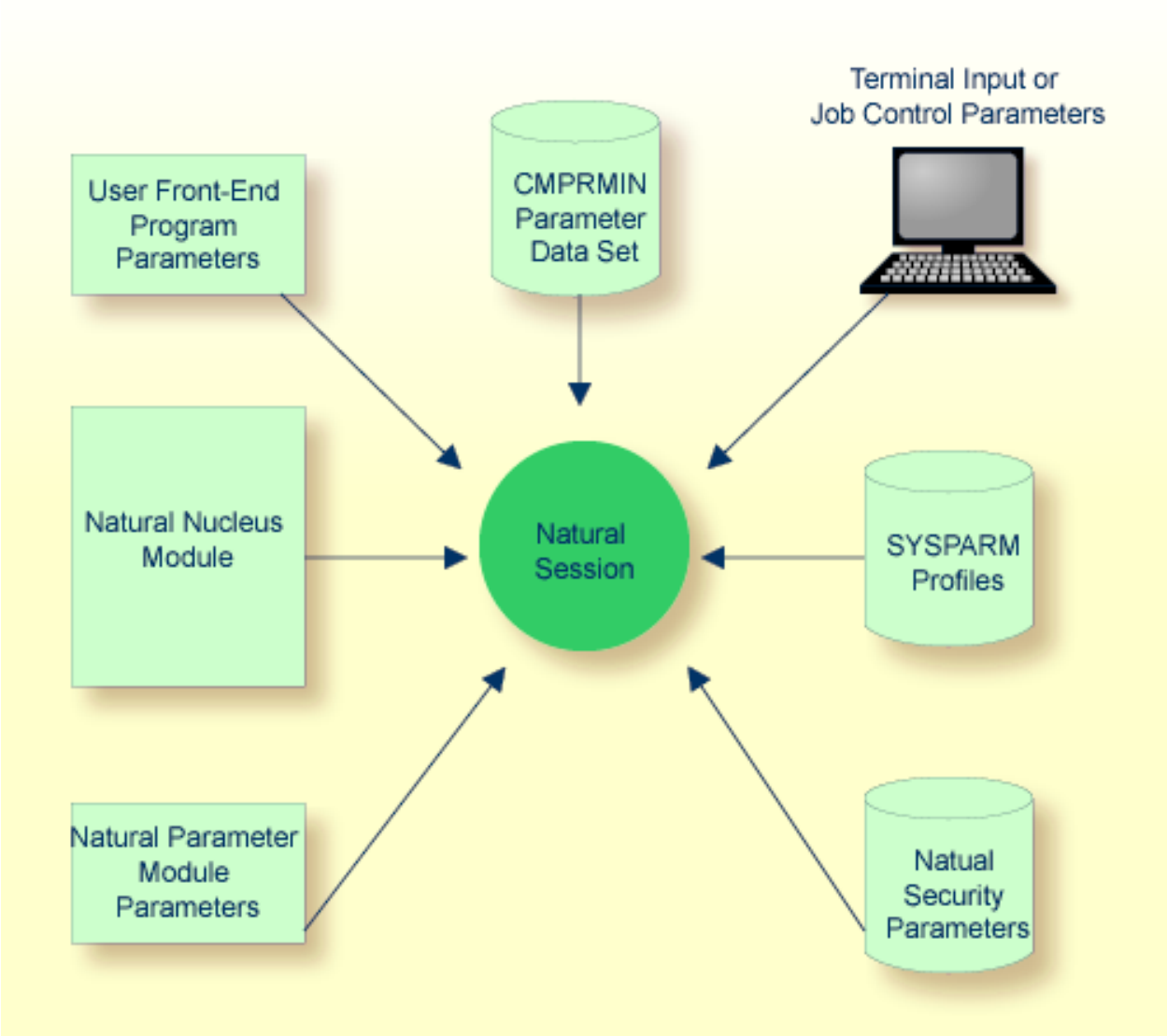
Ausführliche Informationen zu einzelnen Profilparametern finden Sie in der *Parameter-Referenz*-Dokumentation.

## Quellen für die Parameterwertzuweisung

Die Werte für Profilparameter werden aus drei Quellen übernommen:

1. **Statische Zuweisungen**
   Profilparameter, die im Parametermakro `NTPRM` angegeben sind, und **weitere Parametermakros**, die im Natural-Parametermodul enthalten sind.
2. **Dynamische Zuweisungen**
   Parameter, die für die Ausführung der Natural-Sitzung festgelegt wurden. Diese Parameter haben Vorrang vor den statischen Zuweisungen und sind für die aktuelle Natural-Sitzung gültig. Dynamische Parameter können von einem Front-End-Programm, dem Parameter-Dataset (`CMPRMIN`), der JCL zur Sitzungsinitialisierung, der Terminaleingabe oder Natural Security übergeben werden. Darüber hinaus ist es möglich, bestimmte Parameter durch -Statements aus Natural-Programmmen zu überschreiben.
3. **Session-Parameter**
   Parameter, die mit dem Systemkommando `GLOBALS` (oder einem `SET GLOBALS`-Statement) innerhalb der aktuellen Natural-Sitzung angegeben werden. Die Parameter setzen statische und dynamische Zuweisungen außer Kraft.

Grafische Darstellung der Natural-Parameter-Zuweisung:




## Statische Zuweisung von Parameterwerten

Das **Natural-Parametermodul** wird für die statische Zuweisung von Profilparametern für alle Natural-Umgebungen verwendet.

Im Natural-Parametermodul können Sie das Makro `NTPRM` und einige andere Makros verwenden, um die Parameter anzugeben.

Alle im Natural-Parametermodul vorgenommenen Parametereinstellungen (mit Ausnahme des Parameters `CSTATIC`) können zu Beginn einer Natural-Sitzung dynamisch überschrieben werden.

Für einige Profilparameter wird im Natural-Parametermodul ein entsprechendes Makro für die statische Zuweisung verwendet. Die Syntax der statischen und dynamischen Angaben unterscheidet sich daher geringfügig und hat die folgende allgemeine Form:

| | |
|---|---|
| Statisch: | `macro-name keyword1=value,keyword2=value1,value2,...` |
| Dynamisch: | `parameter-name=(keyword1=value,keyword2=value1,value2,...)` |

**Beispiel:**

- Makro im Natural-Parametermodul: `NTSORT WRKSIZE=500,EXT=ON`
- Äquivalenter dynamischer Profilparameter: `SORT=(WRKSIZE=500,EXT=ON)`

Wenn es ein anderes als das `NTPRM`-Parametermakro für einen Profilparameter gibt, wird dieses Makro in der individuellen Parameterbeschreibung angezeigt.

Siehe auch den Abschnitt *Generierung eines Natural-Parametermoduls*.

## Dynamische Zuweisung von Parameterwerten

Sie können Profilparameter zu Beginn einer Natural-Sitzung dynamisch festlegen, um für die Dauer einer einzelnen Natural-Sitzung einzelne Profilparametereinstellungen des **Natural-Parametermoduls** zu überschreiben.

**Beispiel:**

```
NUCNAME='NATNUC#5',IM=D,INTENS=1,DU=OFF,FUSER=(10,32),PROGRAM=' ',
WORK=((1),AM=STD,DEST=WORK1,OPEN=INIT),PS=60,LS=120
```

Alle Profilparameter können dynamisch angegeben werden, außer `CSTATIC`, das nur im **Natural-Parametermodul** statisch angegeben werden kann:

Die dynamischen Parameterzuweisungen werden durch (ein oder mehrere) Kommata oder Leerzeichen getrennt. Enthält der Wert für einen dynamischen Parameter nicht-alphanumerische Zeichen oder Sonderzeichen, muss der Wert in Hochkommata eingeschlossen angegeben werden. Welche Zeichen Sonderzeichen sind, wird im Zeichentabellenmakro `NTSCTAB` von `NATCONFG` definiert. Siehe *Natural-Konfigurationstabellen*.

Die Verwendung von dynamischen Parametern kann durch das Makro `NTDYNP` oder den entsprechenden dynamischen Profilparameter `DYNPARM` aktiviert/deaktiviert werden.

Einfacher können Sie dynamische Parameter-Sets mit dem Profilparameter `PROFILE` oder `SYS` angeben. Darüber hinaus können Sie eine Reihe von dynamischen Parametern in Natural Security setzen.

In dynamische Parameter können Sie Kommentarzeichenfolgen einfügen. Ein Kommentar beginnt mit den Begrenzungszeichen `/*` und endet mit `*/`. Fehlt das Begrenzungszeichen am Ende der Kommentarzeichenfolge, wird bei der Sitzungsinitialisierung eine Fehlermeldung ausgegeben.

**Beispiel:**

```
PARM=MYPARMS /* my comment */ ADANAME=ADALNKR,PROFILE=MYPROF
```

Die Übergabe der dynamischen Parametereinstellungen an Natural erfolgt, wenn die Sitzung gestartet wird. Die Methode, mit der die Parameterwerte an Natural übergeben werden, hängt von der jeweiligen Umgebung ab.

**Beispiel für z/OS im Batch-Modus:**

- Die Werte werden mit dem Schlüsselwort `PARM` in der `EXEC`-Job-Steueranweisung angegeben, die Natural initiiert.
- Darüber hinaus können dynamische Parameter im Dataset `CMPRMIN` angegeben werden.
- Darüber hinaus ist es möglich, ein Front-End-Programm zu schreiben, das die Kontrolle an Natural mit dynamischen Parametern für die Sitzung gemäß den z/OS-Standards übergibt.

## Session-Parameter zur Zuweisung von Parameterwerten zur Laufzeit

Einigen Profilparametern kann zur Laufzeit innerhalb einer Natural-Sitzung ein Wert zugewiesen werden, indem ein entsprechenden Session-Parameter verwendet wird. Der Wert dieses Session-Parameters hat dann Vorrang vor dem Wert des Profilparameters.

Wenn für einen Profilparameter ein entsprechender Session-Parameter existiert, wird dies in der Beschreibung des Profilparameters angegeben.

Session-Parameter werden mit dem Systemkommando `GLOBALS` angegeben.

Beschreibungen der Session-Parameter sind in der *Parameter-Referenz*-Dokumentation enthalten. Ausführliche Informationen zu Systemkommandos finden Sie in der *Systemkommandos*-Dokumentation.

**Beispiel:**

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

Session-Parameter können auch in einem Natural-Programm angegeben werden, und zwar mit dem Statement `SET GLOBALS`.

Manche Profilparameter können auch innerhalb einer Natural-Sitzung durch ein Terminalkommando außer Kraft gesetzt werden. Wenn zu einem Profilparameter ein entsprechendes Terminalkommando existiert, wird dies in der Beschreibung des Profilparameters angegeben. Terminalkommandos werden in der *Terminalkommandos*-Dokumentation beschrieben.

**Beispiel:**

```
SET CONTROL 'T=3279'
```

Der Wert des Profilparameters `TTYPE` für den Terminaltyp 3279 wird durch die Angabe `'T=3279'` im `SET CONTROL`-Statement, die dem Terminalkommando `%T=3279` entspricht, überschrieben.

# 11 Generierung eines Natural-Parametermoduls


- Parameter-Makro NTPRM ..... 82
- Zusätzliche Makros im Natural-Parametermodul ..... 83
- Beispiel für Makros im Natural-Parametermodul ..... 85

Das Natural-Parametermodul wird während der Installation von Natural generiert. Dazu müssen Sie die entsprechenden Installationsjobs ausführen, die von System Maintenance Aid (SMA) bereitgestellt werden. Die Beschreibungen dieser Jobs sind bei den entsprechenden Installationsschritten in der *Installation für z/OS*-Dokumentation enthalten.

Ein Natural-Parametermodul wird aus dem Parametermakro `NTPRM` und bei Bedarf aus zusätzlichen Parametermakros gebaut. Siehe ***Zusätzliche Makros im Natural-Parametermodul***. Sie können die von SMA bereitgestellten Standard-Parametereinstellungen ändern und die Installationsjobs nach Ihren Bedürfnissen anpassen.

**Um ein Natural-Parametermodul mit SMA-Jobs zu bauen:**

1. Passen Sie die Profilparameter im Standard-`NTPRM`-Parametermakro an Ihre Bedürfnisse an, indem Sie die `NTPRM`-Makro-Syntax anwenden (siehe *Parameter-Referenz*-Dokumentation).
2. Fügen Sie bei Bedarf zusätzliche Parametermakros in beliebiger Reihenfolge *nach* dem `NTPRM` hinzu.
3. Assemblieren Sie das Natural-Parametermodul und verlinken Sie es mit dem umgebungsabhängigen Nukleus (siehe *Installation für z/OS*-Dokumentation).
4. Verlinken Sie das Natural-Parametermodul mit dem umgebungsunabhängigen Nukleus (siehe *Installation für z/OS*-Dokumentation), wenn eine der folgenden Bedingungen erfüllt ist:

   - Ihr `NTPRM`-Makro enthält `CSTATIC`-Einträge.
   - Ihr Natural-Parametermodul enthält ein `NTCSTAT`-Makro.

   Der Profilparameter `CSTATIC` und das Parametermakro `NTCSAT` werden in der *Parameter-Referenz*-Dokumentation beschrieben.

   Dieses Kapitel behandelt die folgenden Themen:

## Parameter-Makro NTPRM

Das Parameter-Makro `NTPRM` ist obligatorisch. Es muss im Natural-Parametermodul angegeben werden. Das Makro `NTPRM` enthält die wichtigsten Profilparametereinstellungen, die zur Konfiguration von Natural erforderlich sind. Alle Profilparameter, bei denen in der Beschreibung der einzelnen Parameter in der *Parameter-Referenz*-Dokumentation kein Parameter-Makro angegeben ist, werden im `NTPRM`-Makro definiert.

Siehe auch *NTPRM-Makro-Syntax* (*Parameter-Referenz*-Dokumentation) und ***Beispiel für Makros im Natural-Parametermodul***.

# Zusätzliche Makros im Natural-Parametermodul

Im Natural-Parametermodul können Sie nach dem NTPRM-Makro die in der folgenden Tabelle aufgeführten Parameter-Makros angeben. Diese Makros können Sie in beliebiger Reihenfolge angeben.

Die Verwendung eines oder mehrerer zusätzlicher Parameter-Makros hängt von Ihren individuellen Systemanforderungen und den in Ihrer Natural-Umgebung installierten Produkten ab.

Der Name eines zusätzlichen Parameter-Makros und seine Syntax sind in der Einzelbeschreibung des entsprechenden Profilparameters in der *Parameter-Referenz*-Dokumentation enthalten.

Siehe auch ***Beispiel für Makros im Natural-Parametermodul***.

## Namenskonventionen und Übersicht über die Makros

Zu jedem zusätzlichen Parameter-Makro gehört in der Regel ein entsprechender dynamischer Profilparameter.

Der Name eines zusätzlichen Parameter-Makros beginnt mit NT. Daran anschließend folgt der Name des entsprechenden Profilparameters.

Beispiel: Das Parameter-Makro NTBPI entspricht dem Profilparameter BPI.

Ausnahmen von dieser Regel sind in der folgenden Tabelle angegeben, die eine Übersicht über die verfügbaren Makros enthält.

| Parameter-Makro | Zweck |
|---|---|
| NTALIAS | Definiert externe Alias-Namen für die Module, die mit dem Natural-Nukleus verlinkt sind.<br><br>Entsprechender dynamischer Profilparameter: RCALIAS. |
| NTBPI | Weist Natural-Sitzungen Buffer Pools zu. |
| NTCCTAB | Definiert Druckersteuerungssequenzen. |
| NTCFICU | Aktiviert die Unterstützung von Unicode und Codepages. |
| NTCMPO | Legt Kompilierungsoptionen fest. |
| NTCOMP | Legt Konfigurationseinstellungen für die Natural Com-plete/SMARTS-Schnittstelle fest (Natural under Com-plete/SMARTS). |
| NTCSTAT | Legt die Module fest, die mit dem Natural-Nukleus zu verlinken sind.<br><br>Entsprechender dynamischer Profilparameter: CSTATIC. |
| NTDB | Definiert Datenbanktypen und Optionen für Datenbanken. |
| NTDB2 | Legt Konfigurationseinstellungen für Natural for Db2 fest. |

| Parameter-Makro | Zweck |
|---|---|
| NTDBGAT | Ermöglicht das Debugging von externen Natural-Anwendungen. |
| NTDS | Legt die Größe von Speicherpuffern fest. |
| NTDYNP | Steuert die Verwendung von dynamischen Profilparametern.<br><br>Entsprechender dynamischer Profilparameter: DYNPARM. |
| NTEDBP | Steuert den Buffer Pool-Betrieb des Software AG Editor. |
| NTIMSP | Legt Konfigurationseinstellungen für die Natural IMS TM-Schnittstelle (Natural unter IMS TM) fest.<br><br>Keine dynamische Parameterangabe möglich. |
| NTIMSPE | Definiert umfeldspezifische Parameter-Sets für die Natural IMS TM-Schnittstelle (Natural under IMS TM).<br><br>Keine dynamische Parameterangabe möglich. |
| NTIMSPT | Definiert Natural-Transaktionscodes für das Natural IMS TM-Interface (Natural unter IMS TM).<br><br>Keine dynamische Parameterangabe möglich. |
| NTLFILE | Verknüpft physische Datenbankdateien mit logischen Systemdateien. |
| NTOPRB | Steuert die Verwendung von Datenbank-Open/Close-Kommandos für Adabas oder VSAM. |
| NTOPT | Steuert die Verwendung und die Optionseinstellungen des Natural Optimizer Compiler. |
| NTOSP | Legt Konfigurationseinstellungen für die z/OS-Batch-Schnittstelle fest. |
| NTPGP | Legt Eigenschaften für externe Programme fest. |
| NTPRINT | Legt Druckdateizuweisungen fest. |
| NTRDC | Konfiguriert den Natural Data Collector und seine Trace-Aufzeichnungsfunktion, die von den Dienstprogrammen SYSRDC und dem Natural Profiler verwendet wird. |
| NTRPC | Steuert die Handhabung des Natural RPC (Remote Procedure Call). |
| NTSCTAB | Überschreibt die Scanner-Zeichendefinitionen im NATCONFG-Modul. |
| NTSORT | Steuert das Sortierprogramm, das beim SORT-Statement verwendet wird. |
| NTSYS | Definiert Sets dynamischer Profilparameter. |
| NTTAB | Überschreibt die Definitionen für die Umsetzung von Ausgabezeichen im Modul NATCONFG. |
| NTTAB1 | Definiert alternative Tabellen für die Umsetzung von Ausgabezeichen. |
| NTTAB2 | Definiert alternative Tabellen für die Umsetzung von Eingabezeichen. |
| NTTABA1 | Überschreibt die EBCDIC-zu-ASCII-Konvertierungsdefinitionen im NATCONFG-Modul. |
| NTTABA2 | Überschreibt die ASCII-EBCDIC-Konvertierungsdefinitionen im NATCONFG-Modul. |
| NTTABL | Überschreibt die Umsetzungsdefinitionen der „SYS“ Library-Ausgabe im NATCONFG-Modul. |
| NTTF | Konvertiert Datenbankkennungen und Dateinummern während der Programmausführung. |
| NTTRACE | Legt die mittels Tracing zu verfolgenden Natural-Komponenten fest. |

| Parameter-Makro | Zweck |
|---|---|
| NTTSOP | Legt die Konfigurationseinstellungen für die Natural TSO-Schnittstelle fest (Natural under TSO). |
| NTUSER | Schränkt die Verwendung von dynamischen Parameterstrings und alternativen Natural-Parametermodulen ein. |
| NTUTAB1 | Überschreibt die Definitionen für die Umwandlung von Groß- und Kleinschreibung im Modul NATCONFG. |
| NTUTAB2 | Überschreibt die Groß-/Kleinschreibung-Konvertierungsdefinitionen im NATCONFG-Modul. |
| NTVEXIT | Gibt User Exits für VSAM-Dateien an.<br><br>Entspricht dem Schlüsselwort-Subparameter EXIT des dynamischen Profilparameters VSAM. |
| NTVLSR | Definiert lokale gemeinsam genutzte Ressourcen-Subpools für VSAM-Dateien.<br><br>Entspricht dem Schlüsselwort-Subparameter LSR des dynamischen Profilparameters VSAM. |
| NTVSAM | Legt Konfigurationseinstellungen für Natural for VSAM fest. |
| NTVTVSD | Aktiviert die DFSMS-Transactional VSAM Services.<br><br>Entspricht dem Schlüsselwort-Subparameter TVSD des dynamischen Profilparameters VSAM. |
| NTWEBIO | Aktiviert oder deaktiviert Funktionen der Natural Web I/O Interface-Anzeige. |
| NTWORK | Gibt die Arbeitsdateien an, die während einer Sitzung verwendet werden sollen. |
| NTXML | Aktiviert oder deaktiviert die Statements PARSE XML und REQUEST DOCUMENT. |
| NTZIIP | Konfiguriert die zIIP-Verarbeitung (System z Integrated Information Processor) für z/OS. |

Siehe auch ***Beispiel für Makros im Natural-Parametermodul***.

## Beispiel für Makros im Natural-Parametermodul

In dem folgenden Beispiel für Makro-Definitionen im Natural-Parametermodul bezeichnet *vrs* bzw. *vr* eine Natural-Produktversion.

```
        NTPRM FNR=8,                          System File for NTPRM *
              DBID=001,                       Database ID for NTPRM *
              FNAT=(001,8),                   Natural System File   *
              FUSER=(001,9),                  Natural User File     *
              FDIC=(001,11),                  Predict System File   *
              FSEC=(001,10),                  Natural Security File *
              FREG=(001,52),                  Registry System File  *
              ESIZE=128,                      User Extension Area   *
              SLOCK=SPOD,                     Source Locking        *
              THSIZE=0,                       Thread Size           *
              UCONMAX=0,                      Max. Session Number   *
```

```
               CSTATIC=(CMMSG,                    Static. Modules Links *
               NSPPFUNC),                         Dummy Static. Module  *
               LE=OFF,                            Record Limit Error    *
               RECAT=OFF,                         Allow Stow of Macros  *
               PROFILE=,                          Profile Batch         *
               ADANAME=ADABAS,                    Adabas Link Routine   *
               ADASBV=OFF,                        Form. Buffer not Pass.*
               DFOUT=S,                           Output Format of Date *
               DFSTACK=S,                         Date Format for Stack *
               NUCNAME=NATvrsSH,                  Natural Nucleus Name  *
               AUTO=OFF,                          Automatic Logon       *
               PC=ON,                             PC Connection         *
               LS=250,                            Default Line Size     *
               PS=80,                             Default Page Size     *
               STACK=OFF,                         Initial Natural Cmds. *
               ET=OFF                             END/BACKOUT TRANSACT.
*-----------------------------------------------------------------------*
         NTDB2 BTIGN=ON,                          Ignore Trans. Error   *
               CONVERS=ON,                        Convers. Mode CICS    *
               CONVRS2=OFF,                       Convers. Mode2 CICS   *
               DB2PLAN=PQANDBvr,                  Plan Name             *
               DB2SSID=DB2A,                      Subsystem ID          *
               DB2XID=ON,                         Global Transaction ID *
               DDFSERV=CMFSERV,                   DD Name File Server   *
               DELIMID=OFF,                       Delimited Identifiers *
               MAXLOOP=10,                        Nested Program Loops  *
               MAXSTMT=10,                        Dynamic SQL Statements*
               NNPSF=OFF,                         Set Positive Sign     *
               NSBHOST=IBM2.HQ.SAG,               NSB Server Host Name  *
               NSBPORT=7311,                      NSB Server TCP/IP Port*
               PSCIGN=OFF,                        Positive SQLCODEs     *
               REFRESH=OFF,                       Refresh Setting       *
               RETRYPO=10,                        Positioning Retries   *
               RWRDONL=ON,                        Delimited Identifiers *
               STATDYN=NEVER                      Static Dynamic Switch
*-----------------------------------------------------------------------*
         NTOSP ABEXIT=ESTAE,                      Abend Processing      *
               LBPNAME=' ',                       Local Shared Buffer   *
               LEHDLR=ON,                         LE Error Handler      *
               SUBPOOL=0,                         Subpool for GETMAIN   *
               TIOBSZ=(8,64),                     Primary I/O Buffer    *
               USERID=OFF                         Init-User Job Name
*-----------------------------------------------------------------------*
         NTVSAM BTSUPP=ON,                        BACKOUT TRANSACTION   *
               CLSUPP=ON,                         Close Call at Session *
               DDMCHK=OFF,                        Support of DDM        *
               DDSWITE=0,                         Maximum Entries DLBLY *
               DFBE=10,                           Decoded Format Buffer *
               DFBN=100,                          Format Buffer Entries *
               ENADIS=OFF,                        Enable Disabled Files *
               ENAUNE=OFF,                        Enable Unenabled Files*
               ETSUPP=ON,                         END TRANSACTION       *
```

```
                FORMAT=ON,                        Record Formatting     *
                KEYLGH=126,                       Length of VSAM Keys   *
                OPSUPP=OFF,                       Dynamic Open Calls    *
                PATH=CHECK,                       Path Processing       *
                PSIGNF=OFF,                       Compiler Option PSIGNF*
                RETRY=(OFF,OFF),                  Retry ON ERROR Clause *
                RLS=OFF,                          Record-Level Sharing  *
                ROLLSIZ=550,                      Session Status Info.  *
                SFILE=ON,                         Support of VSAM Files *
                TAFE=10,                          Maximum No. DDMs      *
                TAFN=50,                          Maximum No. DDM Fields*
                TIMEOUT=0,                        Timeout RLS Request   *
                TSAE=20,                          READ/FIND Statement   *
                TVS=OFF,                          Support of DFSMSTVS   *
                UPDL=32768                        Size of Update Table
```

# III z/OS-Umgebung

Dieser Teil enthält Informationen über Natural unter dem Betriebssystem z/OS.

| | |
|---|---|
| **Natural unter z/OS** | Enthält einen Überblick über spezielle Aspekte, die zu berücksichtigen sind, wenn Natural unter z/OS im Online- oder im Batch-Modus ausgeführt wird. |
| **Authorized Services Manager** | Beschreibt die Funktionalität und den Betrieb des Authorized Services Manager (ASM), der unter z/OS verfügbar ist. |
| **Natural Roll Server-Funktionen** | Erläutert die Funktionen des Natural Roll Server im Allgemeinen, seine Verwendung in einem einzelnen z/OS-System und in einer z/OS-Parallel-Sysplex-Umgebung. |
| **Betrieb des Natural Roll Server** | Enthält Informationen zu den Systemanforderungen, dem Betrieb, der Leistungsoptimierung und der Neustartfähigkeit des Roll Server. |

**Anmerkung:** Die Codes, die Natural bei der Verwendung des Roll Servers zur Natural-Sitzungslaufzeit erhalten kann, werden von den entsprechenden Teleprocessing-Schnittstellen (*Natural under CICS* oder *Natural under IMS TM*) ausgegeben. Eine Liste dieser Codes finden Sie unter *Return Codes and Reason Codes of the Roll Server Request* in der *Messages and Codes*-Dokumentation.

# 12 Natural unter z/OS


- Natural-Subsystem ..... 92
- TP-Monitor-Schnittstellen ..... 92
- Datenbankmanagementsystem-Schnittstellen ..... 92
- Natural im Batch-Modus unter z/OS ..... 93
- Natural als Server unter z/OS ..... 93

Dieses Kapitel gibt eine Übersicht über spezielle Aspekte, die beim Einsatz von Natural unter z/OS zu berücksichtigen sind.

## Natural-Subsystem

Ein Natural-Subsystem unter z/OS umfasst folgende Komponenten:

- einen oder mehrere **globale Buffer Pools**,
- einen **Authorized Services Manager**,
- einen **Roll Server**.

Das Natural-Subsystem wird durch den Natural-Profilparameter `SUBSID` und durch entsprechende Startup-Parameter für die oben genannten Komponenten identifiziert. Der Standard-Subsystemname ist `NAT`*`v`*, wobei *`v`* die erste Ziffer der aktuellen Natural-Version ist.

Mittels der Natural-Subsystemtechnik können mehrere Roll Server gleichzeitig verwendet und mehrere unabhängige Sets globaler Buffer Pools erstellt werden - es können sogar mehrere Natural-Laufzeitumgebungen eingerichtet werden, die völlig unabhängig voneinander sind.

## TP-Monitor-Schnittstellen

Informationen zu den TP-Monitor-Schnittstellen, die mit Natural unter z/OS verfügbar sind, finden Sie in den folgenden Abschnitten *TP-Monitor-Schnittstellen*-Dokumentation:

- *Natural unter Com-plete*
- *Natural unter CICS*
- *Natural unter TSO*
- *Natural unter IMS TM*

## Datenbankmanagementsystem-Schnittstellen

Mit Ausnahme des Datenbankmanagementsystems Adabas werden alle Operationen, die eine Interaktion mit der Datenbank erfordern, von einem entsprechenden Natural-Schnittstellenmodul ausgeführt.

Informationen zu den DBMS-Schnittstellen, die mit Natural unter z/OS verfügbar sind, finden Sie in den folgenden Abschnitten der *Datenbankmanagementsystem-Schnittstellen*-Dokumentation:

- *Natural for Db2*

- *Natural for VSAM*

## Natural im Batch-Modus unter z/OS

Siehe ***Natural im Batch-Modus - Allgemeines*** und ***Natural im Batch-Modus unter z/OS***.

## Natural als Server unter z/OS

Natural ist nicht nur eine Programmiersprache, sondern kann auch als Server in einer Client/Server-Umgebung fungieren. Ausführliche Informationen finden Sie unter ***Natural als Server unter z/OS***.

# 13 Authorized Services Manager unter z/OS


- ASM-Übersicht ..... 96
- ASM-Systemanforderungen ..... 97
- ASM starten ..... 99
- ASM-Operator-Kommandos ..... 104
- Struktur der Coupling Facility für ASM zurücksetzen ..... 105
- ASM-Meldungen, Condition Codes und Abend Codes ..... 105

Dieses Kapitel beschreibt die Funktionalität und den Betrieb des Authorized Services Manager (ASM), der mit Natural unter z/OS verfügbar ist.

## ASM-Übersicht

Der Authorized Services Manager (ASM) bietet autorisierte Betriebssystemfunktionen für Natural. Zu diesen Funktionen gehören das Schreiben von SMF-Datensätzen und die z/OS Parallel Sysplex-Kommunikation über die Coupling Facility (CF). Der ASM stellt seine Funktionen über PC-Routinen bereit und läuft in seinem eigenen Adressraum.

Die folgenden autorisierten Funktionen werden bereitgestellt:

- Kommunikation von Natural-Buffer-Pool-Verwaltungsmeldungen,
- Schreibzugriff auf globale Buffer Pools im System-Key,
- Schreiben von SMF-Datensätzen,
- Speicherung von Natural-Sitzungsinformationen im Session Information Pool (SIP),
- Ausführung autorisierter Systemdienste zur Unterstützung von IBM zIIP (System z Integrated Information Processor),
- Ausführung autorisierter Systemdienste für z/OS Shared-Memory-Objekte,
- Ausführung autorisierter Dienste über die RACROUTE-Schnittstelle des z/OS Security Server (RACF oder ein anderes externes Security-Produkt).

Die ersten drei Funktionen sind immer verfügbar, während der Session Information Pool (SIP) benötigt und verwendet wird, wenn Natural mit aktiviertem SYSPLEX läuft. Weitere Informationen zu den Parametern, die Sie angeben müssen, um Session Information Records (SIRs) im SIP zu halten, finden Sie bei den Natural-Parametern `CICSPLX` und `SIPSERV` unter *Natural-CICS-Generierungsparameter* in der *TP-Monitor-Schnittstellen*-Dokumentation. Der SIP, der die SIRs in SIP-Slots enthält, kann mittels Startparameter verfügbar gemacht werden. Weitere Informationen zum Starten des ASM finden Sie unter ***ASM starten***.

In den folgenden Fällen müssen Sie den ASM verwenden:

- Der Natural-Profilparameter `BPPROP` ist auf `PLEX` oder `GLOBAL` oder `GPLEX` gesetzt (Buffer Pool Propagation wird verwendet).
- Globale Natural-Buffer Pools werden im System Key zugeordnet.

  Dies ist erforderlich, wenn Ihr Systemprogrammierer `VSM ALLOWUSERKEYCSA(NO)` in `SYS1.PARMLIB(DIAG`*xx*`)` angegeben hat. Siehe auch ***Zuordnung des Natural GBP*** im Abschnitt *Natural Global Buffer Pool unter z/OS*.
- Natural under CICS wird in einer z/OS Parallel Sysplex-Umgebung verwendet (SIP-Funktion erforderlich).

- Natural under IMS TM wird im terminalorientierten, nicht-konversationellen Modus verwendet (mit der SIP-Funktion).
- Natural under IMS TM wird verwendet, wobei die Accounting-Funktion SMF-Datensätze schreibt.
- Natural under IMS TM wird mit der Broadcasting-Funktion verwendet.
- Natural under IMS TM wird mit der Monitor-Funktion verwendet.
- Die Aktivierung der zIIP-Unterstützung ist erforderlich.
- Die Aktivierung des Shared Memory Objects File Server (FSSM) von Natural for Db2 ist erforderlich.
- Aktivierung des ACEE-Cachings (`SECURITY_CACHING=YES`) des Natural Development Server.

Der Session Information Pool (SIP) enthält die Natural-Sitzungsinformationsdatensätze. Im terminalorientierten, nicht konversationellen Modus benötigen die Natural CICS-Schnittstelle und die Natural IMS TM-Schnittstelle diese Datensätze, um eine Natural-Sitzung nach einer Terminal-E/A fortzusetzen. In einer z/OS-Parallel-Sysplex-Umgebung wird der SIP in der Coupling Facility (CF) erstellt, und ein Speicherobjekt wird als Zwischenpuffer verwendet, um unnötige Zugriffe auf die CF zu vermeiden. Andernfalls wird der SIP in einem Speicherobjekt erstellt. (Ein Speicherobjekt befindet sich in einem adressierbaren 64-Bit-Speicher oberhalb der 2-Gigabyte-Adresse).

Wenn der ASM in einer z/OS-Parallel-Sysplex-Umgebung verwendet wird, muss für jedes Natural-Subsystem in jedem beteiligten z/OS-Image eine ASM-Instanz gestartet werden.

**Hinweis zu Natural/CICS:**

Die CICS System Recovery Table sollte den z/OS-Systemabbruchcode `0D6` enthalten.

## ASM-Systemanforderungen

Dieser Abschnitt beschreibt die Systemerfordernisse für den ASM.


- APF-Berechtigung
- Systemverlinkungsindex
- CF-Struktur (ASM)

- XCF-Signalisierungspfade

### APF-Berechtigung

Verlinken Sie die Module `NATASMvr` (wobei *vr* für die jeweilige Produktversion steht) und `NATBPMGR` mit einer APF Library (Authorized Program Facility), indem Sie den `IEWL`-Parameter `AC(1)` angeben. Siehe *Natural auf z/OS installieren*.

### Systemverlinkungsindex

Da der ASM einen System Linkage Index (System LX) reserviert, müssen Sie prüfen, ob der Wert von `NSYSLX` im Member `IEASYSxx` der Library `SYS1.PARMLIB` hoch genug ist.

**Anmerkung:** Wenn Sie den ASM beenden, ist die Adressraum-ID nicht mehr verfügbar, weil ein System-LX verwendet wurde. Sie wird erst mit dem nächsten IPL wieder verfügbar.

### CF-Struktur (ASM)

Eine CF-Struktur wird verwendet, wenn Sie den SIP in einer z/OS Parallel Sysplex-Umgebung ausführen.

Die Größe einer CF-Struktur kann mit dem IBM-System z Coupling Facility Structure Sizer Tool (CFSizer Utility) berechnet werden, das Sie hier finden:

**https://www.ibm.com/support/pages/cfsizer**

Dort müssen Sie in der Auswahlbox, wo `Choose one` steht, `XCF` wählen.

Es wird ein neues Fenster angezeigt, das beispielsweise vorbelegt ist mit

Number of systems `8` und CLASSLEN `956`

(= 936 + 20 -> CONA + CONALOCKATTR).

Siehe Makro `IXLYCONA` bezüglich dieser Größen:

```
01 SIZE:
*            CONA          -- X'03A8' bytes = 936
*            CONALOCKATTR  -- X'0014' bytes =  20
*            CONALISTATTR  -- X'0028' bytes =  40
*            CONACACHEATTR -- X'001C' bytes =  28
```

Der Authorized Services Manager benötigt CONA + CONALISTATTR = 936 + 40 = 976 bytes.

Wenn Sie `Number of systems = n (LPARs)` auswählen und auf `Submit` drücken, erhalten Sie beispielsweise bei 5 LPARs mit 936 Bytes:

| Function | Type | Structure Name | INITSIZE | SIZE |
|---|---|---|---|---|
| XCF | LIST | IXC..... | 18M | 19M |

Danach folgt eine Reihe von Zeilen mit JCL.

### XCF-Signalisierungspfade

Die XCF Signaling Services werden zur Weiterleitung von Buffer Pool-Verwaltungsmeldungen in einer z/OS-Parallel-Sysplex-Umgebung verwendet. Die Mindestlänge einer Meldung beträgt 64 Byte, die Höchstlänge 2048 Byte. Wie oft Meldungen gesendet werden, hängt davon ab, wie oft Natural-Objekte behandelt werden (mit dem Systemkommando `CATALOG`, `STOW` oder DELETE).

## ASM starten

Sie können den ASM entweder als Batch-Job oder als gestartete Task starten, indem Sie das Modul `NATASM`*`vr`* ausführen, wobei *`vr`* für die entsprechende Produktversion steht. Sie können Parameter in dem JCL `EXEC`-Statement, in einer Parameterdatei oder in beiden angeben. Ein Parameter, der in dem `EXEC`-Statement angegeben wird, überschreibt den entsprechenden Parameter in der Parameterdatei.

Es wird empfohlen, eine Parameterdatei (siehe ***Parameter in der Parameterdatei***) zu verwenden, da die Parameter, die die SIP-Timeout-Verarbeitung steuern, sowie zukünftige Parameter nur in der Parameterdatei angegeben werden können. Vorhandene JCL wird weiterhin unverändert ausgeführt.

Folgende Themen werden behandelt:

- Parameter im JCL EXEC-Statement (ASM)
- Parameter in der Parameterdatei (ASM)

### Parameter im JCL EXEC-Statement (ASM)

Geben Sie im JCL `EXEC`-Statement als `PARM` die folgenden Parameter an:

```
subsystem-id,XCF-group-name,CF-structure-name,number-of-SIP-slots,SIP-slot-size,message-case,Update-ECSA-D
```

Alle Parameter sind positionsgebunden und müssen durch ein Komma voneinander abgetrennt werden. Sie werden in der folgenden Tabelle erläutert:

| Parameter | Mögliche Werte | Standardwert | Kommentar |
|---|---|---|---|
| *subsystem-id* | 4-Byte-Zeichenkette ohne Leerzeichen | NAT*v* | Der angegebene Wert muss mit dem Wert des Natural-Profilparameters SUBSID übereinstimmen (*v* steht für Version).<br><br>**Anmerkung:** Bei Natural under CICS siehe den Parameter CICSPLX im NCMDIR-Makro, um die entsprechende Subsystem-ID zu setzen. |
| *XCF-group-name* | beliebiger gültiger XCF-Gruppenname | keiner | Der Name der XCFGROUP für Signaling Services.<br><br>Ein Stern (*) ergibt den Namen NAT, gefolgt vom Namen des Subsystems. Der Name einer XCFGROUP sollte jedoch *immer* angegeben werden. |
| *CF-structure-name* | beliebiger gültiger CF-Strukturname | keiner | Optional, nur erforderlich, wenn SIP verwendet wird.<br><br>Der Name der CF-Struktur, die für die SIP-Funktion verwendet wird.<br><br>Wenn *XCF-group-name* angegeben wird, während *CF-structure-name* leer bleibt, können Sie nur die Buffer-Pool-Propagierung verwenden. |
| *number-of-SIP-slots* | 1 - 2147483647 | keiner | Optional, nur erforderlich, wenn SIP verwendet wird.<br><br>Die Anzahl der zuzuweisenden Slots, wenn die CF-Struktur noch nicht zugewiesen wurde. Wird sie weggelassen oder mit 0 angegeben, wird die gesamte Struktur für so viele Slots verwendet, wie sie aufnehmen kann. |
| *SIP-slot-size* | 256, 512, 1024, 2048 oder 4096 | 1024 | Der angegebene Wert wird ignoriert, wenn bereits eine CF-Struktur zugeordnet wurde. |
| *message-case* | UCTRAN oder leer | leer | Geben Sie UCTRAN an, wenn der Authorized Services Manager alle seine Meldungen in Großbuchstaben ausgeben soll. |
| Update-ECSA-D | ECSADUPD oder leer | leer | Aktualisieren eines älteren ECSA-Directory-Eintrags, ohne dass ein IPL erforderlich ist. |

### Parameter in der Parameterdatei (ASM)

Die Parameterdatei ist eine physische sequenzielle Datei (DSORG=PS), die mit LRECL=80 und RECFM=FB zugeordnet wird. In Ihrer JCL geben Sie diese Datei mit DDNAME ASMPARM an.

Die Parameter in der Parameterdatei werden als *name=value*-Paare angegeben. Geben Sie einen Parameter pro Zeile an, beginnend in Spalte 1. Das *name=value*-Paar wird durch das erste Leerzeichen abgeschlossen, und der Rest der Zeile wird nicht geprüft. Zeilen, die mit einem Stern (*) in Spalte 1 beginnen, werden als Kommentare behandelt. Parameter werden in Großbuchstaben umgesetzt, bevor sie verarbeitet werden.

Sie können auch einen Punkt als Füllzeichen verwenden, so dass die „=“-Zeichen in derselben Spalte ausgerichtet werden können. Dies ist jedoch *nicht* mit dem Parameter TIMEOUT möglich.

| Parameter | Mögliche Werte | Standardwert | Kommentar |
|---|---|---|---|
| SUBSID=*name* | 4-Byte-Zeichenkette ohne Leerzeichen | NAT*v* | Der angegebene Wert muss mit dem Wert des Natural-Profilparameters SUBSID übereinstimmen (*v* steht für Version).<br>**Anmerkung:** Bei Natural under CICS siehe den Parameter CICSPLX im NCMDIR-Makro, um die entsprechende Subsystem-ID zu setzen. |
| XCFGROUP=*group-name* | beliebiger gültiger XCF-Gruppenname | keiner | Der Name der XCFGROUP für Signaling Services.<br>Ein Stern (*) ergibt den Namen NAT, gefolgt vom Namen des Subsystems.<br>Der Name einer XCFGROUP sollte jedoch *immer* angegeben werden. |
| STRUCTURE=*structure-name* | beliebiger gültiger CF-Strukturname | keiner | Nur für die SIP-Funktion verwenden.<br>Der Name der CF-Struktur, die für die SIP-Funktion verwendet wird.<br>Wenn XCFGROUP angegeben wird, während STRUCTURE leer bleibt, können Sie nur die Buffer-Pool-Propagierung verwenden. |
| NUMSLOTS=*number* | 1 - 2147483647 | keiner | Nur für die SIP-Funktion verwenden.<br>Die Anzahl der zuzuordnenden Slots, wenn die CF-Struktur noch nicht zugeordnet wurde. |

| Parameter | Mögliche Werte | Standardwert | Kommentar |
|---|---|---|---|
| | | | Wenn sie weggelassen oder als 0 angegeben wird, wird die gesamte Struktur für so viele Slots verwendet, wie sie aufnehmen kann. |
| SLOTSIZE=*size* | 256, 512, 1024, 2048 oder 4096 | 1024 | Nur für die SIP-Funktion verwenden.<br><br>Der angegebene Wert wird ignoriert, wenn bereits eine CF-Struktur zugewiesen wurde. |
| MSGCASE=*case* | UPPER oder MIXED | MIXED | Geben Sie UPPER an, wenn der Authorized Services Manager alle seine Meldungen in Großbuchstaben ausgeben soll. |
| NONACTIVITY=*hours* | 1 - 999999 | keiner | Nur für die SIP-Funktion verwenden.<br><br>Die Anzahl der Stunden, die eine SIP-Sitzung inaktiv sein kann, bevor sie gelöscht wird.<br><br>Wird diese Zeit überschritten, wird die Sitzung bei der nächsten geplanten Zeitüberschreitungsprüfung gelöscht.<br><br>Wenn dieser Parameter weggelassen wird, wird keine Zeitüberschreitungsprüfung durchgeführt.<br><br>Dieser Parameter kann mit dem Operator-Kommando TIMEOUT geändert werden (siehe ***ASM-Operator-Kommandos***).<br><br>Sowohl der Authorized Services Manager als auch der Roll Server ermöglichen die Angabe eines Zeitüberschreitungswerts. Wenn Natural in einer SYSPLEX-Umgebung ausgeführt wird, müssen Sie für diesen Parameter und den Parameter non-activity-time (Inaktivitätszeit) des Roll Server denselben Wert festlegen. |
| TIMEOUT=*option* | VERBOSE oder TERSE | VERBOSE | Anzeige (verbose = mit Text) oder Unterdrückung (terse = kein Text) der Meldungen ASM0085 und ASM0086 während eines TIMEOUT-Vorgangs. |

**Anmerkung:** Wenn `TIMEOUTCHECK` angegeben ist und sein Wert vor dem Startzeitpunkt von `NATASMvr` liegt, wird der nächste `TIMEOUTCHECK`-Wert ausgehend vom angegebenen `TIMEOUTCHECK`-Wert berechnet, so als ob bereits ein oder mehrere `TIMEOUTCHECK`s ausgeführt worden wären. Dies ist jedoch erst ab `NATASM93` möglich.

**Example:**

**Beispiele:**

In den folgenden Beispielen steht *v* oder *vr* für die jeweilige ein- oder zweistellige Version des Produkts.

- `//ASM EXEC PGM=NATASMvr,PARM='NATv,NATXCF,CFSIP,1500,512'`

  Die Subsystemkennung (ID) ist `NATv`, die Nachrichtengruppe für die Buffer-Pool-Kommunikation ist `NATXCF`, die Struktur für den Session Information Pool ist `CFSIP`. Es sind 1500 SIP-Slots mit einer Größe von jeweils 512 Byte zu verwenden.

- `//ASM EXEC PGM=NATASMvr,PARM='NATv,NATXCF,CFSIP'`

  Wie oben, außer SIP-Slots:

  Der ASM verwendet so viele SIP-Slots, wie die CFSIP-Struktur aufnehmen kann, mit einer Größe von jeweils 1024 Bytes.

- `//ASM EXEC PGM=NATASMvr,PARM='NATv,NATXCF,,500,512'`

  Der SIP Service soll nicht die Coupling Facility nutzen, sondern 500 SIP-Slots mit einer Größe von jeweils 512 Byte im Speicher anlegen.

- `//ASM EXEC PGM=NATASMvr,PARM='NATv,NATXCF'`

  Der SIP-Service wird nicht verfügbar sein.

- ```
  //ASM EXEC PGM=NATASMvr,PARM='TST5'
  //ASMPARM  DD DISP=SHR,DSN=FB.SYSF.PARMS(ASMPARM1)
  ```

  Datei `FB.SYSF.PARMS(ASMPARM1)`:

  ```
  MSGCASE=M              Mixed case messages
  SUBSID=TST1
  XCFGROUP=HELGA         Message group for global buffer pool administration
  *SIP definitions:
  STRUCTURE=TSTSIP       SIP CF structure
  SLOTSIZE=256
  NUMSLOTS=200
  TIMEOUTCHECK=2135      Delete old sessions at 9:35 pm
  NONACTIVITY=2         Delete sessions that have been inactive for 2 hours or ↩
  more
  ```
  ```

Der SIP-Service soll 200 SIP-Slots in der CF-Struktur `TSTSIP` mit einer Größe von jeweils 256 Byte anlegen. Das zu verwendende Natural-Subsystem ist `TST5`, da der Parameter im `EXEC`-Statement das in der Parameterdatei angegebene Subsystem `TST1` überschreibt.

## ASM-Operator-Kommandos

Die folgenden Kommandos können mit dem `MODIFY`-Kommando an den ASM übergeben werden:

| Kommando | | Beschreibung |
|---|---|---|
| `HELP` | | Zeigt eine Übersicht über die verfügbare Syntax. |
| `TERMinate` \| `STOP` | | Beendet den ASM. |
| `SNAP` | | Debugging-Funktion.<br><br>Der ASM-Adressraum wird in `SYSUDUMP` gespeichert. |
| `VLIST` | | Zeigt den Namen, die Version und den Zeitpunkt der Assemblierung von Modulen an, die mit dem ASM verlinkt sind. |
| `TIMEOUT` | `NAT` *`nnn`* | Legt den Parameter für die Nichtaktivitätszeit fest oder ersetzt ihn. |
| | `REPEAT` *`mmmm`* | Legt das Zeitintervall in Minuten fest, in dem die Zeitüberschreitungsprüfung durchgeführt werden soll, oder ersetzt es. |
| | `TOC` *`hhmm`* | Gibt die Tageszeit der Zeitüberschreitungsprüfung an oder ersetzt diese. |
| | `OFF` | Deaktiviert die Zeitüberschreitungsprüfung. |
| | `ON` | Aktiviert die Zeitüberschreitungsprüfung wieder. |
| | `NOW` | Startet eine sofortige Zeitüberschreitungsprüfung.<br><br>Die normale Zeitüberschreitungsprüfung (falls angegeben) bleibt wirksam. |
| | `TERSE` | Unterdrückt die Meldungen `ASM0078` und `ASM0080` während der `TIMEOUT NOW`-Verarbeitung.<br><br>Die Meldung `ASM0047 Operator command: TIMEOUT NOW` wird ebenfalls unterdrückt. |
| | `VERBOSE` | Zeigt die Meldungen `ASM0078`, `ASM0080` und `ASM0047` während `TIMEOUT NOW`-Verarbeitung an.<br><br>Dies ist die Standardeinstellung. |
| | ?<br><br>(oder keine Angabe) | Zeigt die aktuellen Timeout-Einstellungen an.<br><br>Das Fragezeichen (?) ist optional und kann weggelassen werden. |

Eine Liste der Return Codes und Reason Codes des SIP Service finden Sie unter *SIP Service Return Codes and Reason Codes* in der *Messages and Codes*-Dokumentation.

## Struktur der Coupling Facility für ASM zurücksetzen

Wenn eine CICS- oder IMS TM-Region, die die Session Information Pool (SIP)-Funktion des Authorized Services Manager verwendet, abbricht, gibt der Authorized Services Manager gegebenenfalls einen Fehler für den SIP zurück. Beispielsweise könnte der SIP als voll gemeldet werden, weil die Sitzungsbereinigung vor dem Abbruch der Region nicht durchgeführt worden war. Um einen solchen Fehler zu beheben, müssen Sie die entsprechende Coupling Facility-Struktur löschen:

1. Schalten Sie alle Authorized Services Manager aus, die die betroffene CF-Struktur verwenden.
2. Setzen Sie folgendes Operator-Kommando ab:

   ```
   SETXCF FORCE,STR,STRNAME=structure-name
   ```

   wobei *structure-name* der Name der für die SIP-Funktion verwendeten CF-Struktur ist.
3. Starten Sie alle Authorized Services Manager neu.

## ASM-Meldungen, Condition Codes und Abend Codes

Der ASM schreibt Informations- und Fehlermeldungen in `JESMSGLG` unter Verwendung des `WTO`-Makros (`ROUTCDE=11`). Den Meldungen wird eine Meldungskennung und der ASM-Jobname vorangestellt, z. B.:

```
ASM0005 FBASMvr
```

In diesem Beispiel ist Authorized Services Manager Version *vr* (wobei *vr* für die entsprechende Produktversion steht) aktiv.

Es werden die folgenden Condition Codes verwendet:

| Condition Code | Erläuterung |
|---|---|
| 0 | Normale Beendigung |
| 4 | Authorized Server nicht verfügbar |
| 8 | Kein Platz für Arbeitsbereich |
| 12 | Falsche Parametereingabe |
| 16 | Laufzeitfehler ist aufgetreten |
| 20 | Subtask ist fehlgeschlagen |
| 24 | Abbruch ist aufgetreten |
| 97 | Buffer Pool-Routine nicht verlinkt |
| 98 | Ungültiger Buffer Pool-Typ |

| Condition Code | Erläuterung |
|---|---|
| 99 | Kein Speicher mehr verfügbar |
| >100 | Arbeitsspeicher konnte nicht zugeordnet werden. |

Es werden die folgenden Benutzer-Abbruch-Codes verwendet:

| Abend Code /Abbruchcode | Reason/Ursache | Kommentar |
|---|---|---|
| U0100 | IXCJOIN fehlgeschlagen. | Abend Register 14 enthält den Reason Code. |
| U0101 | IXCQUERY fehlgeschlagen. | Abend Register 14 enthält den Reason Code. |
| U0103 | Liste der aktiven Members ist voll. | Wenden Sie sich an den Support. |
| U0104 | IXCMSGI fehlgeschlagen. | Abend Register 14 enthält den Reason Code. |
| U0105 | Message Exit konnte keinen Purge Task Request Block erhalten. | Wenden Sie sich an den Support. |
| U0106 | Work Space for IXLCONN konnte nicht abgerufen werden. | Wenden Sie sich an den Support. |
| U05*xx* | IXLCONN fehlgeschlagen. | *xx* ist der Reason Code. |
| U06*xx* | IXLLIST WRITE fehlgeschlagen. | *xx* ist der Reason Code. |
| U070*x* | STORAGE OBTAIN - Speicherfreigabe für Subpool 245 fehlgeschlagen. | Wenden Sie sich an den Support. |
| U071*x* | STORAGE RELEASE - Speicherfreigabe für Subpool 245 fehlgeschlagen. | Wenden Sie sich an den Support. |

Eine Beschreibung der Reason Codes finden Sie unter *Programming: Sysplex Services Reference* (IBM documentation). Wenn der Fehler umgebungsspezifisch war und die Ursache unklar ist, wenden Sie sich an den Support.

# 14 Natural Roll Server-Funktionen


- Natural Roll Server - Überblick ..... 108
- Roll-Server in einem einzelnen z/OS-System ..... 109
- Roll Server in einer OS-Parallel-Sysplex-Umgebung ..... 110
- Roll File und LRB ..... 112

Siehe auch ***Betrieb des Natural Roll Server*** .

## Natural Roll Server - Überblick

Mit dem Natural Roll Server kann Natural in einem System mit mehreren Adressräumen wie CICS oder IMS TM ausgeführt werden. Diese Adressräume können sich in mehreren z/OS-Images befinden (z/OS Parallel Sysplex-Umgebung). Natürlich können Sie den Roll Server auch verwenden, wenn Sie ein einzelnes z/OS-System betreiben.

Wenn Natural eine Terminal-Ein-/Ausgabe durchführt, muss es die Kontextdaten der Anwendung (den Thread) speichern: Bevor die Terminal-E/A gestartet wird, wird der Thread an den Roll Server übergeben, der ihn in seinem Local Roll Buffer oder im Roll File (Auslagerungsdatei) speichert. Dieser Vorgang wird als Roll-out (Auslagerung) bezeichnet. Wenn die Terminal-E/A abgeschlossen ist, fordert Natural den Thread vom Roll Server an und setzt die Anwendung fort. Dies wird als Roll-ln (Einlagerung) bezeichnet. In einer z/OS Parallel Sysplex-Umgebung speichert der Roll Server Informationen über die Threads (das Roll File Directory) in einer Datenstruktur in der Coupling Facility. So ist es möglich, dass eine Natural-Anwendung auf verschiedenen z/OS-Systemen zu unterschiedlichen Zeiten ausgeführt wird: Ein Thread kann auf einem System an den Roll Server übergeben und von einem anderen System zurück angefordert werden.

Vor dem Roll-out komprimiert Natural den Thread in einen zusammenhängenden Puffer und dekomprimiert ihn nach dem Roll-in. Je nach der bei Ihnen installierten Natural-Version kann die CPU-Last der Komprimierung und Dekomprimierung vom Hosting-TP-System weggenommen und in entsprechende Routinen innerhalb des Roll Servers verlagert werden. Wenn Sie die Komprimierungs-/Dekomprimierungsfunktion nutzen möchten, müssen Sie das entsprechende Roll Server-Modul installieren, das im entsprechenden Installationsschritt im Abschnitt *Natural auf z/OS installieren* in der *Installation for Natural on z/OS*-Dokumentation beschrieben ist. Außerdem muss Natural Batch for zIIP (zusätzliche Produktlizenz erforderlich) für den Natural-Nukleus verfügbar sein (siehe den entsprechenden Installationsschritt im Abschnitt *Natural auf z/OS installieren*).

Der Roll Server läuft in seinem eigenen Adressraum. Er stellt seine Dienste als PC-Routinen bereit. In einer z/OS-Parallel-Sysplex-Umgebung muss in jedem beteiligten z/OS-Image eine Instanz des Roll Server gestartet werden.

Eine Liste der verwendeten Roll Server Zaps wird im Dataset `JESMSGLG` der gestarteten Roll Server Task angezeigt.

Sie können auch die Funktion **Natural Subsystems and Roll Server Information** des Dienstprogramms SYSTP benutzen: > Funktionscode `R` > **Natural Subsystems / Overview** > Zeilenkommando `Z` > Bildschirm **Zaps Applied to Roll Server**.

**Hinweis zu Natural unter CICS:** Die CICS System Recovery Table sollte den z/OS-Systemabbruchcode `0D6` enthalten.

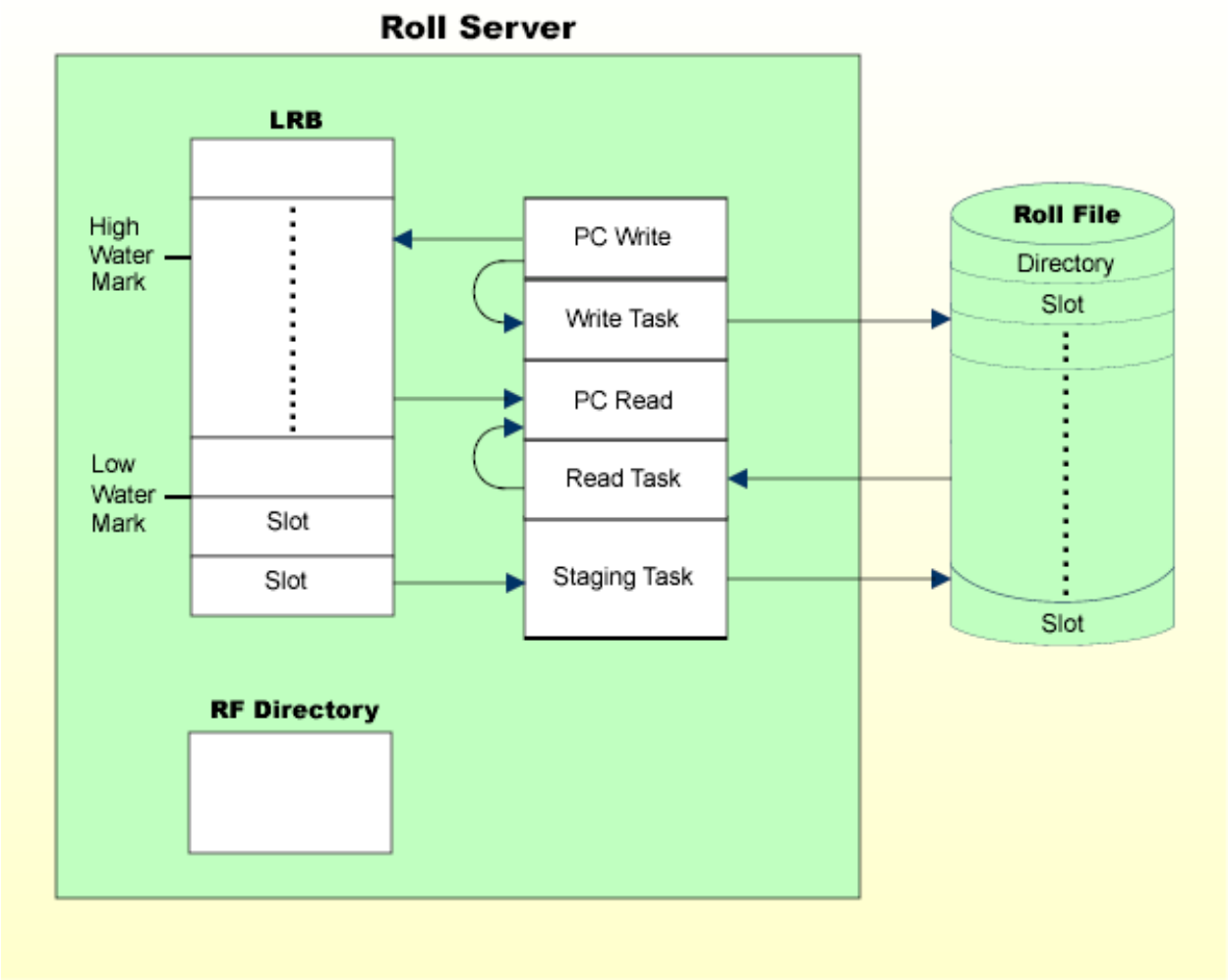
## Roll-Server in einem einzelnen z/OS-System

Wenn der Roll-Server einen Thread durch eine Schreibanforderung erhält (vor der Terminalausgabe), prüft er, ob im lokalen Roll Buffer (LRB) genügend Platz vorhanden ist. Ist dies der Fall, wird der Thread in den LRB kopiert. Wenn nicht, wird der Thread ins Roll File (RF) geschrieben. Ist der Thread größer als die Slot-Größe des Roll File, werden zusätzliche Überlauf-Slots zugewiesen, um den Thread unterzubringen. Die Zuweisung von Überlauf-Slots ist auf das Roll File beschränkt, dem die Natural Session ursprünglich zugewiesen wurde. Wenn das Roll File nicht über genügend freien Speicherplatz verfügt, um die erforderlichen Überlauf-Slots zuzuweisen, wird ein Fehler generiert und die anfordernde Natural-Sitzung abnormal beendet. Überlauf-Slots werden implizit durch eine nachfolgende Schreibanforderung mit einem kleineren Thread freigegeben.

Wenn der Roll Server eine Leseanforderung für den Thread erhält (nach der Terminaleingabe), versucht er, den Thread im LRB zu finden. Wird der Thread gefunden, wird er aus dem LRB in den Adressraum des Anforderers kopiert. Ist dies nicht der Fall, wird der Thread aus dem Roll File gelesen und in den Adressraum des Anforderers kopiert.

Um sicherzustellen, dass das System gut funktioniert und immer genügend Platz im LRB vorhanden ist, gibt es Zuordnungsmarken („Water Marks“). Wenn die höchste Zuordnungsmarke („High Water Mark“) des LRB erreicht ist, wird die Einstufungs-Task („Staging Task“) aktiviert. Sie kopiert den LRB-Inhalt in das Roll File, bis die niedrigste Zuordnungsmarke („Low Water Mark“) erreicht ist. Wo die höchste und die niedrigste Zuordnungsmarke platziert werden, ist daher eine wichtige Frage hinsichtlich der Leistungsoptimierung. Weitere Informationen zur Leistungsoptimierung finden Sie im Abschnitt ***Roll Server-Leistung optimieren***. Für einen Roll Server, der in einem einzelnen z/OS-System läuft, liegt die höchste Zuordnungsmarke standardmäßig bei 80 Prozent und die niedrigste Zuordnungsmarke bei 70 Prozent.

**Roll Server in einem einzelnen z/OS-System:**




## Roll Server in einer OS-Parallel-Sysplex-Umgebung

In einer z/OS-Parallel-Sysplex-Umgebung kommunizieren die Roll Server in den beteiligten z/OS-Images über die XCF-Signaling Services der Coupling Facility (CF), und das Roll File Directory befindet sich in einer XES-Datenstruktur.

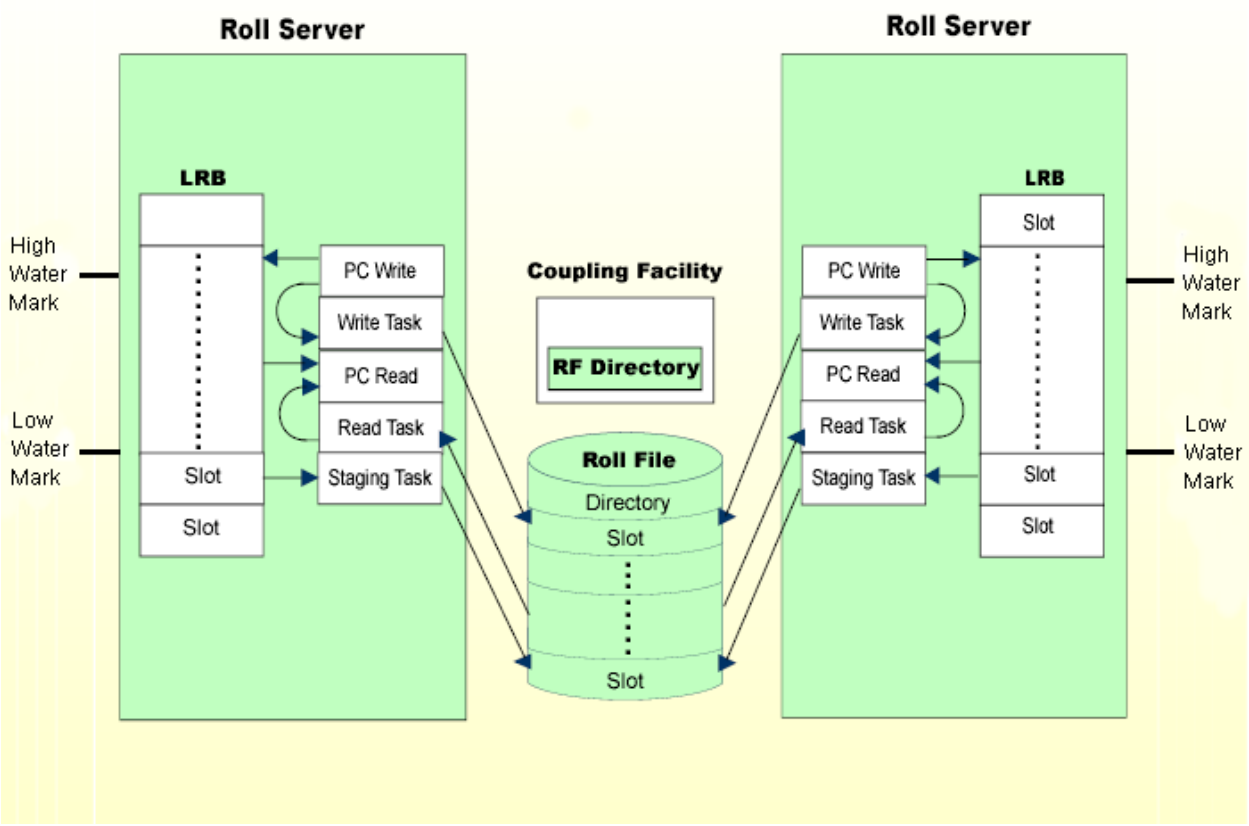
Wenn der Roll-Server einen Thread über eine Schreibanforderung erhält (vor der Terminalausgabe), prüft er, ob im lokalen Roll Buffer (LRB) genügend Platz vorhanden ist. Wenn genügend Platz vorhanden ist, wird der Thread in den LRB kopiert. Reicht der Platz im LRB nicht aus, wird der Thread direkt in das Roll File geschrieben. Das Verzeichnis des Roll Files in der CF-Struktur wird entsprechend aktualisiert. Der Überlauf von Threads wird wie unter *Roll-Server in einem einzelnen z/OS-System* beschrieben behandelt.

Sie können auch höchste und niedrigste Zuordnungsmarken (High und Low Water Marks) in einer parallelen Sysplex-Umgebung setzen. Diese Option ist in älteren Natural-Versionen nicht verfügbar. Der Einstufungs-Task („Staging Task“) schreibt Threads nur dann auf die Platte, wenn die höchste Zuordnungsmarken im LRB erreicht ist, bis die niedrigste Zuordnungsmarke erreicht ist. Wenn ein Thread von einem anderen z/OS-Image angefordert wird und noch nicht auf die Platte geschrieben wurde, sendet der Roll Server auf dem anderen z/OS-Image eine Einstufungsanforderungsmeldung (Stage-Request Message) für diesen bestimmten Thread. Der angeforderte Thread wird dann unabhängig von der höchsten und der niedrigsten Zuordnungsmarke auf die Platte geschrieben.

**Anmerkung:** Wenn Sie eine höchste Zuordnungsmarke von Null angeben, verhält sich der Roll Server identisch zu früheren Versionen von Natural, da alle Threads sofort für die Einstufung auf der Platte eingeplant werden. Bei einem Roll Server, der in einer parallelen Sysplex-Umgebung läuft, sind sowohl die höchste als auch die niedrigste Wassermarke standardmäßig Null.

Wenn der Roll Server eine Leseanforderung für einen Thread erhält (nach einer Terminaleingabe) und die letzte Schreibanforderung im selben z/OS-Image ausgegeben wurde, kopiert der Roll Server den Thread direkt vom LRB in den Adressraum des Anforderers. Wenn die letzte Schreibanforderung nicht aus demselben z/OS-Image stammt, wird der Thread aus dem Roll File gelesen und dann in den Adressraum des Anforderers kopiert. Befindet sich der Thread noch nicht im Roll File, erhält der Roll Server auf dem anderen z/OS-Image eine Stage Request-Meldung. Wenn sich der Thread im Roll File befindet, wird er von dort gelesen und anschließend in den Adressraum des Anforderers kopiert.

**Roll Servers in einer z/OS Parallel Sysplex Umgebung:**




## Roll File und LRB

Das Roll File ist eine BDAM-Datei, die logisch in ein Verzeichnis und in Slots mit fester Länge unterteilt ist. Die Slot-Größe ist ein Parameter der Roll File-Formatierungsroutine `NATRSRFI`. Es sollten mindestens so viele Slots vorhanden sein, wie es aktive Natural-Sitzungen gibt. Die Slot-Größe sollte so groß sein, dass ein durchschnittlicher komprimierter Natural Thread darin Platz findet. Threads, die größer als die Slot-Größe sind, belegen mehrere Slots.

Die Verteilung der Natural-Thread-Größen können Sie in der Funktion **Natural Subsystems and Roll Server Information** des Dienstprogramms SYSTP überprüfen:

> Funktionscode `R`, Bildschirm **Roll Server for Sub-System / Overview** > Zeilenkommando `R` > im Bildschirm **Roll Server for Sub-System ... General Information** blättern mit `PF8` bis zur Seite **Roll Server for Sub-System ... Peak Loads and Thread Sizes**.

Das Verzeichnis der Roll-Datei (Roll File Directory) enthält einen Eintrag für jede aktive Natural-Sitzung, zusammen mit einem Zeitstempel der letzten Schreibanforderung. In einem einzelnen z/OS-System befindet sich das Verzeichnis im Adressraum des Roll Server. In einer z/OS-Parallel-

Sysplex-Umgebung befindet es sich in der Coupling Facility. Das Verzeichnis wird nur dann in das Roll File zurückgeschrieben, wenn der Roll Server beendet wird.

Der lokale Roll Buffer (LRB) fungiert als Cache für das Roll File. Der Roll Buffer ist in einem z/OS-Speicherobjekt enthalten und in Slots mit fester Länge unterteilt. Die Slot-Größe des LRB ist identisch mit der Slot-Größe des entsprechenden Roll File. Wenn der Roll Server ohne Roll File laufen soll, muss die LRB-Slot-Größe als Parameter in dem JCL `EXEC`-Statement angegeben werden. Siehe hierzu ***Betrieb des Natural Roll Server***, ***Natural Roll Server starten***.

Der Roll Server kann mit bis zu fünf verschiedenen Roll Files betrieben werden. Jedes dieser Roll Files ist logisch mit einem lokalen Roll Buffer verbunden. Wenn es fünf Roll Files gibt, gibt es auch fünf entsprechende LRBs. Der Zugriff auf jedes Roll File erfolgt über eigene Lese-, Schreib- und Einstufungs-Tasks (Read, Write, Staging Tasks). Wenn die Roll Files auf verschiedenen Platten in unterschiedlichen Kanälen angelegt werden, kann also gleichzeitig auf die Roll Files zugegriffen werden.

Natural-Benutzer werden dem Roll File zugeordnet, das die meisten freien Slots hat. Um Ihre eigene Zuordnungsmethode zu implementieren, können Sie den User Exit `NATRSU14` verwenden. Weitere Informationen zu diesem User Exit finden Sie im entsprechenden Abschnitt im Kapitel ***Betrieb des Natural Roll Server***.

Wie Ihre Benutzerkennungen in den Roll Files verteilt sind, können Sie mit der Statistikfunktion **Natural Sub-Systems and Roll Server Information** (Funktionscode `R`) des Dienstprogramms SYSTP sehen.

# 15 Betrieb des Natural Roll Server


- Roll Server-Systemanforderungen ..... 116
- Roll File formatieren ..... 118
- Natural Roll Server starten ..... 122
- Roll Server-Meldungen, Condition Codes und Abend Codes ..... 130
- Return Codes und Reason Codes der Roll Server-Anforderung ..... 131
- Roll Server-Betriebsfunktionen ..... 131
- Struktur der Coupling Facility zurücksetzen ..... 133
- Roll Server-Leistung optimieren ..... 133
- Roll Server User Exits ..... 135

Siehe auch ***Natural Roll Server-Funktionen***.

## Roll Server-Systemanforderungen

In diesem Abschnitt werden die Systemerfordernisse für den Roll-Server beschrieben.

- APF-Autorisierung
- System Linkage Index
- Virtueller Speicher
- CF-Struktur (Roll Server)
- XCF-Signalisierungspfade

### APF-Autorisierung

Verlinken Sie das Modul `NATRSMvr` (*vr* steht für die entsprechende Produktversion) mit einer APF-Library, indem Sie den `IEWL`-Parameter `AC(1)` angeben. Siehe *Natural auf z/OS installieren*.

### System Linkage Index

Da der Roll Server einen System Linkage Index (System LX) reserviert, müssen Sie überprüfen, ob der Wert von `NSYSLX` im Member `IEASYSxx` der Library `SYS1.PARMLIB` hoch genug ist.

Wenn der Roll Server beendet wird, ist seine Adressraumkennung (ID) nicht mehr verfügbar, da ein System LX verwendet wurde. Die Kennung wird mit dem nächsten IPL wieder verfügbar.

Sobald ein System LX reserviert wurde, wird er bei jedem Neustart des Roll Server bis zum nächsten IPL wieder verwendet.

### Virtueller Speicher

| Speicher | Größe |
|---|---|
| ECSA | 84 Bytes |
| Privater Programmspeicher | 30 KB oberhalb * |
| Fester Subpool-Speicher (inkl. ELSQA): | 10 KB unterhalb, 50 KB oberhalb * |
| LRB directory | 32+(64*Anzahl der LRB Slots) |
| 100 Slots pro Roll File | 4 KB oberhalb * |
| Jedes weitere Roll File | 30 KB oberhalb * |
| Arbeitsspeicher | je nach Auslastung, ca. 1 MB oberhalb * |

* unterhalb bzw. oberhalb der Grenze

### CF-Struktur (Roll Server)

Eine CF-Struktur wird verwendet, um das Roll File Directory in einer z/OS-Parallel-Sysplex-Umgebung zu führen.

Die Größe einer CF-Struktur kann mit dem IBM-System z Coupling Facility Structure Sizer Tool (CFSizer Utility) berechnet werden, das Sie hier finden:

**https://www.ibm.com/support/pages/cfsizer**

Sie müssen in der Auswahlbox, dort wo `Choose one` steht, `XCF` wählen.

Es wird ein neues Fenster angezeigt, das beispielsweise vorbelegt ist mit

Number of systems `8` und CLASSLEN `956`

(= 936 + 20 -> CONA + CONALOCKATTR).

Siehe Makro `IXLYCONA` bezüglich dieser Größen:

```
01 SIZE:
*            CONA          -- X'03A8' bytes = 936
*            CONALOCKATTR  -- X'0014' bytes =  20
*            CONALISTATTR  -- X'0028' bytes =  40
*            CONACACHEATTR -- X'001C' bytes =  28
```

Der Roll Server verwendet nur `CONA = 936 bytes`.

Wenn Sie `Number of systems = n (LPARs)` auswählen und auf `Submit` drücken, erhalten Sie beispielsweise bei 5 LPARs mit 936 Bytes:

| Function | Type | Structure Name | INITSIZE | SIZE |
|---|---|---|---|---|
| XCF | LIST | IXC..... | 18M | 19M |

Danach folgt eine Reihe von Zeilen mit JCL.

### XCF-Signalisierungspfade

In einer z/OS-Parallel-Sysplex-Umgebung kommunizieren die Roll Server über die XCF Signaling Services. Als Standard-XCF-Gruppenname werden die ganz linken acht Bytes des CF-Strukturnamens verwendet.

Wenn Sie einen eigenen XCF-Gruppennamen angeben möchten, sollten Sie den User Exit `NATRSU24` verwenden. Weitere Informationen zu diesem User Exit finden Sie unter ***User Exit NATRSU24***.

# Roll File formatieren

Um das Roll File zu formatieren, gehen Sie wie folgt vor:

1. Legen Sie es als physischen, sequenziellen Dataset mit einem festen Dataset-Format an.
2. Formatieren Sie es unter Verwendung des Moduls `NATRSRFI`.
3. Wenn der Roll Server in einer z/OS-Parallel-Sysplex-Umgebung ausgeführt wird, setzen Sie die Struktur der Coupling Facility zurück, wie in ***Hinweise zum Formatieren oder Zurücksetzen von Roll Files*** beschrieben.

Bei der Formatierung wird das Role File in das BDAM-Format mit einer geräteabhängigen Blockgröße konvertiert.

**Anmerkung:** Wenn Sie ein bestehendes Roll File einer früheren Version verwenden wollen, genügt es, die Funktion `NATRSRFI RESET` auszuführen.

Zum Formatieren geben Sie die folgende Parameterfolge unter dem DD-Namen `RFIPARMS` oder als `PARM` im JCL `EXEC`-Statement ein:

```
function,dd-name,slot-size,number-of-slots
```

Alle Parameter sind positionsgebunden; sie werden in der nachstehenden Tabelle erläutert:

| Parameter | Beschreibung | |
|---|---|---|
| *function* | `FORMAT` | Formatiert das Roll File. |
| | `RESET` | Alle Roll File-Slots werden zurückgesetzt (als frei markiert). Sie können diesen Parameterwert nur verwenden, wenn das Role File bereits formatiert wurde.<br><br>Der einzige andere zulässige Parameter ist *dd-name*. |
| | `LIST` | Druckt eine Liste der in dem Roll File enthaltenen Session-IDs und ihrer letzten Aktivität.<br><br>Der einzige andere zulässige Parameter ist *dd-name*. |
| *dd-name* | Der Name des `DD`-Statement, unter dem das Roll File angegeben wurde. | |
| *slot-size* | Die Größe eines Roll File-Slots in Bytes. Diese Größe wird auf das nächsthöhere Vielfache der verwendeten Blockgröße gerundet.<br><br>Es wird empfohlen, zunächst eine Slot-Größe zu verwenden, die der Größe des Natural Threads entspricht. Schauen Sie sich dann die Statistiken des Roll Servers an. Sie zeigen | |

| Parameter | Beschreibung |
|---|---|
| | auch die größte Ausprägung einer Thread-Größe an. Verwenden Sie diesen Wert, um die Slot-Größe ggf. zu verringern. |
| *number-of-slots* | Die Anzahl der zuzuordnenden Roll File Slots. Diese Zahl ist die maximale Anzahl der gleichzeitig aktiven Benutzer.<br><br>Dieser Parameter ist optional. Wird er weggelassen, wird das gesamte Roll File so wie zugeordnet formatiert.<br><br>Beachten Sie, dass während der Formatierung der Systemabbruchcode SB37 oder SD37 (oder S209 für eine VSAM-Datei) auftreten kann. Dieser Abbruch wird von der Formatierungsroutine abgefangen und kann ignoriert werden. |

Zur Berechnung des erforderlichen Plattenplatzes in Zylindern für ein Roll File (Parameter `SPACE` des DD-Statements) können Sie die folgende Formel verwenden:

```
number-of-cylinders = ceiling (number-of-slots * slot-size / (30*block-size))
```

Anzahl der Zylinder = Obergrenze (Anzahl der Slots * Slot-Größe / (30*Blockgröße))

oder in Spuren

```
number-of-tracks = ceiling (number-of-slots * slot-size / (2*block-size))
```

Anzahl der Spuren = Obergrenze (Anzahl der Slots * Slot-Größe / (2*Blockgröße))

Die verwendete Blockgröße ist:

23476 für 3380 DASD

27998 für 3390 DASD

22928 für 9345 DASD

Darüber hinaus wird Platz für den Kopf des Roll File-Verzeichnisses (40 Byte) und einen Verzeichniseintrag für jeden Roll File-Slot (24 Byte) benötigt. Somit wird ein zusätzlicher Block für etwa 976 Slots auf 3380, 1164 Slots auf 3390 oder 953 Slots auf 9345 DASD benötigt.

### NATRSRFI-Ausgabe

Wenn ein DD-Statement mit *ddname* RFIPRINT angegeben ist, leitet NATRSRFI seine Ausgabe an diesen Dataset. Wird RFIPRINT weggelassen, so wird die Ausgabe unter Verwendung des WTO-Makros (ROUTDCE=11) in JESMSGLG geschrieben. Beachten Sie, dass RFIPRINT für die Funktion LIST angegeben werden muss.

**NATRSRFI Condition Codes und Abend Codes:**

Die folgenden Condition Codes werden verwendet:

| Code | Erläuterung |
|---|---|
| 0 | Normale Beendigung. |
| 4 | Anzahl der formatierten Slots ist geringer als angefordert. |
| 20 | Parameterfehler. |

Die folgenden User Abend Codes werden verwendet:

| Abend Code | Ursache |
|---|---|
| U0100 | Öffnen bei RFIPARMS oder RFIPRINT fehlgeschlagen. |
| U0101 | Öffnen bei Roll. File fehlgeschlagen. |

**Beispiele:**

In den folgenden Beispielen steht *vr* oder *vrs* für die jeweilige Produktversion.

Beispiel 1:

```
//FBRUNRFI JOB  (FB,218),FB,CLASS=K,MSGCLASS=X,NOTIFY=FB
//FORMAT   EXEC PGM=NATRSRFI
//STEPLIB  DD  DISP=SHR,DSN=NATURAL.NATvr.LOAD
//RF1      DD  DISP=SHR,DSN=FB.SYSF.ROLLF1
//RF2      DD  DISP=SHR,DSN=FB.SYSF.ROLLF2
//RFIPARMS DD  *
FORMAT,RF1,200000,1000
FORMAT,RF2,200000
```

Auszug aus dem daraus resultierenden JESMSGLG:

```
+FBRUNRFI: FORMAT,RF1,200000,1000
+FBRUNRFI: RF1: FB.SYSF.ROLLF1
+FBRUNRFI:      Creation date: 2001/06/13 Volume: ADA002(3390)
IEC031I D37-04,IFG0554P,FBRUNRFI,FORMAT,RF1,305B,ADA002,FB.SYSF.ROLLF1
+FBRUNRFI: Not enough space for 1000 slots.
+FBRUNRFI:       60 Blocks written. Block size is 27998.
+FBRUNRFI:        1 Directory block.
+FBRUNRFI:        8 Blocks per slot. Slot size is 223984.
+FBRUNRFI:        7 Slots initialized. Roll file version vrs.
+FBRUNRFI:        3 Blocks unused.
+FBRUNRFI: FORMAT,RF2,200000
+FBRUNRFI: RF2: FB.SYSF.ROLLF2
+FBRUNRFI:      Creation date: 2001/06/08 Volume: USRF08(3380)
IEC031I D37-04,IFG0554P,FBRUNRFI,FORMAT,RF2,020F,USRF08,FB.SYSF.ROLLF2
+FBRUNRFI:       60 Blocks written. Block size is 23476.
+FBRUNRFI:        1 Directory block.
+FBRUNRFI:        9 Blocks per slot. Slot size is 211284.
+FBRUNRFI:        6 Slots initialized. Roll file version vrs.
+FBRUNRFI:        5 Blocks unused.
```

Beispiel 2:

```
//FBRUNRFI JOB  (FB,218),FB,CLASS=K,MSGCLASS=X,NOTIFY=FB
//FORMAT   EXEC PGM=NATRSRFI,PARM='FORMAT,RF1,200000'
//STEPLIB  DD  DISP=SHR,DSN=NATURAL.NATvr.LOAD
//RF1      DD  DISP=SHR,DSN=FB.SYSF.ROLLF1
//RFIPRINT DD  SYSOUT=X
```

Resultierendes `RFIPRINT`:

```
Natural Roll Server - Roll File Utility Version
vrs
FORMAT,RF1,200000
RF1: FB.SYSF.ROLLF1
Creation date: YYYY/MM/DD Volume: ADA002(3390)
60 Blocks written. Block size is 27998.
1 Directory block.
8 Blocks per slot. Slot size is 223984.
7 Slots initialized. Roll file version vrs.
3 Blocks unused.
```

### Hinweise zum Formatieren oder Zurücksetzen von Roll Files

- Sie können mehrere Roll Files auf einmal formatieren oder zurücksetzen, indem Sie mehrere Parameterzeilen in `RFIPARMS` angeben.
- Sie können ein Roll File nicht formatieren oder zurücksetzen, während der Roll Server aktiv ist.
- Wenn das Roll File in einer z/OS-Parallel-Sysplex-Umgebung formatiert wird, muss auch die Struktur der Roll Server Coupling Facility mit dem Operator-Kommando `SETXCF` gelöscht werden, z. B.:

```
SETXCF FORCE,STR,STRNAME=NATROLL1
```

## Natural Roll Server starten

Sie können den Roll Server entweder als Batch-Job starten oder als Started Task, indem Sie das Modul `NATRSM`*`vr`* ausführen (wobei *`vr`* für die jeweilige Produktversion steht). Das oder die Roll Files müssen als `DD`-Statements mit *`ddname`* `ROLLF1` bis `ROLLF5` definiert sein.

Sie können Parameter im JCL EXEC-Statement, in einer Parameterdatei oder in beiden angeben. Ein Parameter, der in der EXEC-Anweisung angegeben wird, überschreibt den entsprechenden Parameter in der Parameterdatei.

Wir empfehlen die Verwendung einer Parameterdatei. Der Parameter `TIMEOUTREPEAT` (siehe ***Parameter in der Parameterdatei***) und zukünftige Parameter können nur in der Parameterdatei angegeben werden. Bestehende JCL wird unverändert weiter ausgeführt.

In diesem Abschnitt werden die folgenden Themen behandelt:

- Parameter im JCL EXEC-Statement
- Parameter in der Parameterdatei
- Beispiele für das Starten des Roll Servers als Batch Job

### Parameter im JCL EXEC-Statement

**Anmerkung:** Der lokale Roll Buffer befindet sich in einem Speicherobjekt „oberhalb der Grenze“. Verwenden Sie den `MEMLIMIT`-Parameter im `EXEC`-Statement, um sicherzustellen, dass genügend Speicher „oberhalb der Grenze“ zugewiesen werden kann.

Geben Sie in dem JCL EXEC-Statement als `PARM` die folgenden Parameter an:

*subsystem-id,number-of-roll-files,number-of-LRB-slots,LRB-slot-size,CF-structure-name,low-water-mark,high-water-mark,non-activity-time,timeout-check-time,message-case*

Subsystemkennung, Anzahl der Roll Files, Anzahl der LRB-Slots, LRB-Slot-Größe, CF-Strukturname, niedrigste Zuordnungsmarke, höchste Zuordnungsmarke, Inaktivitätszeit, Zeitüberschreitungsprüfzeit, Groß-/Kleinschreibung in Meldungen

Alle Parameter sind positionsgebunden und müssen durch ein Komma voneinander abgetrennt werden. Sie werden in der nachstehenden Tabelle erläutert:

| Parameter | Parametername für Parameter-Dataset | Mögliche Werte | Standardwert | Kommentar |
|---|---|---|---|---|
| *subsystem-id* | SUBSID | 4-Byte-Zeichenkette ohne Leerzeichen | NAT*v* | Der angegebene Wert muss mit dem Wert des Natural-Profilparameters SUBSID (*v* = Versionsnummer) übereinstimmen.<br>**Anmerkung:** Bei Natural unter CICS ist der Parameter ROLLSRV im Makro NCMDIR zur Einstellung der entsprechenden Subsystemkennung zu beachten. |
| *number-of-roll-files* | NUMFILES | 0 - 5 | 1 | In einer Nicht-Parallel-Sysplex-Umgebung unter z/OS kann der Roll Server ohne Roll File arbeiten und nur den speicherinternen Local Roll Buffer verwenden. |
| *number-of-LRB-slots* | NUMSLOTS | 1 - 32767 | keiner | Die Anzahl der LRB-Slots multipliziert mit der Slot-Größe darf 2 TB nicht überschreiten.<br>Für jeden LRB, d. h. für jedes verwendete Roll File, wird die gleiche Anzahl an LRB-Slots zugewiesen. Die Gesamtzahl der LRB-Slots wird nach folgender Formel berechnet:<br>*number-of-roll-files * number-of-LRB-slots* |
| *LRB-slot-size* | SLOTSIZE | 0..67108864 | Roll File Slot-Größe | Wert in Anzahl von Bytes.<br>Dieser Parameter muss angegeben werden, wenn kein Roll File verwendet wird. |

| Parameter | Parametername für Parameter-Dataset | Mögliche Werte | Standardwert | Kommentar |
|---|---|---|---|---|
| | | | | Wenn Roll Files verwendet werden, wird dieser Parameter ignoriert und stattdessen die Slot-Größe der Roll Files verwendet.<br><br>Wenn keine Roll Files verwendet werden, d. h. die Anzahl der Roll Files (*number-of-roll-files*) gleich Null ist, ist dieser Parameter obligatorisch. |
| *CF-structure-name* | STRUCTURE | beliebiger gültiger Strukturname | keiner | Wenn Sie weniger als 16 Zeichen angeben, werden Leerzeichen angefügt.<br><br>Geben Sie diesen Parameter nur an, wenn Sie die Coupling Facility (mit z/OS Parallel Sysplex) verwenden. Andernfalls kann die Verwendung dieses Parameters unnötigen Overhead verursachen. |
| *low-water-mark* | LOWWATER | 0 - 9 | 7 (einzelnes z/OS)<br><br>0 (Sysplex) | Gibt die niedrigste Zuordnungsmarke in Schritten von 10 Prozent der Anzahl der LRB-Slots an.<br><br>Der *low-water-mark*-Wert kann nicht höher sein als der aktuelle *high-water-mark*-Wert.<br><br>Dieser Wert wird immer ignoriert und auf 0 zurückgesetzt, wenn *CF-structure-name* einen Wert hat, unabhängig davon, was angegeben wurde. |
| *high-water-mark* | HIGHWATER | 0 - 10 | 8 (einzelnes z/OS)<br><br>0 (Sysplex) | Analog zum Parameter *low-water-mark*.<br><br>Der Wert 10 bedeutet, dass die Einstufungs-Task (Staging-Task) nie aktiviert wird. Es wird nur empfohlen, den Wert 10 anzugeben, wenn das LRB groß genug ist, um alle gleichzeitig aktiven Natural-Sitzungen zu bedienen. |

| Parameter | Parametername für Parameter-Dataset | Mögliche Werte | Standardwert | Kommentar |
|---|---|---|---|---|
| | | | | Der *high-water-mark*-Wert kann nicht niedriger sein als der aktuelle *low-water-mark*-Wert.<br><br>Dieser Wert wird immer ignoriert und auf 0 zurückgesetzt, wenn *CF-structure-name* einen Wert hat, unabhängig davon, was angegeben wurde. |
| *non-activity-time* | NONACTIVITY | 1 - 999999 | keiner | Die Anzahl der Stunden, die eine Sitzung inaktiv sein kann, bevor sie aus dem Roll File gelöscht wird.<br><br>Wird diese Zeit überschritten, dann wird die Sitzung bei der nächsten geplanten Zeitüberschreitungsprüfung gelöscht.<br><br>Wenn dieser Parameter weggelassen wird, wird keine Zeitüberschreitungsprüfung durchgeführt.<br><br>Dieser Parameter kann mit dem Operator-Kommando TIMEOUT geändert werden, siehe unten.<br><br>Sowohl der Authorized Services Manager als auch der Roll Server gestatten die Angabe eines Timeout-Wertes. Wenn Natural in einer SYSPLEX-Umgebung ausgeführt wird, müssen Sie für diesen Parameter und den NONACTIVITY-Parameter des Authorized Services Manager denselben Wert setzen. |
| *timeout-check-time* | TIMEOUTCHECK | 0000 - 2359 | keiner | Die Tageszeit, zu der die Zeitüberschreitungsprüfung durchgeführt werden soll.<br><br>Sitzungen werden gelöscht, wenn sie länger als die durch den vorhergehenden Parameter festgelegte Inaktivitätszeit inaktiv waren. |

| Parameter | Parametername für Parameter-Dataset | Mögliche Werte | Standardwert | Kommentar |
|---|---|---|---|---|
| | | | | Wird dieser Parameter weggelassen, dann wird keine Zeitüberschreitungsprüfung eingeplant.<br><br>Dieser Parameter kann mit dem Operator-Kommando TIMEOUT geändert werden, siehe unten. |
| *message-case* | MSGCASE | UCTRAN oder leer | leer | Geben Sie UCTRAN an, wenn der Roll Server alle seine Meldungen in Großbuchstaben ausgeben soll. |
| *repeat-timeout-check* | TIMEOUTREPEAT | 0000 - 1440 | keiner | Anzahl der Minuten zwischen aufeinanderfolgenden Zeitüberschreitungsprüfungen.<br><br>Damit kann eine Zeitüberschreitungsprüfung mehrmals am Tag erfolgen. |

### Parameter in der Parameterdatei

Die Parameterdatei ist eine physische sequenzielle Datei (DSORG=PS). In Ihrer JCL geben Sie diese Datei mit DDNAME RSMPARM an.

Die Parameter in der Parameterdatei werden als *name=value*-Paare angegeben. Geben Sie einen Parameter pro Zeile an, beginnend in Spalte 1. Das *name=value*-Paar wird durch das erste Leerzeichen abgeschlossen, und der Rest der Zeile wird nicht geprüft. Zeilen, die mit einem Stern (*) in Spalte 1 beginnen, werden als Kommentare behandelt. Parameter werden in Großbuchstaben umgesetzt, bevor sie verarbeitet werden.

Sie können auch einen Punkt als Füllzeichen verwenden, so dass die „=“-Zeichen in derselben Spalte ausgerichtet werden können. Dies ist jedoch *nicht* mit dem Parameter TIMEOUT möglich.

| Parameter | Mögliche Werte | Standardwert | Kommentar |
|---|---|---|---|
| SUBSID | 4-Byte-Zeichenkette ohne Leerzeichen | NAT*v* | Der angegebene Wert muss mit dem Wert des Natural-Profilparameters SUBSID übereinstimmen (*v* steht für Versionsnummer).<br><br>**Anmerkung:** Bei Natural under CICS siehe den Parameter CICSPLX im NCMDIR-Makro, um die entsprechende Subsystem-ID zu setzen. |
| NUMFILES | 0 - 5 | 1 | In einer Nicht-Parallel-Sysplex-Umgebung unter z/OS kann der Roll Server ohne Roll File arbeiten und nur den speicherinternen Local Roll Buffer verwenden. |

| Parameter | Mögliche Werte | Standardwert | Kommentar |
|---|---|---|---|
| NUMSLOTS | 1 - 32767 | keiner | Die Anzahl der LRB-Slots, die für jedes verwendete Roll File zugewiesen ist.<br><br>Die Gesamtzahl der LRB-Slots wird nach folgender Formel berechnet:<br>NUMFILES * NUMSLOTS |
| SLOTSIZE | 0..67108864 | Roll File Slot-Größe | Wert in Anzahl von Bytes.<br><br>Dieser Parameter muss angegeben werden, wenn kein Roll File verwendet wird.<br><br>Wenn Roll Files verwendet werden, wird dieser Parameter ignoriert und stattdessen die Slot-Größe der Roll Files verwendet. |
| STRUCTURE | beliebiger gültiger Strukturname | keiner | Wenn Sie weniger als 16 Zeichen angeben, werden Leerzeichen angefügt.<br><br>Geben Sie diesen Parameter nur an, wenn Sie die Coupling Facility (mit z/OS Parallel Sysplex) verwenden. Andernfalls kann die Verwendung dieses Parameters unnötigen Overhead verursachen. |
| LOWWATER | 0 - 9 | 7 (einzelnes z/OS)<br><br>0 (Sysplex) | Gibt die niedrigste Zuordnungsmarke in Schritten von 10 Prozent der Anzahl der LRB-Slots an.<br><br>Der LOWWATER-Wert kann nicht höher sein als der HIGHWATER-Wert.<br><br>Dieser Wert wird immer ignoriert und auf 0 zurückgesetzt, wenn STRUCTURE einen Wert hat, unabhängig davon, was angegeben wurde. |
| HIGHWATER | 0 - 10 | 8 (einzelnes z/OS)<br><br>0 (Sysplex) | Analog zum Parameter *low-water-mark*.<br><br>Der Wert 10 bedeutet, dass die Einstufungs-Task (Staging-Task) nie aktiviert wird. Es wird nur empfohlen, den Wert 10 anzugeben, wenn das LRB groß genug ist, um alle gleichzeitig aktiven Natural-Sitzungen zu bedienen.<br><br>Der HIGHWATER -Wert kann nicht niedriger sein als der LOWWATER-Wert.<br><br>Dieser Wert wird immer ignoriert und auf 0 zurückgesetzt, wenn STRUCTURE einen Wert hat, unabhängig davon, was angegeben wurde. |

| Parameter | Mögliche Werte | Standardwert | Kommentar |
|---|---|---|---|
| NONACTIVITY | 1 - 999999 | keiner | Die Anzahl der Stunden, die eine Sitzung inaktiv sein kann, bevor sie aus dem Roll File gelöscht wird.<br><br>Wird diese Zeit überschritten, wird die Sitzung bei der nächsten geplanten Zeitüberschreitungsprüfung gelöscht.<br><br>Wenn dieser Parameter weggelassen wird, wird keine Zeitüberschreitungsprüfung durchgeführt.<br><br>Dieser Parameter kann mit dem Operator-Kommando TIMEOUT, geändert werden (siehe *Betrieb des Roll Server*). |
| TIMEOUTCHECK | 0000 - 2359 | keiner | Die Tageszeit, zu der die Zeitüberschreitungsprüfung durchgeführt werden soll.<br><br>Sitzungen werden gelöscht, wenn sie länger als die mit dem Parameter NONACTIVITY angegebene Nichtaktivitätszeit inaktiv waren.<br><br>Wird dieser Parameter weggelassen, dann wird keine Zeitüberschreitungsprüfung eingeplant.<br><br>Dieser Parameter kann mit dem Operator-Kommando TIMEOUT, geändert werden (siehe *Betrieb des Roll Server*). |
| TIMEOUTREPEAT | 0 - 1440 | keiner | Die Anzahl der Minuten zwischen zwei aufeinanderfolgenden Timeout-Prüfungen.<br><br>Die erste Zeitüberschreitungsprüfung nach dem Roll Server-Start erfolgt nach der mit dem Parameter TIMEOUTREPEAT angegebenen Anzahl an Minuten.<br><br>Wenn TIMEOUTCHECK ebenfalls angegeben ist, wird die erste Zeitüberschreitungsprüfung zu dem mit TIMEOUTCHECK angegebenen Zeitpunkt durchgeführt und danach jeweils nach der mit dem Parameter TIMEOUTREPEAT angegebenen Anzahl an Minuten.<br><br>Die Angabe von 0 oder 1440 steht für keine Wiederholung der Zeitüberschreitungsprüfung. Die Zeitüberschreitungsprüfung wird nur zu der mit dem Parameter TIMEOUTCHECK angegebenen Zeit durchgeführt. |
| MSGCASE | UPPER oder MIXED | mixed case | Die Einstellung MSGCASE=UPPER bewirkt, dass alle Meldungstexte in Großbuchstaben angezeigt werden.<br><br>Bei MSGCASE=MIXED werden alle Meldungstexte in Groß-/Kleinschreibung angezeigt. |

| Parameter | Mögliche Werte | Standardwert | Kommentar |
|---|---|---|---|
| TIMEOUT=*option* | VERBOSE oder TERSE | VERBOSE | Anzeige (verbose = mit Text) oder Unterdrückung (terse = kein Text) der Meldungen RSM0096 während eines TIMEOUT-Vorgangs. |

**Anmerkung:** Der Local Roll Buffer befindet sich in einem Speicherobjekt „oberhalb der Grenze“. Verwenden Sie den MEMLIMIT-Parameter im EXEC-Statement, um sicherzustellen, dass genügend Speicher „oberhalb der Grenze“ zugewiesen werden kann.

**Anmerkung:** Wenn TIMEOUTCHECK angegeben ist und sein Wert vor dem Startzeitpunkt von NATRSMvr, the next TIMEOUTCHECK liegt, wird der nächste TIMEOUTCHECK-Wert ausgehend vom angegebenen TIMEOUTCHECK-Wert berechnet, so als ob bereits ein oder mehrere TIMEOUTCHECKs ausgeführt worden wären. Dies ist jedoch erst ab NATRSM93 möglich.

**Beispiel:**

### Beispiele für das Starten des Roll Servers als Batch Job

In den folgenden Beispielen steht *vr* für die jeweilige Version des Produkts.

```
//   EXEC PGM=NATRSMvr,PARM='NAvr,,1000'
//ROLLF1 DD DSN=SYSF.ROLLFILE
```

Die Subsystemkennung ist NA*vr*, es wird ein Roll File verwendet (Standard), und der lokale Roll Buffer hat 1000 Slots. Die verwendete Slot-Größe ist identisch mit der Slot-Größe des Roll File. Die Niedrigste Zuordnungsmarke ist 70 Prozent (Standard), die höchste Zuordnungsmarke ist 80 Prozent (Standard).

```
//   EXEC PGM=NATRSMvr,PARM=',5,1000,150000,NATROLL1',MEMLIMIT=800M
//ROLLF1 DD DSN=DASD1.ROLLFILE
//ROLLF2 DD DSN=DASD2.ROLLFILE
//ROLLF3 DD DSN=DASD3.ROLLFILE
//ROLLF4 DD DSN=DASD4.ROLLFILE
//ROLLF5 DD DSN=DASD5.ROLLFILE
```

Die Subsystemkennung ist NAT*v* (Voreinstellung), es werden fünf Roll Files verwendet, und jeder der fünf lokalen Roll Buffers hat 1000 Slots. Die Größe der LRB-Slots beträgt 150000 Bytes. Das Verzeichnis der Roll Files befindet sich in der Struktur NATROLL1 der Coupling Facility. Niedrigste und höchste Zuordnungsmarken werden ignoriert, da jeder Thread in das Roll File geschrieben wird (siehe ***Natural Roll Server-Funktionen***). Da dieser Job für z/OS gedacht ist, gibt die Option MEMLIMIT 800 Megabyte für die lokalen Roll Buffers an.

**Anmerkung:** Der Roll Server wird in den folgenden Fällen nicht gestartet:

- Ein anderer Roll Server läuft mit der gleichen *subsystem-id*.

- Ein anderer Roll Server greift auf ein Roll File zu, das in seiner JCL angegeben ist.
- Ein Roll File wurde mit dem Kommando `SETXCF FORCE` neu formatiert, ohne die CF-Struktur zurückzusetzen.

## Roll Server-Meldungen, Condition Codes und Abend Codes

Der Roll Server schreibt Informations- und Fehlermeldungen unter Verwendung des `WTO`-Makros (`ROUTCDE=11`) in `JESMSGLG`. Den Meldungen werden eine Meldungskennung und der Jobname des Roll Servers vorangestellt, zum Beispiel:

```
RSM0019 FBRSMvrs: Roll Server Version vrs is active
```

wobei *`vrs`* für die jeweilige Produktversion steht.

Die Meldungen werden im Abschnitt *Roll Server Messages* in der *Messages and Codes*-Dokumentation erläutert.

### Condition Codes des Roll Server Started Task

Die folgenden Condition Codes werden verwendet:

| | |
|---|---|
| 0 | Normale Beendigung |
| 12 | Falsche Parametereingabe |
| 16 | Laufzeitfehler |
| 20 | Abbruch ist aufgetreten |
| >100 | Initialisierungsfehler |

### Benutzer-Abend Codes

Wenn ein XCF- oder XES Service Call einen unerwarteten Return Code ausgibt, wird ein Abend (Abbruch) mit Dump (Speicherauszug) erzwungen. Register 14 des Abend-Registers enthält den Reason Code. Eine Beschreibung der Ursache finden Sie unter *Programming: Sysplex Services Reference* (IBM-Dokumentation). Wenn der Fehler nicht umgebungsspezifisch war, senden Sie den Dump an den Support.

Die folgenden Benutzer-Abbruchcodes werden verwendet:

| Abend Code | Ursache |
|---|---|
| U0200 | IXLCONN fehlgeschlagen |
| U0201 | IXLFORCE fehlgeschlagen |
| U0202 | IXLLIST fehlgeschlagen |
| U0203 | IXLDISC fehlgeschlagen |
| U0204 | IXCLEAVE fehlgeschlagen |
| U0301 | IXLLIST fehlgeschlagen |
| U0302 | IXCMSGO fehlgeschlagen |
| U0401 | IXLLIST fehlgeschlagen |
| U0501 | IXLLIST fehlgeschlagen |

## Return Codes und Reason Codes der Roll Server-Anforderung

Dies sind Codes, die Natural von den PC-Services-Routinen des Roll-Servers erhalten kann. Sie werden von den jeweiligen Teleprocessing-Schnittstellen (Natural CICS Interface oder Natural IMS TM Interface) gemeldet.

Eine Liste dieser Codes finden Sie unter *Return Codes and Reason Codes of the Roll Server Request* in der *Messages and Codes*-Dokumentation.

## Roll Server-Betriebsfunktionen

Die folgenden Kommandos können über das Operator-Kommando MODIFY an den Roll Server übergeben werden:

| Kommando | Beschreibung |
|---|---|
| HELP | Zeigt eine Übersicht über die verfügbare Syntax. |
| DIAGNOSE | Debugging-Funktion, die nur auf Anraten des Supports hin verwendet werden sollte.<br><br>Dieses Kommando hat keine Funktion. Es soll in Zukunft in Verbindung mit speziellen Zaps verwendet werden, um bei Bedarf die Diagnose von spezifischen Kundenproblemen zu unterstützen. |
| HWM *n* | High Water Mark: Setzt die höchste LRB-Zuordnungsmarke auf *n* mal 10 Prozent der Anzahl der LRB-Slots (*n* = 0 - 10).<br><br>Der HWM-Wert kann nicht niedriger sein als der aktuelle LWM-Wert.<br><br>HWM wird immer ignoriert und auf 0 zurückgesetzt, wenn STRUCTURE einen Wert hat. |

| Kommando | Beschreibung | |
|---|---|---|
| | Wenn *n* nicht angegeben wird, werden die aktuell niedrigste und die aktuell höchste Zuordnungsmarke angezeigt. | |
| LWM *n* | Low Water Mark: Setzt die höchste LRB-Zuordnungsmarke auf *n* mal 10 Prozent der Anzahl der LRB-Slots (*n* = 0 - 9).<br><br>Der LWM-Wert kann nicht höher sein als der aktuelle HWM-Wert.<br><br>LWM wird immer ignoriert und auf 0 zurückgesetzt, wenn STRUCTURE einen Wert hat.<br><br>Wenn *n* nicht angegeben wird, werden die aktuell niedrigste und die aktuell höchste Zuordnungsmarke angezeigt. | |
| RESET | Setzt die Intervallzähler zurück. | |
| SNAP | Debugging-Funktion: Der Adressraum des Roll Server wird in SYSUDUMP gespeichert. | |
| STATS | Schreiben von Roll Server-Statistiken nach JESMSGLG unter Verwendung des WTO-Makros (ROUTCDE=11). Die Statistiken enthalten Informationen über Roll-Out- und Roll-In-Aktivitäten sowie Roll-Datei-Ein-/Ausgaben. | |
| TERMinate \| STOP | Stoppt den Roll Server. Das Verzeichnis des Roll File und alle geänderten LRB-Slots werden in das Roll File geschrieben und der Adressraum wird geschlossen. Die Adressraumkennung ist bis zum nächsten IPL nicht mehr verfügbar.<br><br>Statistiken werden unter Verwendung des WTO-Makros (ROUTCDE=11) in JESMSGLG geschrieben. Die Statistiken enthalten Informationen über Roll-Out- und Roll-In-Aktivitäten sowie Roll-Datei-Ein-/Ausgaben. | |
| TIMEOUT | NAT *nnn* | Gibt den Parameter für die Nicht-Aktivitätszeit an oder ersetzt ihn. |
| | TOC *hhmm* | Gibt die Tageszeit der Zeitüberschreitungsprüfung an oder ersetzt diese. |
| | OFF | Deaktiviert die Zeitüberschreitungsprüfung. |
| | ON | Aktiviert die Zeitüberschreitungsprüfung wieder. |
| | NOW | Startet eine sofortige Zeitüberschreitungsprüfung. Die normale Zeitüberschreitungsprüfungsplanung (falls angegeben) bleibt in Kraft. |
| | TERSE | Unterdrückt die Meldungen RSM0072 und RSM0074 während der TIMEOUT NOW-Verarbeitung. Die Meldung RSM0047 Operator command: TIMEOUT NOW wird ebenfalls unterdrückt. |
| | VERBOSE | Zeigt die Meldungen RSM0072, RSM0074 und RSM0047 Operator-Kommando: TIMEOUT NOW.<br><br>Dies ist die Standardeinstellung. |
| | REPEAT *hhmm* | Gibt das Intervall an, in dem eine Zeitüberschreitungsprüfung erfolgt, oder ersetzt es. |

| Kommando | Beschreibung | |
|---|---|---|
| | ?<br><br>(oder keine Angabe) | Zeigt die aktuellen Timeout-Einstellungen an. Das Fragezeichen (?) ist optional und kann weggelassen werden. |

## Struktur der Coupling Facility zurücksetzen

Wenn ein TP-Monitor oder eine Server-Region, die den Roll Server verwendet, abbricht, gibt der Roll Server möglicherweise einen Fehler zurück. Beispielsweise könnte das Roll Server-Verzeichnis als voll gemeldet werden, weil vor dem Abbruch der Region keine Sitzungsbereinigung durchgeführt wurde. Um einen solchen Fehler zu beheben, müssen Sie die entsprechende Coupling Facility-Struktur löschen:

1. Schalten Sie alle Roll Server ab, die die betroffene CF-Struktur verwenden.
2. Setzen Sie das Operator-Kommando `SETXCF FORCE,STR,STRNAME=`*`structure-name`* ab, wobei *`structure-name`* der Name der für die Roll Server verwendeten CF-Struktur ist.
3. Starten Sie alle Roll Server neu.
4. Formatieren Sie die Roll Files.

Wenn im normalen Betrieb die Roll Files oder der LRB aufgrund der Anzahl der gleichzeitig aktiven Benutzer voll werden, starten Sie den Roll Server neu und geben Sie größere Roll Files oder einen größeren LRB an.

Sie können die Auslastung der Roll Files auch mit dem Dienstprogramm SYSTP überwachen: Funktionscode `R`, Funktion **Natural Subsystems and Roll Server Information**.

## Roll Server-Leistung optimieren

Als allgemeine Regel für die Leistungsverbesserung des Roll Servers gilt: Geben Sie dem Roll Server eine höhere Dispatching-Priorität als den Adressräumen, in denen Natural läuft.

Um herauszufinden, wo die Performance-Schwachstellen liegen, können Sie die Systemleistung mit der Funktion **Natural Subsystems and Roll Server Information** des Dienstprogramms SYSTP. Zeilenkommando `R` (Statistical Information) analysieren.

Achten Sie bei der Analyse der Roll Server-Statistiken insbesondere auf die folgenden Werte:

- Die Anzahl der direkten Schreibvorgänge.

  **Direct write** bedeutet, dass der empfangene Natural-Thread direkt in das Roll File geschrieben wurde.

Dafür gibt es zwei mögliche Ursachen:

1. Kein LRB-Slot verfügbar. Erhöhen Sie den LRB.
2. Der komprimierte Thread war größer als ein einzelner LRB-Slot. Erhöhen Sie die Größe des LRB-Slots.

- Die Anzahl der direkten Lesevorgänge.

  **Direct read** bedeutet, dass der angeforderte Thread nicht mehr im LRB war und direkt aus dem Roll File gelesen werden musste.

  Wenn das Verhältnis zwischen direkten Lesevorgängen und der Gesamtzahl der Lesevorgänge in einem einzelnen z/OS-System sehr hoch ist, dann ist die LRB zu klein. Erhöhen Sie sie.

  Wenn der Anteil der direkten Lesevorgänge an der Gesamtzahl der Lesevorgänge in einer z/OS-Parallel-Sysplex-Umgebung sehr hoch ist, kann dies auch bedeuten, dass es viele systemübergreifende Aktivitäten gibt, was wiederum bedeutet, dass eine Natural-Sitzung während ihrer Lebendauer ziemlich häufig die z/OS-Images wechselt.

- Die Anzahl der Staging-Wartezeiten (in einer einzelnen z/OS-Umgebung).

  Ein **Staging Wait** ist eine Situation, in der eine Schreibanforderung warten musste, bis die Einstufungs-Task (Staging Task) den LRB-Slot in das Roll File geschrieben hatte. Wenn das Verhältnis zwischen Staging-Wartezeiten und der Gesamtzahl der Schreibanforderungen sehr hoch ist, deutet dies darauf hin, dass die höchste und die niedrigste Zuordnungsmarke (Water Mark) unpassend gesetzt sind oder dass es einen Engpass auf dem Roll File-Gerät/Roll File-Kanal gibt.

  Aufgrund von Erfahrungen mit Stresstests wird Folgendes empfohlen:

  Wenn das Verhältnis der maximalen Anzahl aktiver Benutzer zur Anzahl der LRB-Slots sehr klein ist, sollten Sie die höchste Zuordnungsmarke erhöhen. Wenn nicht, sollte die höchste Zuordnungsmarke gesenkt werden.

  Die Differenz zwischen der höchsten und der niedrigsten Zuordnungsmarke sollte nicht größer als drei (30 Prozent) sein.

  Wenn die Anzahl der LRB-Slots deutlich größer ist als die maximale Anzahl der gleichzeitigen Benutzer, sollte die höchste Zuordnungsmarke idealerweise auf 10 gesetzt werden.

# Roll Server User Exits

Für den Roll Server gibt es zwei User Exits.

- NATRSU14
- NATRSU24

Dafür werden Beispiel-Quellcodemodule ausgeliefert.

### User Exit NATRSU14

Gibt die zu verwendende Roll File-Nummer an.

Konventionen für den Einstiegsaufruf:

- Register 1 adressiert die Parameterliste, die durch den folgenden DSECT beschrieben wird:

```
PLIST      DSECT
PLRSVER    DS CL4    Roll server version  (= 'vrs')
PLNRF      DS H      Number of roll files
PLUID      DS CL16   Userid
PLTSNUM1   DS H      Total number of slots  Roll file 1
PLUSNUM1   DS H      Number of slots in use Roll file 1
PLTSNUM2   DS H      Total number of slots Roll file 2
PLUSNUM2   DS H      Number of slots in use Roll file 2
PLTSNUM3   DS H      Total number of slots Roll file 3
PLUSNUM3   DS H      Number of slots in use Roll file 3
PLTSNUM4   DS H      Total number of slots Roll file 4
PLUSNUM4   DS H      Number of slots in use Roll file 4
PLTSNUM5   DS H      Total number of slots Roll file 5
PLUSNUM5   DS H      Number of slots in use Roll file 5
PLISTL EQU *-PLIST
```

  wobei *vrs* für die jeweilige Version des Produkts steht.
- Register 13 zeigt auf einen 36-Vollwort-Speicherbereich.
- Register 14 enthält die Rücksprungadresse.
- Register 15 enthält die Einstiegsadresse von NATRSU14.

Rücksprungaufrufkonvention: Register 15 enthält die Nummer des Roll File im Binärformat.

**Anmerkung:** Wenn in diesem User Exit Zugriffsregister verändert werden, müssen diese Zugriffsregister gespeichert und bei der Rückkehr wiederhergestellt werden. Dieser User Exit wird im primären Adressierungsmodus mit PSW Key 8 aufgerufen. Da er im speicherübergreifenden Modus läuft, darf kein SVC außer SVC 13 verwendet werden.

### User Exit NATRSU24

Gibt den zu verwendenden XCF-Gruppennamen an.

Einstiegsaufrufkonventionen:

- Register 1 zeigt auf einen 8-Byte-Bereich, in dem der Gruppenname generiert werden muss.
- zeigt auf einen 18-Vollwort-Speicherbereich.
- Register 14 enthält die Rücksprungadresse.
- Register 15 enthält die Einstiegsadresse von `NATRSU24`.

Als Vorgabe für den Gruppennamen verwendet der Roll Server die ganz linken 8 Bytes des CF-Strukturnamens.

Dieser User Exit wird im Primärmodus, `PSW Key 8` und im Task-Modus aufgerufen.

# IV Natural im Batch-Modus

Dieser Teil behandelt Aspekte, die bei der Ausführung von Natural im Batch-Modus zu berücksichtigen sind.

| | |
|---|---|
| **Natural im Batch-Modus - Allgemeines** | Generell zu beachtende Aspekte bei der Ausführung von Natural im Batch-Modus: Adabas-Datasets, Sort-Datasets, Unterstützung von Subtasking-Sitzungen in Batch-Umgebungen. |
| **Natural im Batch-Modus unter z/OS** | Behandelt spezielle Aspekte, die bei Natural im Batch-Modus unter dem Betriebssystem z/OS zu berücksichtigen sind. |

Eine Zusammenfassung der Profilparameter, die gelten, wenn Natural im Batch-Modus verwendet wird, finden Sie unter *Batch-Modus* im Abschnitt *Profilparameter gruppiert nach Kategorie* in der *Parameter-Referenz*-Dokumentation.

# 16 Natural im Batch-Modus - Allgemeines


- Adabas-Datasets ..... 140
- Sort-Datasets ..... 140
- Subtasking-Session-Unterstützung bei Batch-Modus-Umgebungen ..... 140

Dieses Dokument behandelt allgemeine Aspekte, die bei der Ausführung von Natural im Batch-Modus zu berücksichtigen sind.

## Adabas-Datasets

Adabas-Datasets dürfen nur im Einzelbenutzermodus angegeben werden. Sie sind identisch mit denen, die für die Ausführung eines normalen Anwendungsprogramms mit Adabas erforderlich sind. Ausführliche Informationen zu Adabas-Datasets finden Sie in der entsprechenden Adabas-Dokumentation.

## Sort-Datasets

Sort-Datasets müssen angegeben werden, wenn ein Natural-Programm, das ein `SORT`-Statement enthält, während der Natural-Sitzung ausgeführt werden soll.

Die Erfordernisse sind identisch mit denen für die Ausführung eines normalen COBOL- oder PL/1-Anwendungsprogramms, das das Sortierprogramm des Betriebssystems aufruft, und können je nach verwendetem Sortierprogramm variieren.

Natural benötigt die Zwischen-Datasets `SORTIN` und `SORTOUT` nicht, sondern kommuniziert mit dem Sortierprogramm über die Schnittstellen der User-Exit-Routinen `E15` und `E35`.

## Subtasking-Session-Unterstützung bei Batch-Modus-Umgebungen


- Zweck
- Voraussetzungen
- Funktionalität
- Starten einer Natural-Sitzung
- Starten einer Subtask

- Zugriff auf den Benutzer-Parameterbereich

### Zweck

Mit der Unterstützung von Subtasking können Sie mehrere Natural-Batch-Modus-Sitzungen innerhalb eines Adressraums ausführen. Dies ermöglicht eine parallele Verarbeitung innerhalb eines Adressraums, anstatt aufeinander folgende Job-Steps auszuführen, was den Durchsatz erheblich steigern kann.

Typischerweise nutzen Client/Server-Anwendungen und -Produkte diese Funktionalität, z. B. der Natural Remote Procedure Call (RPC). Es können mehrere Server-Subtasks gestartet werden, um mit entfernten Clients zu kommunizieren.

### Voraussetzungen

Wenn Sie den Natural-Nukleus neu starten wollen, muss er als Modul „reentrant“ gelinkt sein (Linkage-Editor-Option `RENT`).

Die Adabas-Link-Routine (`ADALNK`) muss mit Reentrancy-Unterstützung generiert werden.

### Funktionalität

Sie starten eine Subtask, indem Sie ein `CALL`-Statement aus einem Natural-Programm heraus absetzen. Die neue Natural-Sitzung („Subtask“) wird mit einer erweiterten Frontend-Parameterliste gestartet. Diese Liste enthält bis zu drei Parameter-Sets:

- dynamische Natural-Profilparameter,
- Startup-Parameter,
- Benutzerparameter.

Variablennamen für Standard-E/A-Datasets (z. B. `CMPRINT`) und andere Parameter für den Start der Batch-Modus-Schnittstelle können vom startenden Programm im Startup-Parameterbereich übergeben werden. Standard-E/A-Datasets können undefinierte oder Dummy-Datasets sein. Sie können einer Sitzung zugehören oder von mehreren Sitzungen gemeinsam genutzt werden.

Außerdem steht eine `CALL`-Schnittstelle zum Lesen des Benutzer-Parameterbereichs mit einem Natural-Programm zur Verfügung.

## Starten einer Natural-Sitzung

**Erweiterte Parameterliste**

Die Natural-Batch-Modus-Schnittstelle ohne erweiterte Parameterliste erhält die initiale Kontrolle vom Betriebssystem über den Standard-Linkage-Aufruf. Register 1 zeigt auf eine Adresse, bei der das höherwertige Bit als letzter Adressindikator eingeschaltet ist. Diese Adresse verweist auf ein Halbwortfeld, das die Länge des folgenden Parameterbereichs enthält.

Die erweiterte Parameterliste enthält bis zu drei Parameteradressen. Dies wird durch das höherwertige Bit in der letzten Adresse angezeigt, die die erste, zweite oder dritte Adresse sein kann. Alle Parameteradressen verweisen auf ein Halbwortfeld, das die Parameterlänge des folgenden Parameterbereichs enthält. Die Länge Null bedeutet, dass es keinen Parameterbereich gibt.

- Der erste Parameterbereich enthält die dynamischen Profilparameter für die Natural-Sitzung.
- Der zweite enthält spezielle Startup-Parameter für die Initialisierung der Batch-Modus-Schnittstelle.
- Der dritte enthält einen Benutzer-Parameterbereich, auf den während der Natural-Sitzung zugegriffen werden kann.

**Startup-Parameterbereich**

Wenn mehrere Natural-(Sub-)Tasks im Batch-Modus in derselben Region laufen, greifen diese Sitzungen standardmäßig auf dieselben Natural-Standard-E/A-Datasets zu (z. B. `CMPRINT`, `CMSYNIN` usw.), da es keine Natural-Profilparameter gibt, um diese Dateinamen festzulegen. Auch die Natural-Systemvariablen `*INIT-ID` und `*INIT-USER` sind standardmäßig identisch, da sie für den Batch-Modus definiert sind.

Um eindeutige Standard-E/A-Dataset-Namen und eindeutige Kennungen (IDs) für Natural-Subtask-Sitzungen bereitzustellen, können die Startup-Parameter in der erweiterten Parameterliste verwendet werden, um die Standardeinstellungen des Natural-Systems zu überschreiben. Der Startup-Parameterbereich ist eine Tabelle mit Paaren von 8-Zeichen-Feldern:

- Der erste Eintrag enthält das 8-Byte-Schlüsselwort, das ersetzt werden soll,
- der zweite Eintrag enthält den 8-Byte-Ersetzungswert.

Schlüsselwörter und Ersetzungswerte müssen ggf. mit Leerzeichen am Ende aufgefüllt werden.

Die folgenden Schlüsselwörter sind gültig:

| Schlüsselwort | Erläuterung |
|---|---|
| `CMHCOPY` | Ständiges Ziel der Hardcopy |
| `CMSYNIN` | Name des Kommando-Eingabe-Dataset |
| `CMOBJIN` | Name des Objekt-Eingabe-Dataset |
| `CMPRINT` | Name des Standard-Ausgabe-Dataset |
| `CMPRMIN` | Name des Eingabe-Dataset für dynamische Parameter |

| Schlüsselwort | Erläuterung |
|---|---|
| CMPLOG | Name des Ausgabe-Dataset für dynamische Parameter |
| CMTRACE | Name des Trace-Ausgabe-Dataset |
| INITID | Name des Job-Step (Systemvariable *INIT-ID) |
| MSGCLASS | Spool-Klasse für die dynamische Zuordnung von CMPRINT und CMTRACE |
| NATRJE | Name des Job-Submit-Dataset |
| STEPLIB | Name der Programm-Load-Library (siehe auch Profilparameter LIBNAM) |
| SUBPOOL | z/OS-Speicher-Subpool (0 - 127, rechtsbündig) |
| USERID | Initiale Benutzerkennung (Systemvariable *INIT-USER) |

Die Verwendung dieser Einträge ist optional. Es ist keine bestimmte Reihenfolge vorgeschrieben. Ein leerer Wert für einen Dataset bedeutet, dass dieser Dataset nicht verfügbar oder leer ist.

**Benutzer-Parameterbereich**

Das Format des Benutzer-Parameterbereichs ist frei. Auf ihn kann von jedem Natural-Programm aus über eine spezielle CALL-Schnittstelle zugegriffen werden, siehe *Zugriff auf den Benutzer-Parameterbereich*.

### Starten einer Subtask

Das folgende Call Interface wird bereitgestellt, damit Natural-Programme eine Subtask im selben Adressraum starten können.

| | |
|---|---|
| PGMNAME | Name des Natural-Nukleus, der die Kontrolle übernimmt (obligatorisch). Um mit demselben Nukleus neu zu starten, kann als erstes Zeichen ein Stern angegeben werden. Der eigentliche Nukleus-Name wird in diesem Feld zurückgegeben. |
| NATPARML | Dynamischer Natural-Parameterbereich |
| STRPARML | Startup-Parameterbereich |
| USRPARML | Benutzerparameterbereich |

Alle Parameterbereiche müssen mit der Länge der folgenden Parameter beginnen. Das folgende Beispiel veranschaulicht die Verwendung von CMTASK.

**Beispiel:**

```
DEFINE DATA LOCAL
01 PGMNAME (A8) INIT <'*'>
01 PARM1
02 NATPARML (I2) INIT <30>
02 NATPARMS (A30) INIT <'INTENS=1,IM=D,STACK=MYPROG'>
01 PARM2
02 STRPARML (I2) INIT <32>
02 STRPARM1 (A16) INIT <'CMPRINT SYSPRINT'>
02 STRPARM2 (A16) INIT <'CMPRMIN MYPARMS'>
01 PARM3
```

```
02 USRPARML (I2) INIT <80>
02 USRPARMS (A80) INIT <'special user parameters'>
END-DEFINE
CALL 'CMTASK' PGMNAME NATPARML STRPARML USRPARML
END
```

Ein Beispielprogramm, ASYNBAT, ist in der Library SYSEXTP vorhanden.

### Zugriff auf den Benutzer-Parameterbereich

Der beim Startup übergebene Benutzerparameterbereich kann von jedem Natural-Programm mit dem folgenden CALL-Statement gelesen werden:

```
CALL 'CMUPARM' USRPARML USRPARMS
```

USRPARML ist die Länge (I2) des Bereichs USRPARMS (vor dem Aufruf) und die Länge der zurückgegebenen Daten (nach dem Aufruf). USRPARMS ist der Parameterdatenbereich.

Wenn die Länge der zurückzugebenden Daten größer als die Bereichslänge ist, werden die Daten auf die Bereichslänge abgeschnitten. Die folgenden Rückgabecodes sind möglich:

| | |
|---|---|
| 0 | Daten erfolgreich verschoben |
| 4 | Daten verschoben, aber abgeschnitten |
| 8 | Keine Daten vorhanden |
| 12 | Längenwert nicht positiv |
| 16 | Unzureichende Anzahl an Parametern |

Ein Beispielprogramm, GETUPARM, ist in der Library SYSEXTP vorhanden.

# 17 Natural im Batch-Modus unter z/OS


- Natural z/OS-Batch-Schnittstelle ..... 146
- Treiberparameter für z/OS Batch ..... 146
- Von Natural im z/OS-Batch-Modus verwendete Datasets ..... 146

Dieses Kapitel behandelt spezielle Aspekte, die Natural im Batch-Modus unter dem Betriebssystem z/OS betreffen.

Siehe auch folgende Abschnitte im Kapitel ***Natural im Batch-Modus - Allgemeines***:

- ***Adabas-Datasets***
- ***Sort-Datasets***
- ***Subtasking-Session-Unterstützung bei Batch-Modus-Umgebungen***

## Natural z/OS-Batch-Schnittstelle

Die Natural z/OS-Batch-Schnittstelle besteht aus dem Objektmodul `NATOS`, das während des Installationsvorgangs für das Basisprodukt Natural (*Installation for z/OS*-Dokumentation) mit dem Natural-Nukleus verknüpft wird.

Sie können die Natural z/OS-Batch-Schnittstelle an Ihre Anforderungen anpassen, indem Sie die Parametereinstellungen im Makro `NTOSP` im Natural-Parametermodul während des entsprechenden Installationsschritts ändern.

`NTOSP` ist vollständig reentrant und kann oberhalb der 16-MB-Grenze laufen. Mehrere Natural-Sitzungen können innerhalb einer Batch-Region parallel gestartet werden. Siehe ***Subtasking-Session-Unterstützung bei Batch-Modus-Umgebungen***.

## Treiberparameter für z/OS Batch

Informationen zu den Treiberparametern, die für z/OS im Batch-Modus verfügbar sind, finden Sie in der Beschreibung des Profilparameters `OSP` oder des Parameter-Makros `NTOSP` in der *Parameter-Referenz*-Dokumentation.

## Von Natural im z/OS-Batch-Modus verwendete Datasets

Die folgenden Datasets sind erforderlich, wenn bestimmte Funktionen während einer Natural z/OS-Batch-Modus-Sitzung verwendet werden:

| Dataset | Erläuterung |
|---|---|
| CMEDIT | Software AG Editor-Arbeitsdatei |
| CMHCOPY | Hardcopy-Druckausgabe |
| CMOBJIN | Eingabe für Natural-INPUT-Statements |
| CMPLOG | Ausgabe des dynamischen Profilparameter-Reports |
| CMPRINT | Ausgabe des Primär-Reports |
| CMPRMIN | Dynamische Profilparameter-Eingabe |
| CMPRT*nn* | Zusätzliche Reports 01 - 31 |
| CMSYNIN | Primäre Kommandoeingabe |
| CMTRACE | Externe Trace-Ausgabe |
| NATRJE | Job Submit-Ausgabe |
| STEPLIB | Load Library für externe Module |
| CMWKF*nn* | Arbeitsdateien 01 - 32 |

Diese Datasets werden im Folgenden beschrieben.

Für sequenzielle Datenausgabe-Sets gelten die folgenden DCB-RECFM/LRECL-Standardinformationen:

RECFM=FBA und LRECL=133

### CMEDIT - Software AG Editor-Arbeitsdatei

Der VSAM-Dataset für die Software AG Editor-Arbeitsdatei ist erforderlich, wenn ein lokaler oder globaler Software AG Editor Buffer Pool verwendet werden soll.

Wenn er nicht in der JCL definiert ist, wird der Name der Editor-Arbeitsdatei, der im Subparameter DSNAME des Profilparameters EDBP oder im Parametermakro NTEDBP angegeben ist, von Natural verwendet, um die dynamische Zuordnung für die Editor-Arbeitsdatei durchzuführen.

Alternativ kann der Profilparameter EDPSIZE verwendet werden, um mit einem Editor-Hilfs-Buffer-Pool zu arbeiten, für den keine Editor-Arbeitsdatei erforderlich ist. Informationen zur Installation des Software AG Editor finden Sie unter *Software AG Editor installieren* in der *Installation für z/OS*-Dokumentation.

### CMHCOPY - Optionale Report-Ausgabe für Hardcopy

Der Standardname des Hardcopy-Ausgabe-Dataset ist `CMHCOPY`. Er kann geändert werden durch

- den Subparameter `DEST` des Profilparameters `PRINT` für Druckdatei `0`,
- den Profilparameter `HCDEST`, der ein Äquivalent zu `PRINT=((0),DEST=...)`, ist,
- die Einstellung der Systemvariablen `*HARDCOPY` während der Sitzung,
- das Terminalkommando `%H` während der Sitzung.

Mit den Subparametern des Profilparameters `PRINT` für Druckdatei `0` können die Standardwerte für den Hardcopy-Dataset geändert werden. Der voreingestellte Dataset-Name `CMHCOPY` impliziert `CLOSE=FIN` für den Hardcopy-Print-Dataset, d.h. nachdem der Dataset für die Ausgabe geöffnet wurde, wird eine nachträgliche Änderung des Hardcopy-Print-Ausgabe-Dataset-Namens nicht beachtet. Wird zum Zeitpunkt des Öffnens ein anderer Name festgelegt, wird der Hardcopy-Dataset entsprechend dem Subparameter `CLOSE` des Profilparameters `PRINT` für Druckdatei `0` geschlossen.

Während der Sitzung kann der Hardcopy-Dataset durch die dynamische Zuordnung über die Anwendungsprogrammierschnittstelle USR2021N, siehe Dienstprogramm SYSEXT freigegeben und neu zugewiesen werden (vor dem Öffnen oder nach dem Schließen).

### CMOBJIN - Eingabe für Natural-INPUT-Statements

Dieser Dataset kann verwendet werden, um Daten mit dem Natural-`INPUT`-Statement zu lesen, anstatt aus dem primären Input-Dataset `CMSYNIN`.

Die Verwendung von `CMOBJIN` wird durch den Profilparameter `OBJIN` gesteuert. Die Länge der Eingabe-Satzlänge für Natural wird durch den Profilparameter `SL` bestimmt. Die maximal unterstützte Satzlänge (`LRECL`) beträgt `255`. Das Datensatzformat (`RECFM`) kann fest oder variabel sein.

### CMPLOG - Ausgabe des Reports der dynamischen Profilparameter

Wenn der Profilparameter `PLOG=ON` gesetzt und der Dataset `CMPLOG` verfügbar ist, werden die ausgewerteten dynamischen Profilparameter während der Sitzungsinitialisierung in diesen Dataset geschrieben. Wenn der Dataset `CMPLOG` nicht verfügbar ist, werden die ausgewerteten dynamischen Profilparameter in `CMPRINT` geschrieben.

### CMPRINT - Primäre Report-Ausgabe

CMPRINT wird für den primären Output-Report verwendet, der aus den DISPLAY-, PRINT- und WRITE-Statements in einem Natural-Programm resultiert.

Das Datensatzformat (RECFM) für CMPRINT ist FBA. Wenn keine DCB-Informationen für LRECL aus dem Dataset oder der JCL verfügbar sind, wird LRECL=133 als Voreinstellung verwendet. Wenn keine DCB-Informationen für BLKSIZE aus dem Dataset oder der JCL verfügbar sind, ist BLKSIZE das 10-fache des Wertes von LRECL als Voreinstellung.

Wenn in der JCL nicht definiert, wird CMPRINT dynamisch zugeordnet als

```
//CMPRINT DD SYSOUT=*
```

wenn der erste Datensatz geschrieben werden soll.

### CMPRMIN - Dynamischer Parameter-Dataset

CMPRMIN kann als dynamischer Parameter-Dataset verwendet werden, um die Längenbeschränkung für die Zeichenkette im Job-Control-Schlüsselwort PARM des EXEC-Statement zu umgehen.

Falls verfügbar, wird diese Datei während der Sitzungsinitialisierung gelesen, um die dynamischen Profilparameter zu erhalten.

Alle Eingabesätze von CMPRMIN werden zu einer einzigen Parameterzeichenkette verkettet. Nur die ersten 72 Positionen eines jeden CMPRMIN-Datensatzes sind von Bedeutung. Nachfolgende Leerzeichen am Ende jedes Datensatzes werden abgeschnitten. Wenn das letzte Nicht-Leerzeichen ein Komma ist, werden alle nachstehenden Leerzeichen abgeschnitten, andernfalls wird nur ein Leerzeichen als Trennzeichen belassen. Es werden keine Kommas eingefügt.

Zusätzliche dynamische Parameter können mit dem Job-Control-Schlüsselwort PARM übergeben werden. Sie werden an das Ende der Parameterzeichenkette angehängt, die aus der Eingabe von CMPRMIN gebildet wurde, d.h. sie können dazu verwendet werden, die Parameter von CMPRMIN zu überschreiben.

Ein Kommentar beginnt mit einem Schrägstrich /* und endet mit einem Schrägstrich */ und kann an beliebiger Stelle in Ihrem Dataset stehen. Kommentare können sich über mehrere Zeilen in Ihrer Datei erstrecken.

### CMPRTnn - Zusätzliche Reports 01 - 31

Diese Datasets können von Natural-Druckdatei-Statements wie `WRITE` (*nn*) verwendet werden. Wenn keine DCB-Informationen (z.B. `RECFM`, `LRECL`, `BLKSIZE`) vorhanden sind, werden die Standardwerte durch den Profilparameter `PRINT` oder das Makro `NTPRINT` im Natural-Parametermodul definiert. Die Namen der Druckdateien können durch den Subparameter `DEST` überschrieben werden.

### CMSYNIN - Primäre Kommandoeingabe

Dieser Dataset wird verwendet, um Kommandoeingaben und Daten zu lesen, die vom Natural-`INPUT`-Statement angefordert werden. Letzteres wird durch den Profilparameter `OBJIN` gesteuert (siehe auch `CMOBJIN`).

Die Länge der Eingabesatzdaten für Natural wird durch den Profilparameter `SL` bestimmt. Die maximal unterstützte Satzlänge (`LRECL`) beträgt `255`. Das Satzformat (`RECFM`) kann fest oder variabel sein.

### CMTRACE - Optionale Report-Ausgabe für das Natural-Tracing

Wenn der Profilparameter `ETRACE=ON` gesetzt ist oder das entsprechende Terminalkommando `%TRE+` abgesetzt wurde, wird jede Natural-Trace-Ausgabe während der Sitzung in den `CMTRACE`-Dataset geschrieben. Um die Natural-Komponenten zu definieren, die überwacht werden sollen, ist der Profilparameter `TRACE` erforderlich.

Wenn der Dataset `CMTRACE` nicht verfügbar ist, wird er dynamisch wie folgt zugewiesen

```
//CMTRACE DD SYSOUT=*
```

wenn der erste Trace-Satz geschrieben werden soll.

### NATRJE - Job Submit-Ausgabe

Dieser Dataset wird für die **Natural Job Submitting Utility** verwendet. Wenn er nicht definiert ist, wird er dynamisch zugeordnet als

```
//NATRJE DD SYSOUT=(A,INTRDR)
```

wenn der erste Job übergeben wird.

### STEPLIB - Load Library für externe Module

`STEPLIB` ist der Standardname der Load Library für das Laden externer Module, z.B:

- den umgebungsunabhängigen Nukleus (Profilparameter `NUCNAME`),
- ein separates Adabas-Linkroutinenmodul (Profilparameter `ADANAME`),
- das Session-Backend-Programm (Profilparameter `PROGRAM`),
- alle externen Subprogramme, die nicht mit dem **Natural-Parametermodul** verlinkt sind..

Der Name der Load Library kann über den Profilparameter `LIBNAM` geändert werden. Der angegebene Name der Load Library muss durch ein DD-Statement in der JCL definiert werden.

### CMWKFnn - Arbeitsdateien 01 - 32

Diese Datasets können von Natural-Arbeitsdatei-Statements wie `READ WORK` *nn* und `WRITE WORK` *nn* verwendet werden.

Wenn keine DCB-Informationen (`RECFM`, `LRECL`, `BLKSIZE` usw.) in der JCL oder im VTOC-Eintrag für den Dataset vorhanden sind, werden die Standardwerte durch den Profilparameter `WORK` oder das Makro `NTWORK` im Natural-Parametermodul definiert.

Die Workfile-Dataset-Namen können durch den Subparameter `DEST` überschrieben werden.

# V Natural Buffer Pools

Dieser Teil enthält Informationen über die verschiedenen Speicherverwaltungsfunktionen, die einem Natural-Administrator unter dem Betriebssystem z/OS zur Verfügung stehen.

**Natural Buffer Pool - Allgemeines**

**Natural Global Buffer Pool unter z/OS**

Einen funktionalen Überblick über den Natural Buffer Pool finden Sie unter *Natural Buffer Pool* in der *Natural-Systemarchitektur*-Dokumentation.

Eine Übersicht über die Natural-Profilparameter, mit denen Sie die Natural-Buffer-Pools beeinflussen können, finden Sie unter *Speicherverwaltung* im Abschnitt *Profilparameter sortiert nach Kategorien* in der *Parameter-Referenz*-Dokumentation.

# 18 Natural Buffer Pool - Allgemeines


- Natural Buffer Pool - Funktionsprinzip ..... 156
- Buffer Pool-Überwachung und -Verwaltung ..... 162
- Globaler Natural Buffer Pool ..... 165

Der Buffer Pool ist ein Speicherbereich, in dem Natural-Programme in Vorbereitung auf ihre Ausführung abgelegt werden. Programme werden in den Buffer Pool hinein- und herausgeschoben, wenn Natural-Benutzer Natural-Objekte anfordern. Vom Konzept her hat er eine ähnliche Funktion wie ein Betriebssystem, das Programme in und aus einem reentranten Bereich lädt. Der Natural Buffer Pool ist ein integraler Bestandteil von Natural in allen unterstützten Umgebungen.

## Natural Buffer Pool - Funktionsprinzip

Natural generiert reentranten (ablauf-invarianten) Natural-Objektcode. Ein kompiliertes Programm wird in den Buffer Pool geladen und vom Buffer Pool aus ausgeführt. So ist es möglich, dass eine einzige Kopie eines Natural-Programms von mehreren Benutzern gleichzeitig ausgeführt werden kann.

In diesem Abschnitt werden die folgenden Themen behandelt:

- Objekte im Puffer Pool
- Verzeichniseinträge
- Text Pool
- Buffer Pool-Hashtabelle
- Buffer Pool-Initialisierung
- Buffer Pool-Suchmethoden
- Lokaler Buffer Pool
- Globaler Buffer Pool
- Buffer Pool Cache (BPC)

### Objekte im Puffer Pool

Objekte im Puffer Pool können Programme, Unterprogramme, Maps (Masken) und globale Datenbereiche (GDAs) sein. Globale Datenbereiche werden nur zur Kompilierung in den Buffer Pool gestellt. In diesem Fall werden zwei Objekte mit demselben Namen in den Buffer Pool geladen: die GDA selbst und die entsprechende Symboltabelle.

### Verzeichniseinträge

Wenn ein Natural-Objekt in den Buffer Pool geladen wird, wird diesem Objekt ein als Verzeichniseintrag bezeichneter Kontrollblock zugeordnet.

Ein Verzeichniseintrag enthält Informationen wie den Namen des Objekts, die Library, zu der es gehört, die Datenbankkennung (DBID) und die Nummer der Natural-Systemdatei (FNR), aus der das Objekt abgerufen wurde, sowie einige statistische Informationen (z.B. die Anzahl der Benutzer, die das Programm zu einem bestimmten Zeitpunkt gleichzeitig ausführen).

Wenn ein Benutzer ein Programm ausführt, überprüft Natural die Verzeichniseinträge, um festzustellen, ob das Programm bereits in den Buffer Pool geladen wurde. Wenn dies nicht der Fall

ist, wird eine Kopie des Programms aus der entsprechenden Natural-Systemdatei abgerufen und in den Buffer Pool geladen.

Wenn ein Objekt in den Buffer Pool geladen wird, können ein oder mehrere andere Natural-Objekte, die gerade nicht ausgeführt werden, aus dem Buffer Pool gelöscht werden, um Platz für das neu geladene Objekt zu schaffen. Wenn das neue Objekt geladen wird, wird ein neuer Verzeichniseintrag erstellt, um dieses Objekt zu identifizieren.

Wenn ein Objekt aus der Systemdatei gelöscht wird, wird es auch aus dem Buffer Pool gelöscht, sobald es nicht mehr verwendet wird. Wenn ein Objekt neu katalogisiert oder gestowed wird, wird seine alte Version ebenfalls aus dem Buffer Pool gelöscht, sobald sie nicht mehr verwendet wird. Wenn es erneut zur Ausführung angefordert wird, wird die neue Version aus der Systemdatei in den Buffer Pool geladen.

### Text Pool

Der eigentliche Objektcode eines Programms, der in den Buffer Pool geladen wird, befindet sich in einem Bereich, der Text Pool genannt wird, und muss als zusammenhängendes Stück Speicher innerhalb dieses Text Pool zugewiesen werden. Dieser Text Pool ist in eine Anzahl von 4-KB-Puffern unterteilt. Diese Größe ist willkürlich und kann vom Natural-Administrator nach eigenem Ermessen geändert werden. Wenn ein Objekt geladen wird, werden ein oder mehrere zusammenhängende Textpuffer zugeordnet, um den Objektcode des Objekts zu speichern.

### Buffer Pool-Hashtabelle

Dieser Abschnitt gilt nur für Buffer Pools mit `TYPE=NAT`.

Um die Suchzeit für die Suche nach einem Objekt im Buffer Pool-Verzeichnis zu beschleunigen, wird eine Hashtabelle verwendet. Die Anzahl der Einträge in der Hashtabelle ist doppelt so groß wie die Anzahl der Verzeichniseinträge, aufgerundet auf die nächste Primzahl. Dadurch wird sichergestellt, dass zu jedem Zeitpunkt nur die Hälfte der Tabelle gefüllt ist und dass die Wahrscheinlichkeit von Kollisionen nahe Null liegt. Infolgedessen ist die durchschnittliche Anzahl der Versuche, ein vorhandenes Objekt in der Hashtabelle zu finden, theoretisch kleiner als 2.

Das Hashkriterium ist der acht Zeichen lange Programmname. Wenn mehr als ein Programmname an dieselbe Stelle in der Hashtabelle gehasht wird, löst eine Überlauftechnik die Kollisionen auf.

Der Speicherbedarf für die Hashtabelle beträgt etwa 16 Byte pro Textblock. Dadurch wird der verfügbare Speicherplatz im Text Pool zwischen 1,6 % (1 KB Textblöcke) und 0,1 % (16 KB Textblöcke) reduziert.

### Buffer Pool-Initialisierung

Im Falle eines globalen Buffer Pool erfolgt die Initialisierung während des Starts des globalen Buffer Pool.

Bei einem lokalen Buffer Pool variiert der Initialisierungszeitpunkt in Abhängigkeit von der Umgebung.

- Im Batch-Modus und unter TSO erfolgt die Initialisierung zu Beginn der Ausführung der Natural-Sitzung.
- In einer TP-Monitorumgebung erfolgt die Initialisierung im Allgemeinen, wenn der erste Benutzer Natural unter diesem TP-Monitor aufruft. Unter Com-plete und CICS ist es auch möglich, den lokalen Buffer Pool zu initialisieren, wenn der TP-Monitor gestartet wird (siehe auch ***Tipp*** im Abschnitt *Preload-Liste*).

### Buffer Pool-Suchmethoden

Wie bereits erwähnt und im Folgenden erläutert, gibt es verschiedene Suchmethoden für die Zuordnung von Speicherplatz im Buffer Pool.

**Um eine Suchmethode auszuwählen, verwenden Sie:**

- Die Natural-Profilparameter `BPMETH` und `BPI`.
  Oder das Makro `NTBPI` im Natural-Parametermodul.
  Oder den Funktionsparameter `METHOD` des globalen Buffer Pool.

Eine Beschreibung dieser Parameter und des Makros `NTBPI` finden Sie in der Natural-*Parameter-Referenz*-Dokumentation.

Nachfolgend finden Sie Informationen zu den Suchmethoden:

- `METHOD=S`
- `METHOD=N`
- **Auswahl der Suchmethoden**

**METHOD=S**

`METHOD=S` bedeutet, dass ein Auswahlprozess als Suchalgorithmus für die Zuweisung von Speicherplatz im Buffer Pool verwendet wird, um den für einen neuen Ladevorgang erforderlichen Platz zu erhalten.

Der verwendete Auswahlprozess ist eine Kombination aus den Suchalgorithmen Algorithmus 1 und Algorithmus 2:

- **Algorithmus 1**
  Suchalgorithmus 1 versucht, im Buffer Pool Speicherplatz zu finden, der entweder frei ist oder von einem unbenutzten (aktiven, aber nicht verwendeten) Objekt belegt ist.

  Wird freier Speicherplatz in genau der erforderlichen Objektgröße gefunden, wird der Auswahlprozess sofort beendet. Andernfalls wird die Suche fortgesetzt, indem der Buffer Pool von oben nach unten durchsucht wird und die Einträge im Buffer Pool auf die am besten passende Größe verglichen werden. Bei unbenutzten Objekten wird bei der Suche auch die zuletzt angehängte Zeit des Objekts berücksichtigt, d. h. der Zeitpunkt, an dem das Objekt zuletzt bei einem Lade- oder Suchvorgang referenziert wurde.

  Wenn der Auswahlprozess abgeschlossen ist, wird entweder freier Speicherplatz oder der Speicherplatz eines unbenutzten Objekts mit einer Größe größer oder gleich der angeforderten Größe ausgewählt. Für die Suche gilt folgende Vorrangregel: zuerst wird der freie Speicherplatz und dann der Speicherplatz der unbenutzten Objekte ausgewählt. Bei unbenutzten Objekten werden die ältesten Objekte zuerst entfernt.

  Wenn der Auswahlprozess von Algorithmus 1 nicht erfolgreich war, wird Algorithmus 2 aufgerufen.

- **Algorithmus 2**
  Suchalgorithmus 2 beginnt, wenn Algorithmus 1 fehlschlägt.

  Algorithmus 2 beginnt die Suche an einer Position im Buffer Pool, die von Algorithmus 1 übergeben wird, und versucht, zwei oder mehr Entitäten (freier Speicher und/oder von ungenutzten Objekten belegter Platz) zu kombinieren, um den erforderlichen Speicherplatz für einen neuen Ladevorgang zu erhalten. Das Alter des Objekts wird jedoch nicht berücksichtigt.

  Algorithmus 2 setzt die Verarbeitung bis zum unteren Ende des Textdatensatzabschnitts fort und fährt gegebenenfalls bis zum oberen Ende des Textdatensatzabschnitts fort, um einen letzten Durchlauf von oben nach unten zu machen. Wenn immer noch kein Platz vorhanden ist, schlägt Algorithmus 2 fehl, das Objekt kann nicht geladen werden und Natural gibt eine entsprechende Fehlermeldung aus.

**METHOD=N**

`METHOD=N` bedeutet, dass der nächste freie oder ungenutzte Speicherplatz verwendet wird, um den für einen neuen Ladevorgang erforderlichen Speicherplatz zu erhalten. Ungenutzter Platz ist Platz, der von einem aktiven, aber nicht benutzten Objekt belegt ist.

Die Suche nach dem nächsten freien Speicherplatz beginnt an einem Zeiger, der den Buffer Pool in einem Wrap-around-Verfahren durchläuft. Alle nächstverfügbaren Buffer Pool-Einträge, die frei sind oder ungenutzte Objekte enthalten, können verwendet und möglicherweise miteinander verkettet werden, um die angeforderte Speichermenge zu erhalten.

Wenn während einer Zuordnungsanforderung das untere Ende des Buffer Pool erreicht wird, wird der Zeiger an das obere Ende des Buffer Pool gesetzt und eine letzte Suche im Buffer Pool von oben nach unten durchgeführt. Wenn das Ende des Buffer Pool erneut erreicht wird

und das Objekt nicht geladen werden kann, schlägt das Laden fehl und Natural gibt eine entsprechende Fehlermeldung aus.

`METHOD=N` kann insbesondere für große Buffer Pools in Kombination mit der **Buffer Pool Cache**-Funktion in Betracht gezogen werden. Einzelheiten dazu finden Sie im folgenden Abschnitt *Auswahl der Suchmethoden*.

**Auswahl der Suchmethoden**

Standardmäßig wird `METHOD=S` verwendet. Der Vorteil dieser Methode besteht darin, dass eine sorgfältige Suche durchgeführt wird, um Platz zuzuordnen, wobei die Größe und das Alter der Objekte berücksichtigt werden und gewährleistet wird, dass die entbehrlichsten ungenutzten Objekte aus dem Buffer Pool entfernt werden, um Platz für einen neuen Ladevorgang zu schaffen.

Ein Nachteil von `METHOD=S` kann die hohe CPU-Zeit sein, die durch den Auswahlprozess beim Durchsuchen des Buffer Pool von oben nach unten verbraucht wird.

Der Vorteil von `METHOD=N` ist der kurze Auswahlprozess und in der Regel ein geringerer Durchsuchungsaufwand, der weniger CPU-Zeit für die Platzzuordnung erfordert. Dies kann bei großen Buffer Pools von Bedeutung sein.

Der Nachteil von `METHOD=N` ist, dass die Objekte weniger sorgfältig für die Entfernung aus dem Buffer Pool ausgewählt werden. Um einen Anstieg der Adabas-Ein-/Ausgaben für das Zurückladen entfernter Objekte zu vermeiden, empfehlen wir, `METHOD=N` in Kombination mit der Buffer Pool Cache-Funktion zu verwenden.

### Lokaler Buffer Pool

Mit Natural, wie es auf dem Installationsmedium mitgeliefert wird, betreiben Sie einen *lokalen* Buffer Pool. Dabei handelt es sich um einen Buffer Pool-Bereich, der in derselben Partition (oder Region oder Adressraum) der jeweils benutzten Umgebung zugeordnet wird.

In einer Batch- oder TSO-Umgebung hat beispielsweise jeder Benutzer seinen eigenen lokalen Buffer Pool. In einer TP-Monitorumgebung wie Com-plete, CICS oder IMS TM gibt es einen Buffer Pool pro TP-Monitor, aus dem alle TP-Benutzer ausführen.

### Globaler Buffer Pool

In einer z/OS-Umgebung wird ein globaler Buffer Pool aus dem CSA- oder ECSA-Speicher zugewiesen. In einer solchen Umgebung können alle TSO-Benutzer, Batch-Benutzer und TP-Monitor-Benutzer aus einem gemeinsamen globalen Bereich ausführen.

Weitere Informationen über den globalen Buffer Pool finden Sie unter ***Globaler Natural Buffer Pool***.

## Buffer Pool Cache (BPC)

Dieser Abschnitt gilt für globale und lokale Buffer Pools mit `TYPE=NAT`.

Der Buffer Pool Cache ist in Verbindung mit globalen und lokalen Buffer Pools verfügbar. Er wird nur für Natural-Objekte (Programme, Subprogramme, Maps usw.) verwendet.

Wenn ein Buffer Pool nicht groß genug ist, um alle von den verschiedenen Benutzern angeforderten Objekte aufzunehmen, werden spezielle Überlastungsstrategien verwendet, um vorhandene Objekte durch angeforderte Objekte zu ersetzen. Die Anzahl der Überlastungssituationen steht in direktem Zusammenhang mit der Effizienz des Buffer Pool. Der beste und effizienteste Weg, die unerwünschten Überlastungen zu reduzieren und damit die Leistung des Buffer Pool zu verbessern, besteht darin, ihn einfach zu vergrößern.

Diese Möglichkeit ist jedoch bei den meisten Kundenstandorten aufgrund des Fehlens von verfügbarem Speicher im primären Adressraum und/oder der z/OS (E)CSA nicht anwendbar.

Um die oben beschriebene Situation zu verbessern, wird ein Buffer Pool Cache verwendet. Der Hauptzweck dieses Lösungsansatzes besteht darin, den Verlust aller Objekte zu verhindern, die aufgrund von situationsbedingter Knappheit an Buffer Pool-Speicher aus dem Buffer Pool gelöscht wurden. Das bedeutet, dass das Löschen eines Objekts zu einem Auslagern in den Buffer Pool Cache führt. Der angestrebte Nutzen dieser Funktion ist eine Verringerung der für das Laden von Objekten verwendeten Datenbankaufrufe und folglich eine Performance-Verbesserung.

**Anmerkung zu globalen Buffer Pools:**
Der Buffer Pool Cache-Bereich wird als Datenbereich oder als gemeinsames 64-Bit-Speicherobjekt „oberhalb der Grenze“ zugewiesen.
Wenn ein Datenbereich für einen Buffer Pool angelegt wird (durch Angabe von `C64=N`), wird die Eigentümerrolle der Ersteller-Task zugewiesen. Wenn die Ersteller-Task beendet wird, löscht das System den Datenbereich automatisch. Daher bleibt eine Ersteller-Task mindestens so lange aktiv, wie der Buffer Pool Cache, dessen Eigentümer diese Task ist, verwendet wird, auch wenn der für `RESIDENT` angegebene Wert `N` ist.
Wenn ein Speicherobjekt für einen Buffer Pool angelegt wird (durch Angabe von `C64=Y`), wird die Eigentümerschaft dem System zugewiesen (nicht der Ersteller-Task). Aus diesem Grund wird das Speicherobjekt nicht gelöscht, wenn die Ersteller-Task beendet wird. Wenn Sie die Ersteller-Task nach der Ausführung ihrer Funktion aktiv lassen wollen, müssen Sie `RESIDENT=Y` angeben.

**Anmerkung zu lokalen Buffer Pools:** (nur z/OS, nicht für Com-plete und nicht für IMS TM)
Der Buffer Pool Cache wird in einem Datenbereich oder in einem Speicherobjekt „oberhalb der Grenze“ zugeordnet, d. h. im 64-Bit-Speicher (nur z/OS). Wenn ein Datenbereich oder ein Speicherobjekt für einen Buffer Pool angelegt wird (siehe Profilparameter `BPCSIZE` und `BPC64`), wird die Eigentümerschaft der Ersteller-Task zugewiesen. Wenn diese Task beendet wird, löscht das System automatisch den Datenbereich oder das Speicherobjekt.

# Buffer Pool-Überwachung und -Verwaltung

Das Natural-Dienstprogramm SYSBPM liefert statistische Informationen über den aktuellen Status des Buffer Pool. Mit SYSBPM können Sie den Buffer Pool auch an Ihre Anforderungen anpassen.

Die folgenden Themen werden behandelt:

- Preload-Liste
- Sperrliste
- Propagation von Buffer Pool-Änderungen
- Performance-Aspekte

## Preload-Liste

Eine Preload-Liste ist eine Liste von Objekten, die in den Buffer Pool geladen werden und dort als resident verbleiben. Wenn ein Benutzer ein solches Objekt zur Ausführung anfordert, befindet es sich immer schon im Buffer Pool und braucht nicht aus der Systemdatei geladen zu werden.

Dies kann die Performance verbessern, eine Fragmentierung des Buffer Pools vermeiden oder sicherstellen, dass zentrale Fehlertransaktionen immer verfügbar sind, selbst dann wenn die Datenbank, die die Systemdatei enthält, nicht aktiv ist.

Zu Beginn der Natural-Sitzung prüft Natural, welche Objekte aus der Preload-Liste sich schon im Buffer Pool befinden. Diejenigen, die sich nicht in der Liste befinden, werden dann aus der Systemdatei in den Buffer Pool geladen. Diese Prüfung und das Laden werden auch durchgeführt, wenn der Buffer Pool mit Hilfe des Dienstprogramms SYSBPM verbunden, erneut verbunden und neu initialisiert wird. Wenn ein globaler Buffer Pool durch ein `REFRESH`-Kommando neu initialisiert wird, findet keine Prüfung bei bestehenden Natural-Sitzungen statt. Das heißt, solange keine neue Natural-Sitzung gestartet wird, die auf diesen Buffer Pool zugreift, werden die auf der Preload-Liste stehenden Objekte nicht geladen.

Das Laden der Preload-Liste wird nicht serialisiert. Das heißt, wenn mehrere Natural-Sitzungen gleichzeitig gestartet werden, versucht jede Sitzung, alle in der Preload-Liste genannten Objekte in den Buffer Pool zu laden. Dies kann zu doppelten Einträgen desselben Natural-Objekts im Buffer Pool führen (siehe auch Hinweis unten).

Eine Preload-Liste wird durch ihren Namen identifiziert und einem bestimmten Buffer Pool zugeordnet, indem ihr Name als Startup-Parameter (für einen globalen Buffer Pool) oder im `NTBPI`-Makro (für einen lokalen Buffer Pool) angegeben wird. So kann für jeden Buffer Pool eine andere Preload-Liste definiert werden oder die gleiche Preload-Liste kann für verschiedene Buffer Pools verwendet werden.

Wenn die angegebene Preload-Liste nicht gefunden werden kann oder wenn in der Preload-Liste enthaltene Objekte nicht geladen werden können, gibt Natural bei der Sitzungsinitialisierung eine

entsprechende Warnmeldung aus. In jedem Fall wird der Preloading-Vorgang bei jeder nachfolgenden Sitzungsinitialisierung wiederholt.

Da die Objekte in der Preload-Liste als erste geladen werden, befinden sie sich am Anfang des Buffer Pool (es sei denn, beim ersten Preloading konnten nicht alle Objekte geladen werden; in diesem Fall können sich die Objekte an einer beliebigen Stelle des Buffer Pool befinden).

Um Preload-Listen zu pflegen, können Sie das Dienstprogramm SYSBPM verwenden, siehe *Preload-Liste verwalten* in der *SYSBPM Utility*-Dokumentation

**Tipp:** Um Probleme mit beim Laden der Objekte in einer Preload-Liste durch Benutzersitzungen zu vermeiden, wird folgende Vorgehensweise empfohlen:

- **Bei einem globalen Buffer Pool:**
  Starten Sie unmittelbar nach der Zuordnung oder Aktualisierung des globalen Buffer Pool eine Natural-Batch-Sitzung, die auf den globalen Buffer Pool zugreift und ein `FIN`-Kommando ausführt.
- **Bei einem lokalen Buffer Pool unter CICS und Com-plete:**
  Starten Sie während des Startups des TP-Systems eine asynchrone Natural-Sitzung, die auf den lokalen Buffer Pool zugreift, und legen Sie ein `FIN`-Kommando auf den Natural-Stack. Stellen Sie sicher, dass diese Natural-Sitzung den Namen der Preload-Liste in Ihrem `NTBPI`-Makro referenziert.

### Sperrliste

Um zu verhindern, dass ein Natural-Objekt ausgeführt wird, können Sie es auf eine „schwarze Liste“ (Blacklist) setzen: Das Objekt kann dann nicht in den Buffer Pool geladen werden. Und wenn es sich bereits im Buffer Pool befindet, kann es nicht ausgeführt werden. Ein Benutzer, der die Ausführung eines solchen Objekts anfordert, erhält dann eine entsprechende Fehlermeldung.

Sie können nicht nur einzelne Objekte, sondern auch ganze Libraries auf die Sperrliste setzen.

**Beispiele**

- Die Sperrliste kann nützlich sein, wenn Sie ein Upgrade einer Natural-Anwendung durchführen und nicht möchten, dass die Benutzer mit dieser Anwendung weiterarbeiten, bis Sie das Upgrade abgeschlossen haben. Ohne die Sperrliste könnte ein Benutzer ein neues Modul ausführen, das wiederum ein altes Modul aufruft - was zu einem abnormalen Abbruch der Natural-Sitzung führen könnte. Mit der Sperrliste wird der Benutzer eine Meldung erhalten, dass das angeforderte Objekt derzeit nicht ausgeführt werden kann, und kann dann seine Natural-Sitzung fortsetzen.
- Performance-Aspekte können ein weiterer Grund für die Verwendung der Sperrliste sein, um zu verhindern, dass bestimmte ressourcenintensive Objekte in einer bestimmten Umgebung ausgeführt werden.

- Sie können die Sperrliste verwenden, um die Ausführung von Testprogrammen in einer Produktionsumgebung zu verhindern.

Zum Verwalten der Sperrliste können Sie das Dienstprogramm SYSBPM verwenden, siehe *Sperrliste verwalten* in der *SYSBPM Utility*-Dokumentation.

### Propagation von Buffer Pool-Änderungen

**Anmerkung:** Unter z/OS ist die Propagation von Buffer Pool Änderungen immer auf das Natural-Subsystem beschränkt, in dem die Änderung aufgetreten ist (Details zum Natural-Subsystem siehe ***Natural-Subsystem***). Daher bedeutet „alle globalen Buffer Pools" in diesem Zusammenhang „alle globalen Buffer Pools innerhalb desselben Subsystems".

In einigen Umgebungen ist es wichtig, dass Änderungen, die in einem (lokalen oder globalen) Buffer Pool stattfinden, auch an alle anderen globalen Buffer Pools propagiert werden, d.h. dieselben Änderungen werden automatisch auch in den anderen globalen Buffer Pools vorgenommen, um die Konsistenz der Buffer Pools und der verwendeten Natural-Anwendungen zu gewährleisten. Dies ist in einer z/OS Parallel Sysplex-Umgebung besonders wichtig.

Wenn beispielsweise ein Natural-Programm in einem z/OS-Image neu katalogisiert wird, führt die Propagation dazu, dass das Programm aus allen anderen globalen Buffer Pools in der z/OS-Parallel-Sysplex-Umgebung gelöscht wird, so dass seine neue Version aus der Systemdatei geladen werden muss, wenn das Programm erneut ausgeführt werden soll.

Die folgenden Änderungen werden propagiert:

- ein Objekt wird aus dem Buffer Pool gelöscht,
- die Sperrliste des Buffer Pool wird geändert,
- der Buffer Pool wird re-initialisiert.

Änderungen können an alle anderen globalen Buffer Pools im aktuellen z/OS-Image oder in der gesamten z/OS Parallel Sysplex-Umgebung oder an alle anderen globalen Buffer Pools mit demselben Namen innerhalb der z/OS Parallel Sysplex-Umgebung propagiert werden.

Die Propagation wirkt sich nicht auf die Objekte in einem anderen globalen Buffer Pool aus, die als resident in diesem Buffer Pool definiert sind.

Die Aktivierung der Propagation von Änderungen und die Steuerung ihres Geltungsbereichs erfolgt durch den Natural-Profilparameter `BPPROP`.

**Anmerkung:** Da die Weiterverbreitung asynchron erfolgt und ein Objekt erst aus einem Buffer Pool gelöscht wird, wenn es nicht mehr verwendet wird, kann es einige Zeit dauern, bis die aktuelle Version eines Objekts in allen Buffer Pools verfügbar ist.

Die Propagation von Änderungen an andere *lokale* Buffer Pools ist nicht möglich.

### Performance-Aspekte

Allgemeine Hinweise zu leistungsbezogenen Fragen bezüglich Buffer Pool und BP Cache finden Sie unter *Performance-Aspekte* in der Natural *SYSBPM Utility*-Dokumentation.

## Globaler Natural Buffer Pool

Der Natural Global Buffer Pool ist eine optionale Natural-Komponente, die für das Betriebssystem z/OS zur Verfügung steht. Weitere Informationen siehe ***Global Buffer Pool unter z/OS***.

Folgende Themen werden in diesem Abschnitt behandelt:

- Verwendete Profilparameter
- Vorgehensweise beim Öffnen/Schließen eines Buffer Pool

### Verwendete Profilparameter

Die folgenden Natural-Profilparameter werden verwendet, um einen globalen Buffer Pool einzurichten:

| | |
|---|---|
| `BPNAME` | Gibt den Namen des globalen Buffer Pool an (siehe `BPNAME`). `BPNAME=' '` (leer) wird verwendet, um eine Verbindung mit dem *lokalen* Buffer Pool herzustellen. |
| `SUBSID` | Gibt die Kennung des zu verwendenden Natural-Subsystems an (siehe Profilparameter `SUBSID`). Während des Natural-Startvorgangs versucht Natural, den globalen Buffer Pool anhand dieser Parameter zu finden. |

Während des Natural-Startvorgangs versucht Natural, den globalen Buffer Pool anhand dieser Parameter zu finden.

### Vorgehensweise beim Öffnen/Schließen eines Buffer Pool

Mit dem Makro `NTBPI` im Natural-Parametermodul oder mit dem entsprechenden Profilparameter `BPI` können Sie mehr als einen Buffer Pool definieren.

Bei der Sitzungsinitialisierung versucht Natural, eine Verbindung mit dem als ersten definierten Buffer Pool herzustellen. Wenn dies fehlschlägt, versucht Natural, eine Verbindung zu dem als zweiten definierten Buffer Pool herzustellen. Schlägt auch dies fehl, wird es beim nächsten definierten Buffer Pool versucht usw. Wenn ein Versuch, eine Verbindung zu einem Buffer Pool aufzubauen, fehlschlägt, gibt Natural eine entsprechende Meldung aus.

Die gleiche Vorgehensweise gilt, wenn ein Buffer Pool beendet wird:

Wenn Sie den aktuell verbundenen Buffer Pool schließen, während eine Natural-Sitzung noch aktiv ist, versucht Natural, eine Verbindung zu einem anderen Buffer Pool herzustellen (in der Reihenfolge, in der diese definiert sind) und die Sitzung fortzusetzen.

Der Natural-Administrator kann also einen globalen Buffer Pool schließen, ohne alle aktiven Natural-Sitzungen beenden zu müssen.

# 19 Natural Global Buffer Pool under z/OS


- Verwendung eines Natural Global Buffer Pool ..... 168
- Voraussetzungen ..... 168
- Betrieb eines Natural Global Buffer Pool (GBP) ..... 168
- GBP-Manager-Parametermodul ..... 170
- GBP-Betriebsfunktionen ..... 171
- Funktionsparameter des globalen Buffer Pools ..... 173
- Beispiele für NATBUFFER-Angaben ..... 181
- Beispiele für NATGBPvr-Ausführungsjobs ..... 181
- Lokalisierung ..... 183
- Meldungen ..... 184

Dieses Kapitel beschreibt Zweck und Verwendung eines Natural Global Buffer Pool (GBP) unter dem Betriebssystem z/OS.

## Verwendung eines Natural Global Buffer Pool

### Zweck

Der globale Natural Global Buffer Pool ist ein Speichersegment, das aus dem z/OS Extended Common System Area (ECSA) oberhalb von 16 MB (oder, falls gewünscht, aus dem CSA-Speicher darunter) zugewiesen wird und von Natural zum Laden und Ausführen von Natural-Programmen genutzt wird.

### Vorteile

Mit einem globalen Buffer Pool teilen sich mehrere Natural-Sitzungen unter verschiedenen TP-Monitoren (mehrere Kopien von CICS, TSO, IMS TM usw.) und/oder in mehreren Batch-Sitzungen denselben Bereich. Dadurch wird weniger Speicherplatz benötigt, als für einen lokalen Buffer Pool in jeder einzelnen Umgebung erforderlich wäre.

## Voraussetzungen

Die folgenden Voraussetzungen müssen erfüllt sein, damit Sie einen globalen Buffer Pool verwenden können:

1. Das Modul `NATGBPvr` muss in eine APF-Library (Authorized Program Facility) verlinkt worden sein. Siehe den entsprechenden Schritt in *Natural auf z/OS installieren* in der *Installation für z/OS*-Dokumentation.
2. Der globale Buffer Pool muss gestartet worden sein. Siehe den entsprechenden Schritt in *Installing Natural on z/OS* in der *Installation for z/OS*-Dokumentation.

## Betrieb eines Natural Global Buffer Pool (GBP)

Ein globaler Buffer Pool wird unter Verwendung des Programms `NATGBPvr` betrieben, das in einer APF-Bibliothek (Authorized Program Facility) ausgeführt werden muss.

Die folgenden Themen werden in diesem Abschnitt behandelt:

- Zuordnung des Natural GBP
- Einrichten des Natural GBP

- Starten des Natural GBP-Betriebsprogramms
- Anhalten des Natural GBP-Betriebsprogramms

**Anmerkung:** Im folgenden Dokument steht *vrs* oder *vr* für die jeweilige Version des Produkts. Informationen zu Produktversionen finden Sie unter Version im *Glossar*.

### Zuordnung des Natural GBP

Wenn der z/OS-Parameter `ALLOWUSERKEYCSA(YES)` explizit in `SYS1.PARMLIB(DIAGxx)` angegeben wurde, wird ein globaler Natural Buffer Pool im User Key zugewiesen, so dass Natural-Sitzungen, die auf einen globalen Buffer Pool zugreifen, Schreibrechte für diesen Buffer Pool haben.

Wenn `ALLOWUSERKEYCSA(NO)` wirksam ist, wird ein globaler Natural Buffer Pool im System Key zugewiesen. Dadurch haben Natural-Sitzungen, die auf einen globalen Buffer Pool zugreifen, keine Schreibberechtigung für diesen Buffer Pool. Diese Natural-Sitzungen rufen den Authorized Services Manager (ASM) auf, um alle Buffer Pool-Funktionen auszuführen. Folglich ist die Installation des ASM zwingend erforderlich. Der ASM wird nicht nur aufgerufen, um ein Natural-Objekt in den Buffer Pool zu laden, sondern auch, um den Nutzungszähler (Use Count) eines Natural-Objekts zu verwalten, wenn die Ausführung dieses Natural-Objekts gestartet oder beendet wird. Die Aufrufe an den Authorized Services Manager erhöhen den Ressourcenverbrauch von Natural. Der damit verbundene Aufwand ist schwer vorhersehbar und hängt vom Anwendungsprofil ab (Verhältnis von Programmaufrufen zur Programmausführungszeit).

### Einrichten des Natural GBP

Die vom Betriebsprogramm `NATGBPvr` bereitgestellten **GBP-Betriebsfunktionen** werden aktiviert, indem sie

- in einer Parameterkarte (`PARM=`) angegeben werden,
- aus einer Datei gelesen werden (siehe unten),
- oder durch das Operator-Kommando `MODIFY` geliefert werden.

`NATGBPvr` erwartet das erste Kommando im Parameterfeld (`PARM=`) des `EXEC`-Statements.

Sie können Folgendes eingeben:

- eine dieser **GBP-Betriebsfunktionen**
- oder einen Verweis auf eine Eingabedatei mit `CF=dd-name`, wobei `dd-name` für einen in der JCL definierten DD-Namen steht. Es werden nur Card Image-Dateien unterstützt, d.h. `RECFM=F,LRECL=80`, und nur die ersten 72 Bytes des Eingabedatensatzes werden beachtet. Jeder in der Eingabedatei enthaltene Satz stellt ein Kommando dar. Leere Sätze oder Sätze mit einem vorangestellten Stern (*) werden ignoriert. Eine Datei wird bis zum End-Of-File (EOF) verarbeitet.

  Beispiel: `PARM='CF=SYSIN1'`

Wenn das Parameterfeld nicht angegeben oder leer ist, werden die Kommandos standardmäßig aus der Datei `SYSIN` gelesen.

Es ist nur möglich, jeweils eine Funktion an der Konsole oder eine Funktion pro Zeile über die Kommandodatei einzugeben, ansonsten wird eine Fehlermeldung ausgegeben.

Jedes Kommando, das von der Parameterkarte, aus der Datei oder von der Bedienerkonsole empfangen wird, wird auf der Bedienerkonsole angezeigt.

### Starten des Natural GBP-Betriebsprogramms

Um das Programm `NATGBPvr` zu starten, müssen Sie entweder eine Started Task starten oder einen Job übergeben, der `NATGBPvr` ausführt.

**Wichtig:** Um sicherzustellen, dass der globale Buffer Pool nach einem Systemausfall erhalten bleibt, sollte der globale Buffer Pool automatisch während des Maschinen-IPL gestartet werden.

### Anhalten des Natural GBP-Betriebsprogramms

Nachdem alle Kommandos abgearbeitet sind, wird das Programm `NATGBPvr` beendet, es sei denn, dass

- `RESIDENT=Y` angegeben wurde
- oder ein Buffer Pool mit einem Cache erstellt wurde.

`NATGBPvr` gibt einen der folgenden Condition Codes zurück:

| Condition Code | Erläuterung |
|---|---|
| 0 | Alle Funktionen wurden erfolgreich ausgeführt. |
| 20 | Ein Fehler ist aufgetreten. Details siehe Meldungsprotokoll.<br><br>Siehe auch den Funktionsparameter `CC` des globalen Buffer Pool. |

## GBP-Manager-Parametermodul

Das GBP-Manager-Parametermodul `NATGBPRM` wird verwendet, um globale Verarbeitungsoptionen festzulegen, die für alle Funktionen und Buffer Pools gelten. Das globale Buffer Pool-Parametermodul wird in Quellcode- und Objektform ausgeliefert, wobei alle Standardwerte gesetzt sind.

Der folgende Parameter ist verfügbar:

- UCTRAN - Unterstützung von Klein-/Gemischtschreibung

### UCTRAN - Unterstützung von Klein-/Gemischtschreibung

Dieser Parameter aktiviert oder deaktiviert die Unterstützung der Klein-/ Gemischtschreibung bei den globalen Buffer Pool-Meldungen.

| | |
|---|---|
| `UCTRAN=NO` | Die Unterstützung der Klein-/Gemischtschreibung ist vollständig aktiviert.<br><br>Dies ist der Standardwert. |
| `UCTRAN=YES` | Alle globalen Buffer Pool-Meldungen werden in Großbuchstaben ausgegeben. |

## GBP-Betriebsfunktionen

Die folgenden Funktionen sind verfügbar:

- HELP - Übersicht über die verfügbare Syntax anzeigen
- ADDCACHE - Cache für einen vorhandenen globalen Buffer Pool zuordnen
- CREATE - Globalen Buffer Pool anlegen
- DELCACHE - Cache eines globalen Buffer Pool freigeben
- FSHUT - Globalen Buffer Pool abschalten
- GLOBALS - Globale Parametereinstellungen anzeigen
- LISTCACHE - Alle globalen Buffer Pool Caches eines Jobs auflisten
- NOP - Kein Betrieb
- REFRESH - Globalen Buffer Pool neu initialisieren
- SHOWBP - Vorhandene Buffer Pools anzeigen
- TERMINATE - GBP-Betriebsprogramm beenden
- ZAPS - Auf GBP angewendete Zaps anzeigen

**Anmerkung:** Wenn keine Funktion angegeben ist, wird `CREATE` angenommen, wenn der Profilparameter `BPNAME` angegeben ist, ansonsten wird `NOP` angenommen.

### HELP - Übersicht über die verfügbare Syntax anzeigen

Diese Funktion gibt eine Liste der verfügbaren Syntax-Kommandos und ggf. die Standardwerte der Funktionsparameter aus.

### ADDCACHE - Cache für einen vorhandenen globalen Buffer Pool zuordnen

Mit dieser Funktion wird Cache-Speicher bei einem vorhandenen globalen Buffer Pool hinzugefügt.

### CREATE - Globalen Buffer Pool anlegen

Mit dieser Funktion wird ein globaler Buffer Pool mit den angegebenen Parametern angelegt.

### DELCACHE - Cache eines globalen Buffer Pool freigeben

Mit dieser Funktion wird der Cache-Speicher eines globalen Buffer Pool entfernt, ohne dass der Buffer Pool selbst heruntergefahren wird.

**Anmerkung:** Ein Datenraum-Cache kann nur von der Task gelöscht werden, der er zugehört. Ein Speicherobjekt-Cache „oberhalb der Grenze“ kann von jeder Task gelöscht werden.

### FSHUT - Globalen Buffer Pool abschalten

Der globale Buffer Pool wird heruntergefahren und der Speicherbereich, einschließlich Buffer Pool- und Cache-Speicher, wird freigegeben.

Wenn sich keine aktiven Objekte im Puffer Pool befinden, wird `FSHUT` sofort ausgeführt.

Wenn sich noch aktive Objekte im Buffer Pool befinden, wird dies dem Bediener angezeigt. Je nach Einstellung des Parameters `CONFIRM` wird der Bediener aufgefordert, eine Bestätigung einzugeben, oder `FSHUT` wird sofort ausgeführt.

### GLOBALS - Globale Parametereinstellungen anzeigen

Diese Funktion zeigt alle globalen Parametereinstellungen an, d.h. Parameter, die nicht nur für das Statement gelten, für das sie angegeben wurden.

Darüber hinaus wird der Storage Key (Speicherschlüssel) des/der globalen Buffer Pool(s) angezeigt.

### LISTCACHE - Alle globalen Buffer Pool Caches eines Jobs auflisten

Diese Funktion listet alle globalen Buffer Pool Caches auf, die zurzeit zu dem Job gehören.

### NOP - Kein Betrieb

Dieser Funktionscode kann verwendet werden, um globale Parameter zu setzen. Er führt keine Buffer Pool-Operation durch.

### REFRESH - Globalen Buffer Pool neu initialisieren

Mit dem Kommando `REFRESH` können Sie einen bereits aktiven Buffer Pool neu initialisieren. Da keine Speicherzuordnung stattfindet, bleiben die Buffer Pool-Größe und der Speicherort (oberhalb oder unterhalb von 16 MB) unverändert. Es ist jedoch möglich, die Größe der Textblöcke zu ändern (siehe Parameter `NATBUFFER`).

Sie sollten diese Funktion nur benutzen, wenn das Feld **Current Use Count** (siehe *Felder für Buffer-Pool-Objekte* in *Verzeichnisinformationen anzeigen*) gleich Null ist (siehe Warnung unten) oder wenn der Buffer Pool zerstört wurde.

**Vorsicht:** Wenn Sie den Buffer Pool neu initialisieren, während Natural-Objekte von aktiven Sitzungen in diesem Buffer Pool ausgeführt werden, sind die Ergebnisse der aktiven Sitzungen unvorhersehbar und Natural kann sogar abbrechen.

### SHOWBP - Vorhandene Buffer Pools anzeigen

Zeigt alle derzeit existierenden Buffer Pools an.

### TERMINATE - GBP-Betriebsprogramm beenden

Das GBP-Betriebsprogramm wird beendet. Diese Beendigung hat *keine* Auswirkungen auf einen aktiven globalen Buffer Pool.

### ZAPS - Auf GBP angewendete Zaps anzeigen

Zeigt alle Zaps an, die auf das globale Buffer Pool-Betriebsprogramm angewendet wurden.

## Funktionsparameter des globalen Buffer Pools

Die Funktionen des Natural-GBP-Betriebsprogramms können mit Hilfe von Parametern gesteuert werden. Diese Parameter können in beliebiger Reihenfolge angegeben werden. Sie können so abgekürzt werden, so dass sie noch eindeutig sind.

**Anmerkung:** Wenn Sie mehrere globale Buffer Pools mit zugehörigem Cache starten wollen, empfiehlt es sich, einen einzigen Job oder (nur unter z/OS) eine einzige Started Task zu verwenden und die verschiedenen `CREATE`-Kommandos in einem Eingabe-Dataset bereitzustellen. Siehe ***Beispiel 4*** im Abschnitt *Natural Global Buffer Pool unter z/OS*.

Die folgenden Funktionsparameter sind verfügbar:

| Parameter | Angabe |
|---|---|
| BPNAME | Name des globalen Buffer Pool |
| BPLIST | Name der Preload-Liste |
| BPCSIZE | Puffer Pool Cache-Größe |
| C64 | Typ des Buffer Pool Cache-Speichers |
| CC | Condition Code zählen |
| CONFIRM | Bestätigung |
| IDLE | Wartezeit vor der Prüfung |
| METHOD | Suchalgorithmus für die Zuweisung von Speicherplatz im Buffer Pool |
| NATBUFFER | Puffer-Größe, Modus, Textblockgröße |
| RESIDENT | Verhalten nach Funktionsausführung |
| SUBSID | Kennung des Natural-Subsystems |
| TYPE | Typ des Buffer Pool |

## BPNAME - Name of Global Buffer Pool

BPNAME=*value* ist erforderlich (außer für die Funktion TERMINATE). Er gibt den Namen des zu erstellenden globalen Buffer Pool an.

| Wert | Erläuterung |
|---|---|
| 8 Bytes | Der Name des globalen Buffer Pool.<br><br>**Anmerkung:** Wenn der angegebene Name kürzer als 8 Bytes ist, werden Leerzeichen angehängt. |
| * | Für die Funktionen DELCACHE und FSHUT können Sie einen Stern (*) als Wert von angeben.<br><br>FSHUT schaltet alle Buffer Pools für die angegebenen Natural-Subsysteme ab. Wenn der Subparameter CONFIRM auf Y gesetzt ist, müssen Sie zusätzlich eine Bestätigung angeben.<br><br>DELCACHE löscht alle, der Task zugehörigen Datenraum-Caches, aus dem angegebenen Natural-Subsystem. Um Caches zu löschen, die sich im Speicher „oberhalb der Grenze“ befinden, müssen Sie auch einen vollständigen BPNAME und eine SUBSID angeben. |

### BPLIST - Name of Preload List

`BPLIST=value` gibt den Namen der Preload-Liste an.

| Wert | Erläuterung |
|---|---|
| 8 Bytes | Der Name der Preload-Liste.<br><br>**Anmerkung:** Wenn der angegebene Name kürzer als 8 Bytes ist, werden Leerzeichen angehängt. |

### BPCSIZE - Buffer Pool Cache-Größe

`BPCSIZE=value` gibt die Speichermenge an, die für die Zuordnung eines Buffer Pool Cache verwendet wird.

| Wert | Erläuterung |
|---|---|
| `100 - 2097148` | Wenn der Subparameter `C64=N` ist, ist dieser Wert die Menge des zugewiesenen Speichers (in KB) für einen Datenbereich für den Puffer Pool Cache. Der angegebene Wert wird auf die nächste 4-KB-Grenze gerundet. |
| `100 - 58720256`<br><br>(max `57344` MB) | Wenn der Subparameter `C64=Y` ist, ist dieser Wert die Menge des zugeordneten Speichers (in KB) für ein Speicherobjekt „oberhalb der Grenze“ für den Buffer Pool Cache. Der angegebene Wert wird auf die nächste 1-MB-Grenze gerundet. |

**Anmerkungen:**

1. Die Cache-Größe kann auch in Einheiten von MB oder GB angegeben werden, z. B. durch Angabe von 10M für 10 MB.
2. Wird der Parameter `BPCSIZE` weggelassen (oder auf Null gesetzt), dann wird der Buffer Pool nicht mit einem Cache versorgt.
3. Ein Cache wird nur für Buffer Pools mit `TYPE=NAT` unterstützt.
4. Der Subparameter `C64` entscheidet, ob der Cache im Datenbereich oder im gemeinsamen 64-Bit-Speicher „oberhalb der Grenze“ angelegt wird.

### C64 - Type of Buffer Pool Cache Storage

`C64=value` bestimmt den Typ des Speichers für den Buffer Pool Cache. Die folgenden Werte sind möglich:

| Wert | Erläuterung |
|---|---|
| Y | Der Speicher für den Buffer Pool Cache ist ein „oberhalb der Grenze“ liegendes Speicherobjekt (im 64-Bit-Speicher). |
| N | Der Speicher für den Buffer Pool Cache ist ein Datenbereich. Dies ist der Standardwert. |

**Anmerkung:** Dieser Parameter ist nur anwendbar, wenn `BPCSIZE` angegeben ist.

### CC - Condition Code zählen

`CC=value` legt fest, ob ein Condition Code ignoriert wird, wenn er von einem vom globalen Puffer Pool Manager ausgeführten Kommando zurückgegeben wird.

| Wert | Erläuterung |
|---|---|
| Y | Der nach der Kommandoausführung zurückgegebene Condition Code wird für den Condition Code des `NATGBPvr`-Jobs gezählt. Dies ist der Standardwert.<br><br>Dies ist der Standardwert. |
| N | Der nach der Kommandoausführung zurückgegebene Condition Code wird ignoriert.<br><br>Dies kann zu einem Job Response Code von Null führen, obwohl die Ausführung des Kommandos fehlgeschlagen ist. |

**Anmerkung:** Dieser Parameter ist für alle Kommandos gültig.

Beispiel für die Kommandoausführung:

Der globale Buffer Pool `NATGBP1` wird mit der folgenden Kommandofolge angehalten und neu gestartet:

```
FSHUT BPN=NATGBP1,S=NAT92,CONFIRM=N,CC=N
CREATE BPN=NATGBP1,S=NAT92,N=(1024),M=S,BPC=4096,I=60
```

Das Kommando `FSHUT` gibt normalerweise den Condition Code 20 zurück, wenn es ausgeführt wird und der Buffer Pool nicht aktiv ist. Wenn jedoch `CC=N` gesetzt ist, wird jeder Condition Code ignoriert. In diesem Fall wird nur dann ein Job Response Code größer als Null zurückgegeben, wenn das folgende `CREATE`-Kommando fehlschlägt.

### CONFIRM - FSHUT-Bestätigung

CONFIRM=*value* steuert das FSHUT-Verhalten, wenn sich noch aktive Objekte im Buffer Pool befinden.

| Wert | Erläuterung |
|---|---|
| Y | Eine Bestätigung für die FSHUT-Funktion ist vom Bediener erforderlich. Der Bediener kann entscheiden, ob er die FSHUT-Funktion abbrechen oder erzwingen möchte.<br><br>Dies ist der Standardwert. |
| N | FSHUT wird ohne Bediener-Interaktion erzwungen. |

**Anmerkung:** Dieser Parameter ist nur für das FSHUT-Kommando gültig, mit dem er angegeben wurde, d. h. CONFIRM muss mit jedem FSHUT-Parameter angegeben werden, und er gilt nicht für nachfolgende FSHUT-Kommandos.

### IDLE - Wartezeit vor der Prüfung

IDLE=*value* wird ignoriert, wenn die Task keinen Buffer Pool Cache besitzt.

| Wert | Erläuterung |
|---|---|
| Numerisch | Die Anzahl der Sekunden, die vergehen, bevor das GBP-Betriebsprogramm für jeden Buffer Pool Cache prüft, ob der zugehörige Buffer Pool noch aktiv ist. Wenn nicht, wird dieser Buffer Pool Cache freigegeben. Wenn der letzte Buffer Pool Cache, der der Task gehört, freigegeben wurde, wird die Task beendet, es sei denn, es wurde RESIDENT=Y angegeben. |
| 60 | Dies ist der Standardwert. |

**Anmerkungen:**

1. IDLE ist ein „globaler“ Parameter. Einmal angegeben, gilt IDLE auch für nachfolgende Kommandos, ohne dass Sie ihn erneut angeben müssen.
2. Unter z/OS prüft das GBP-Betriebsprogramm den angegebenen IDLE-Zeitwert auch mit dem Timeout-Wert des Jobs: Der angegebene IDLE-Zeitwert kann IDLE intern reduzieren, um Timeout-Abbrüche zu verhindern (S322).

### METHOD - Suchalgorithmus für die Zuweisung von Speicherplatz im Buffer Pool

METHOD=*value* steuert, welcher Algorithmus für die Zuordnung von Speicherplatz im Natural Buffer Pool verwendet werden soll.

| Wert | Erläuterung |
|---|---|
| N | Gibt an, dass der nächste verfügbare ungenutzte oder freie Platz verwendet werden soll.<br><br>Die Suche nach dem nächsten freien Speicherplatz erfolgt anhand eines Zeigers auf Verzeichniseinträge, der sich in einem Umlaufverfahren (Wrap-around) bewegt. Diese Methode kann in Kombination mit einem Buffer Pool Cache verwendet werden.<br><br>Dies ist der Standardwert. |
| S | Gibt an, dass ein Auswahlverfahren für die Zuordnung von Speicherplatz verwendet werden soll.<br><br>Das Auswahlverfahren besteht aus dem Durchsuchen des gesamten Buffer Pool-Verzeichnisses und dem Vergleichen verschiedener Einträge, um den am besten geeigneten Eintrag zu finden. Diese Methode war früher als „Algorithmus 1+2“ bekannt. |

**Anmerkung:** Dieser Parameter ist nur für die Funktion `CREATE` gültig. Wenn Sie die Zuordnungsmethode ändern möchten, starten Sie den Buffer Pool neu.

### NATBUFFER - Buffer-Größe, Modus, Textblockgröße

`NATBUFFER=(size,mode,tsize)` gibt die Größe und den Modus des Buffer Pool und die Textblockgröße an.

| Syntax | Wert | Erläuterung |
|---|---|---|
| `NATBUFFER=(size,mode,tsize)` | *size* | Die Menge des zuzuordnenden Speichers (in KB). Die Pool-Größe kann auch in Einheiten von MB angegeben werden, z. B. durch Angabe von 10M für 10 MB.<br><br>Für den Natural Buffer Pool (`TYPE=NAT`)<br><br>■ beträgt die mögliche Größe in KB (`256` – `2097148`) oder in MB (`1M` – `2047M`).<br>■ Die Standardgröße ist 2 MB.<br><br>Für andere Buffer Pools<br><br>■ beträgt die mögliche Größe in KB (`100` – `2097148`) oder in MB (`1M` – `2047M`).<br>■ Die Standardgröße ist 256 KB.<br><br>Die angegebene Speichermenge wird immer auf ein Vielfaches von 4 KB aufgerundet.<br><br>Neben dem durch die Größe (*size*) angegebenen Speicherplatz wird eine Seite (4 KB) schreibgeschützter Speicher für Verwaltungszwecke zugeordnet. |

| Syntax | Wert | Erläuterung |
|---|---|---|
| | *mode* | bestimmt, ob der globale Buffer Pool oberhalb oder unterhalb 16 MB zugewiesen werden soll.<br><br>Mögliche Werte sind: XA = oberhalb (Standard), BL = unterhalb. |
| | *tsize* | bestimmt die Textblockgröße (in KB).<br><br>Mögliche Werte sind: 1, 2, 4, 8, 12 und 16. Der Standardwert ist 4. |
| *size*, *mode* und *tsize* müssen in der oben gezeigten Reihenfolge angegeben werden. | | |

**Anmerkung:** Wenn NATBUFFER nicht angegeben wird, werden die Standardwerte verwendet. Siehe auch ***Beispiele für NATBUFFER-Angaben***.

### RESIDENT - Verhalten nach Funktionsausführung

RESIDENT=*value* gibt das Verhalten des GBP-Betriebsprogramms nach der Ausführung der angegebenen Funktion an. Die folgenden Werte sind möglich:

| Wert | Erläuterung |
|---|---|
| Y | Das GBP-Betriebsprogramm bleibt nach dem Ausführen der angegebenen Funktion aktiv und wartet auf weitere Kommandos. Einmal festgelegt, gilt RESIDENT=Y auch für nachfolgende Kommandos, ohne dass Sie es erneut angeben müssen. (Um das GBP-Betriebsprogramm zu beenden, müssen Sie die Funktion TERMINATE benutzen.) |
| N | Das GBP-Betriebsprogramm wird nach der Ausführung der angegebenen Funktion beendet, wenn kein weiteres Kommando verfügbar ist. Wenn die Task einen Buffer Pool Cache besitzt, wird RESIDENT=N ignoriert und die Task wird nicht beendet. |
| A | Das GBP-Betriebsprogramm entscheidet automatisch, wie es sich nach Abarbeitung aller Kommandos verhalten soll. Es wird beendet, wenn<br><br>■ kein weiteres Kommando verfügbar ist und<br>■ kein Buffer Pool mit einem zugehörigen Cache existiert, der von dieser Task angelegt wurde.<br><br>Mit anderen Worten: Wenn die Task keinen Buffer Pool Cache besitzt, funktioniert RESIDENT=A genauso wie RESIDENT=N. Wenn die Task einen Buffer Pool Cache besitzt, funktioniert RESIDENT=A genauso wie RESIDENT=Y, schaltet aber automatisch auf RESIDENT=N um, wenn der letzte Buffer Pool, dessen zugehöriger Buffer Pool Cache im Besitz dieser Task war, beendet wurde.<br><br>Dies ist die Standardeinstellung. |

**Anmerkung:** RESIDENT ist ein „globaler“ Parameter. Einmal angegeben, gilt RESIDENT auch für nachfolgende Kommandos, bis er ausdrücklich angegeben/überschrieben wird.

### SUBSID - Kennung des Natural-Subsystems

SUBSID=*value* gibt die Kennung (ID) des Natural-Subsystems an.

| Wert | Erläuterung |
|---|---|
| 4 bytes | Die 4-Byte-ID des Natural-Subsystems.<br><br>Einmal angegeben, gilt SUBSID auch für nachfolgende Kommandos, ohne dass Sie sie erneut angeben müssen.<br><br>Der Standardwert ist NAT*v*, wobei *v* die erste Ziffer der aktuellen Natural-Version ist. |
| NAT*v* | Dies ist der Standardwert. *v* ist die erste Ziffer der aktuellen Natural-Version. |

**Anmerkungen:**

1. SUBSID ist ein „globaler“ Parameter, d.h. einmal angegeben, gilt SUBSID auch für nachfolgende Kommandos, bis er ausdrücklich angegeben/überschrieben wird.
2. Für die Funktionen DELCACHE, FSHUT und SHOWBP können Sie SUBSID=* angeben, um, unabhängig vom Subsystem, einen bestimmten Buffer Pool zu verarbeiten. Alternativ können Sie BPNAME=* angeben, um alle Buffer Pools aus allen Subsystemen zu verarbeiten. In diesem Fall fordert die Funktion FSHUT eine zusätzliche Bestätigung an, und zwar unabhängig vom Wert des Subparameters CONFIRM.
3. Weitere Informationen über das Natural-Subsystem finden Sie unter ***Natural-Subsystem (z/OS)***.

### TYPE - Typ des Buffer Pool

TYPE=*value* gibt den Typ des Puffer Pool an. Mögliche Werte sind:

| Wert | Erläuterung |
|---|---|
| NAT | Natural Buffer Pool (dies ist der Standardwert). |
| SORT | Sortier-Buffer Pool. |
| EDIT | Editor Buffer Pool. |
| MON | Monitor Buffer Pool. |
| RNM | Natural Review Buffer Pool. |

## Beispiele für NATBUFFER-Angaben

Die folgenden Beispiele beziehen sich auf den Parameter `NATBUFFER`, mit dem die Buffer-Größe, der Modus und die Textblockgröße eingestellt werden können, wobei der Parametername in seiner Kurzform (`N`) angegeben wird.

Beispiel 1: Um einen globalen Buffer Pool oberhalb 16 MB mit einer Größe von 1 MB und einer Textblockgröße von 1 KB zuzuordnen, geben Sie an:

```
N=(1000,,1)
```

oder

```
N=(1M,,1)
```

Beispiel 2: Um einen globalen Buffer Pool oberhalb 16 MB mit einer Größe von 10 MB und einer Textblockgröße von 4 KB zuzuordnen, geben Sie an:

```
N=(10000)
```

oder

```
N=(10M)
```

Beispiel 3: Um einen globalen Buffer Pool oberhalb 16 MB mit einer Größe von 256 KB und einer Textblockgröße von 4 KB zuzuordnen, geben Sie an:

```
N=(,,)
```

Dies ist gleichbedeutend mit dem Weglassen des `NATBUFFER`-Parameters, da dann die Standardwerte gelten.

## Beispiele für NATGBPvr-Ausführungsjobs

Die folgenden Abschnitte zeigen Beispiele für Batch-Jobs zur Erstellung und Beendigung eines globalen Buffer Pool.

In den folgenden Beispielen steht die Schreibweise *`vrs`* oder *`vr`* für die jeweilige Produktversion. Informationen zu Produktversionen finden Sie unter Version im *Glossar*.

**Beispiel 1:**

```
//GBPSTART JOB
//*
//* Starts a global buffer pool with the name NATvrGBP, a size of 1 MB and
//* a text block size of 4 KB. The global buffer pool is allocated above 16 MB.
//* The subsystem used is NATv.
//* After the allocation, the job GBPSTART terminates.
//*
//STEP EXEC PGM=NATGBPvr,PARM='BPN=NATvrGBP,N=(1M)'
//SETPLIB DD DISP=SHR,DSN=USER.APF.LINKLIB
```

**Beispiel 2:**

```
//GBPRES JOB
//*
//* Starts a global buffer pool with the name GBP, a default size of
//* 100 KB and a text block size of 1 KB. The global buffer pool is allocated
//* below 16 MB. The subsystem used is SAGS.
//* After the allocation, the job GBPRES will wait for further commands.
//* Further commands may be entered using the MODIFY operator command:
//* F GBPRES,command-string
//*
//STEP EXEC PGM=NATGBPvr,PARM='BPN=GBP,N=(,BL,1),S=SAGS,R=Y'
```

**Beispiel 3:**

```
//GBPSTOP
//*
//* Stops the global buffer pool GPB if it contains no active objects. If it
//* does contain active objects, the operator console will prompt for a reply.
//* Depending on the reply, the shutdown will be forced (Y) or aborted (N).
//* The subsystem used is NATv.
//*
//STEP EXEC PGM=NATGBPvr,PARM='FSHUT,BPN=GPB'
```

**Beispiel 4:**

```
//GBPSTRT2
//* Read commands from SYSIN1:
//*
//* Start two global buffer pools (subsystem ID Nvrs) with names
//*   NATGBP1 - size=1024KB and a cache with size 2048KB, and
//*   NATGBP2 - size=2048KB without cache.
//* Display all buffer pools of subsystem ID Nvrs.
//*
//* Note: The job does not terminate by itself, but stays resident and waits
//*       for operator commands because it owns the data space allocated for
//*       buffer pool NATGBP1.
```

```
//*
//* To shut down the buffer pools, send the operator command MODIFY with
//* parameter CF=SYSIN2 to execute the corresponding FSHUTs.
//*
//STEP EXEC PGM=NATGBPvr,PARM='CF=SYSIN1'
//SYSIN1  DD *
CREATE,BPN=NATGBP1,S=Nvrs,N=(1M),BPC=2M
CREATE,BPN=NATGBP2,S=Nvrs,N=(2M)
SHOWBP S=Nvrs
//SYSIN2 DD *
FSHUT,BPN=NATGBP1,S=Nvrs
FSHUT,BPN=NATGBP2,S=Nvrs
SHOWBP S=Nvrs
//*
```

## Lokalisierung

Das Modul NATGBPTX wird als Quellcode ausgeliefert. Es enthält alle Fehlermeldungen in Englisch in gemischter Schreibweise. Die Meldungen können bei Bedarf in andere Sprachen übersetzt werden. In diesem Fall muss das „neue“ Quellcode-Modul NATGBPTX assembliert und das Modul NATGBP*vr* neu verlinkt werden.

Um die globalen Buffer Pool-Meldungen einschließlich ihrer variablen Teile in Großbuchstaben auszugeben, muss das globale Buffer Pool-Parametermodul NATGBPRM mit dem **UCTRAN**-Parameter auf YES gesetzt und das Modul NATGBP*vr* muss neu verlinkt werden.

Um das Modul NATGBP*vr* neu zu verlinken, müssen Sie die folgende JCL verwenden:

```
//SYSLIN  DD *
SETCODE AC(1)
SETOPT PARM(REUS=RENT)
INCLUDE NATLIB(NATGBPMG)
INCLUDE SMALIB(NATGBPRM)
INCLUDE SMALIB(NATGBPTX)
INCLUDE NATLIB(NATBPMGR)
NAME NATGBPvr(R)
/*
```

# Meldungen

Siehe *Natural Global Buffer Pool Manager Messages* in der Natural *Messages and Codes*-Dokumentation

# VI Message Buffer Pool verwenden

# 20 Message Buffer Pool verwenden


- Zweck ..... 188
- Voraussetzungen ..... 188
- Betrieb des Message Buffer Pool ..... 189
- Beispiele für NATMBPvr-Ausführungsjobs ..... 190
- Message Buffer Pool-Betriebsfunktionen ..... 191
- Funktionsparameter ..... 192
- Meldungen ..... 194

Dieser Teil beschreibt die Verwendung des **Message Buffer Pool**, der unter z/OS verwendet werden kann.

Siehe auch folgende Themen in der *SYSBPM Utility*-Dokumentation:

- *Message Pool auswählen*
- *Message Buffer Pool-Statistiken anzeigen*

**Anmerkung:** In der SYSBPM Utility wird der Message Buffer Pool häufig auch verkürzt als **Message Pool** bezeichnet.

## Zweck

Der Message Buffer Pool ist ein Cache-Speicher, in dem die Natural-Systemmeldungen und die Benutzertexte gespeichert werden.

Bevor eine Fehlermeldung ausgegeben wird, prüft Natural zunächst, ob der entsprechende Meldungstext im Message Buffer Pool vorhanden ist. Wenn ja, wird dieser Text ausgegeben. Andernfalls wird die Fehlermeldung aus der Datenbank gelesen und im Message Buffer Pool gespeichert.

Der Message Buffer Pool ist nur als globaler Buffer Pool verfügbar. Seine Verwendung ist optional und wird durch den Natural-Profilparameter `BPI` oder das entsprechende Makro `NTBPI` gesteuert.

Wenn ein Message Buffer Pool verwendet wird, wird er in einem Datenbereich zugeordnet.

## Voraussetzungen

Die folgenden Voraussetzungen müssen erfüllt sein, wenn Sie den Message Buffer Pool verwenden wollen:

1. Das Modul `NATMBP`*`vr`* muss in eine APF-Bibliothek (Authorized Program Facility) verlinkt worden sein. Siehe den entsprechenden Schritt in *Natural auf z/OS installieren* in der *Installation für z/OS*-Dokumentation.
2. Der Message Buffer Pool muss angelegt und gestartet worden sein. Siehe den entsprechenden Schritt in *Natural auf z/OS installieren* in der *Installation für z/OS*-Dokumentation.
3. Der Schlüsselwort-Subparameter `TYPE` des Profilparameters `BPI` oder des Makros `NTBPI` muss auf `MSG` gesetzt sein.

# Betrieb des Message Buffer Pool

Der Message Buffer Pool wird unter Verwendung des Programms `NATMBPvr` betrieben, das in einer APF-Bibliothek (Authorized Program Facility) ausgeführt werden muss.

Die folgenden Themen werden behandelt:

- Message Buffer Pool einrichten
- Message Buffer Pool-Betriebsprogramm starten
- Message Buffer Pool-Betriebsprogramm stoppen

**Anmerkung:** In diesem Dokument steht die Schreibweise `vrs` oder `vr` für die jeweilige Produktversion (siehe auch *Version* im *Glossar*).

## Message Buffer Pool einrichten

Die von `NATMBPvr` zur Verfügung gestellten **Funktionen** (siehe auch ***Funktionsparameter***) werden dadurch aktiviert, dass sie

- durch eine Parameterkarte (`PARM=`) geliefert werden,
- aus einer Datei gelesen werden (siehe unten),
- oder durch das Operator-Kommando `MODIFY` geliefert werden, wenn `NATMBPvr` noch nicht beendet wurde.

`NATMBPvr` erwartet das erste Kommando im Parameterfeld (`PARM=`) des `EXEC`-Statements.

Sie können Folgendes eingeben:

- eine der im Abschnitt ***Message Buffer Pool-Betriebsfunktionen*** beschriebenen Funktionen,
- oder einen Verweis auf eine Eingabedatei mit `CF=<dd-name>`, wobei `<dd-name>` für einen in der JCL definierten DD-Namen steht.

  Es werden nur Card Image-Dateien unterstützt, d.h. `RECFM=F,LRECL=80`, und nur die ersten 72 Bytes des Eingabedatensatzes werden berücksichtigt.

  Jeder in der Eingabedatei enthaltene Satz stellt ein Kommando dar.

  Leere Sätze oder Sätze mit einem vorangestellten Stern (`*`) werden ignoriert.

  Eine Datei wird bis zum End-Of-File (EOF) verarbeitet.

  Beispiel: `PARM='CF=SYSIN1'`

Wenn das Parameterfeld nicht angegeben oder leer ist, werden die Kommandos standardmäßig aus der Datei `SYSIN` gelesen.

Es ist nur möglich, jeweils eine Funktion an der Konsole oder eine Funktion pro Zeile über die Kommandodatei einzugeben, andernfalls wird eine Fehlermeldung zurückgegeben.

Jedes Kommando, das von der Parameterkarte, per Dateieingabe oder per Eingabe über die Bedienerkonsole empfangen wird, wird auf der Bedienerkonsole angezeigt.

### Message Buffer Pool-Betriebsprogramm starten

Um das Programm `NATMBPvr` zu starten, können Sie entweder eine Started Task starten oder einen Job übergeben, der `NATMBPvr` ausführt.

### Message Buffer Pool-Betriebsprogramm stoppen

Das Programm `NATMBPvr` wird mit der Funktion `TERMINATE` (siehe ***Message Buffer Pool-Betriebsfunktionen***) oder im Notfall mit dem Betriebsprogramm `CANCEL` beendet.

## Beispiele für NATMBPvr-Ausführungsjobs

Die folgenden Beispiele zeigen Beispiele für Batch-Jobs zur Erstellung und Beendigung eines globalen Pufferpools.

- Beispiel 1
- Beispiel 2
- Beispiel 3

**Anmerkung:** In den folgenden Beispielen steht die Schreibweise *vrs* oder *vr* für die jeweilige Produktversion (siehe auch *Version* im *Glossar*).

### Beispiel 1

```
//MBPSTART JOB
//*
//* Starts a message buffer pool with the name NATvrMBP and
//* a size of 10 MB.
//* The subsystem used is NATv.
//*
//STEP EXEC PGM=NATMBPvr,PARM='BP=NATvrMBP,SI=10'
//SETPLIB DD DISP=SHR,DSN=USER.APF.LINKLIB
```

### Beispiel 2

```
//MBPRES JOB
//*
//* Starts a message buffer pool with the name MBP and a default size of
//* 100 MB. The subsystem used is SAGS.
//*
//STEP EXEC PGM=NATMBPvr,PARM='BP=MBP,S=SAGS'
```

### Beispiel 3

```
//MBPSTRT2
//* Read commands from SYSIN1:
//*
//* Start 2 message buffer pools (subsystem ID Nvrs) with name
//*   NATMBP1 - size=1000MB
//*   NATMBP2 - size=2000MB
//* If the buffer pools should shut down, send operator command MODIFY with
//* parameter "CF=SYSIN2" to execute the corresponding FSHUTs.
//*
//STEP EXEC PGM=NATMBPvr,PARM='CF=SYSIN1'
//SYSIN1  DD *
CREATE,BP=NATMBP1,S=Nvrs,SI=1000M
CREATE,BP=NATMBP2,S=Nvrs,SI=2000M
SHOWBP S=Nvrs
//SYSIN2 DD *
FSHUT,BP=NATMBP1,S=Nvrs
FSHUT,BP=NATMBP2,S=Nvrs
//*
```

## Message Buffer Pool-Betriebsfunktionen

Die folgenden Funktionen sind verfügbar:

- HELP - Zeigt eine Übersicht über die verfügbare Syntax
- CREATE - Message Buffer Pool anlegen
- FSHUT - Message Buffer Pool außer Betrieb setzen
- TERMINATE oder STOP- Message Buffer Pool-Betriebsprogramm beenden
- ZAPS - Anzeige der auf den Message Buffer Pool angewendeten Zaps

**Anmerkung:** Die Funktionsnamen können abgekürzt werden. Es genügt, wenn Sie nur das erste Zeichen verwenden, z.B. `T` für `TERMINATE`.

### HELP - Zeigt eine Übersicht über die verfügbare Syntax

Diese Funktion gibt eine Liste der verfügbaren Syntax-Kommandos und ggf. die Standardwerte der Funktionsparameter aus.

### CREATE - Message Buffer Pool anlegen

Mit dieser Funktion wird ein Message Buffer Pool mit den angegebenen **Parametern** angelegt.

### FSHUT - Message Buffer Pool außer Betrieb setzen

Der Message Buffer Pool mit den angegebenen **Parametern** wird freigegeben.

### TERMINATE oder STOP- Message Buffer Pool-Betriebsprogramm beenden

Das Betriebsprogramm des Message Buffer Pool wird beendet. Zuvor werden alle aktiven Message Buffer Pools freigegeben.

### ZAPS - Anzeige der auf den Message Buffer Pool angewendeten Zaps

Zeigt alle auf das Message Buffer Pool-Betriebsprogramm angewendeten Zaps an.

## Funktionsparameter

Die Funktionen des Betriebsprogramms des Message Buffer Pool können mit Hilfe von Parametern gesteuert werden. Diese Parameter können in beliebiger Reihenfolge angegeben und dürfen abgekürzt werden.

Die folgenden Parameter sind verfügbar:

`BPNAME` **Name des Message Buffer Pool**.
`BPLIST` **Name der Preload-Liste (optional)**.
`SUBSID` **Natural-Subsystemkennung (ID)**.
`SIZE` **Größe des Message Buffer Pool**.

**Anmerkung:** Der unterstrichene Teil des Parameternamens markiert die kürzest mögliche Abkürzung.

### BPNAME - Name des Message Buffer Pool

`BPNAME=value` gibt den Namen des anzulegenden Message Buffer Pool an.

| Wert | Erläuterung |
|---|---|
| 8 Bytes | Der Name des Message Buffer Pool.<br><br>**Anmerkung:** Wenn der angegebene Name kürzer als 8 Byte ist, werden Leerzeichen angehängt. |
| `MTBP` | Dies ist der Standardwert. |

### BPLIST - Name der Preload-Liste (optional)

`BPLIST=value` gibt den Namen der optionalen Preload-Liste an.

| Wert | Erläuterung |
|---|---|
| 8 Bytes | Der Name der Preload-Liste.<br><br>**Anmerkung:** Wenn der angegebene Name kürzer als 8 Byte ist, werden Leerzeichen angehängt.<br><br>Es gibt keinen Standardwert. |

### SUBSID - Natural-Subsystemkennung

`SUBSID=value` gibt die Kennung (ID) des Natural-Subsystems an.

| Wert | Erläuterung |
|---|---|
| 4 Bytes | Die Kennung des Natural-Subsystems.<br><br>**Anmerkung:** Wenn der angegebene Name kürzer als 8 Byte ist, werden Leerzeichen angehängt. |
| `NATv` | Dies ist der Standardwert, wobei *v* die erste Ziffer der aktuellen Natural-Version ist. |

### SIZE - Größe des Message Buffer Pool

`SIZE=value` gibt die Größe des Message Buffer Pool an.

| Wert | Erläuterung |
|---|---|
| 1 - 2000 MB | Die Größe des Message Buffer Pool. |
| 1 | Dies ist der Standardwert. |

**Anmerkung:** Ein Message Buffer Pool der Größe 1 MB kann bis zu 7000 Fehlermeldungen aufnehmen.

## Meldungen

Siehe *Message Buffer Pool Messages* in der *Messages and Codes*-Dokumentation.

# VII System-Spool-Zugang verwalten

# 21 System-Spool-Zugang verwalten


- Zweck ..... 198
- Voraussetzung ..... 198
- Verwendung der Write-to-Spool-Funktion ..... 198

Dieses Kapitel beschreibt die Funktion **Write-to-Spool** für Natural.

Siehe auch *Installing and Activating the Write-to-Spool Feature* im Abschnitt *Installing Entire System Server Interface on z/OS* in der *Installation for z/OS* -Dokumentation.

## Zweck

Mit der Funktion **Write-to-Spool** können Natural-Benutzer Reports direkt in den System-Spool schreiben. Die Funktion kann in jeder Natural-Umgebung (Com-plete, TSO, CICS, IMS TM, Batch usw.) verwendet werden. Sie nutzt die Entire System Server View (Datensicht) `WRITE-SPOOL`.

Unter z/OS ist `SYSOUT` Teil des Entire System Server-Job-Stream im JES-Spool und kann von jeder Software verarbeitet werden, die Ausgaben im JES-Spool erwartet, z. B. Entire Output Management. Der JES-Spool kann ein JES2- oder ein JES3-Spool sein.

## Voraussetzung

Um die Funktion **Write-to-Spool** nutzen zu können, muss der Entire System Server installiert sein.

## Verwendung der Write-to-Spool-Funktion

Die **Write-to-Spool**-Funktion wird über eine „Zugriffsmethode“ (Access Method) abgewickelt, die `ESS` für Entire System Server heißt. Sie können Ihren Drucker im Natural-Parametermodul oder dynamisch in Ihren Session-Parametern definieren.

### Wie Sie Ihren Drucker definieren

**Vorgehensweise:**

1 Definieren Sie den Drucker im **Natural-Parametermodul**.

Verwenden Sie das Makro `NTPRINT`, um die Druckernummer (*n*) und die Zugriffsmethode (`AM` anzugeben:

```
NTPRINT (n),AM=ESS
```

Beispiel:

```
NTPRINT (1,3),AM=ESS
```

In diesem Beispiel sind die Drucker 1 und 3 für die Verwendung mit der Zugriffsmethode `ESS` (Entire System Server) definiert.

**Oder:**

Definieren Sie den Drucker während des Sitzungsstarts, indem Sie den Profilparameter `PRINT` angeben, z. B:

```
PRINT=((1-6),AM=ESS)
```

In diesem Beispiel sind die Drucker 1 bis 6 für die Verwendung mit der Zugriffsmethode `ESS` (Entire System Server) definiert.

2 Verlinken Sie die Module der Zugriffsmethode mit dem Natural-Nukleus.

Siehe *Write-to-Spool Feature installieren und aktivieren* in *Entire System Server Interface installieren* in der *Installation für z/OS*-Dokumentation.

**Oder:**

Laden Sie die Module der Zugriffsmethode dynamisch, indem Sie die Profilparameter `RCA` und `RCALIAS` angeben:

```
RCA=(NATAM11),RCALIAS=(NATAM11,NATPWSAM)
```

wobei `NATPWSAM` das ausgelieferte Write-to-Spool-Modul ist, das die Standardparameter enthält.

Wenn Sie ein Modul mit angepassten Parametern verlinkt haben, verwenden Sie stattdessen den Namen dieses Moduls.

3 Definieren Sie das JES-Ziel (Destination) mit der Option `OUTPUT` des Statements `DEFINE PRINTER`. Sie können eines der folgenden Beispiele verwenden, je nachdem, ob Sie die Ausgabe an eine Spool-Datei, einen lokalen JES-Drucker oder über einen entfernten JES-Knoten an einen entfernten Benutzer oder ein entferntes Gerät senden wollen.

Beispiel:

```
DEFINE PRINTER (n) OUTPUT 'LOCAL' /* For printing on local JES/POWER printers
```

**Oder:**

```
DEFINE PRINTER (n) OUTPUT 'DAEF' /* For printing to JES spool called DAEF
```

**Oder:**

```
DEFINE PRINTER (n) OUTPUT 'DEST=node-name,REMOTE-USERID=user-id'  /* For printing ↩
to remote JES nodes
```

wobei:

- *n* die Nummer des NTPRINT-Eintrags im **Natural-Parametermodul** ist, wie in Schritt 1 beschrieben.
- *node-name* der Name des entfernten JES-Knotens ist.
- *user-id* die Kennung des Benutzers oder Geräts ist, das die Ausgabe erhält.

Reports können nun mit einem der folgenden Statements in den System-Spool geschrieben werden:

```
DISPLAY (n)
```

oder

```
PRINT (n)
```

wobei *n* die Nummer des NTPRINT-Eintrags im **Natural-Parametermodul** in **Schritt 1** ist.

Benutzer können das Ausgabeformat und die Anzahl der Exemplare mit den Klauseln FORMS und COPIES des Statements DEFINE PRINTER festlegen.

Beispiel:

```
DEFINE PRINTER (2) OUTPUT 'DEST'
                   FORMS 'FORM'
```

Die Voreinstellungen für Elemente wie Entire System Server-Knoten, Formulare und Ausgabeklasse finden Sie im Modul NATWSPDF.

## Beispiele

**Beispiel 1**

Angenommen, Sie verwenden die Auslieferungseinstellungen und führen das Natural-Programm aus:

```
DEFINE PRINTER (2) OUTPUT 'WK1'
WRITE (2) 'THIS IS A SMART RECORD'
CLOSE PRINTER (2)
```

Während der Ausführung dieses Programms können Sie die folgenden Felder mit ihren Werten in der Anzeige **Display Active Tasks** sehen:

```
DDNAME     DSID Owner C Dest Rec-Cnt Forms Wtr PageDef FormDef
SYS00001   104  WKK   A WK1     2     STD
```

Wenn Sie diesen Datensatz durchsuchen, können Sie sehen:

```
Page 1
THIS IS A SMART RECORD
```

**Beispiel 2**

Angenommen, Sie verwenden das Standard-Member:

| Parameter | Erläuterung (mögliche Werte) |
|---|---|
| `WSPDFLT NODE=55526,` | Zielknotennummer des Entire System Server (maximal 5 Zeichen) |
| `PROGRAM=HUGO,` | JES-Writer (maximal 8 Zeichen) |
| `CLASS=Y,` | `SYSOUT`-Klasse (1 Zeichen) |
| `HOLD=YES,` | Hold (`YES` oder `NO`) |
| `CNTL=A,` | Wagenrücklaufsteuerung (`A` oder `M`) |
| `FORM=WOFO,` | Formular (maximal 4 Zeichen)Formular (maximal 4 Zeichen) |
| `RMT=JESWOLF,` | JES remote (maximal 8 Zeichen) |
| `FORMDEF=FOWOLF,` | Formulardefinition (maximal 6 Zeichen) |
| `PAGEDEF=PAWOLF` | Seitendefinition (maximal 6 Zeichen) |

Führen Sie das folgende Natural-Programm aus:

```
DEFINE PRINTER (2) OUTPUT 'WK1'
WRITE (2) 'THIS IS A SMART RECORD'
CLOSE PRINTER (2)
```

Während der Ausführung dieses Programms können Sie die folgenden Felder mit ihren Werten in der Anzeige **Display Active Tasks** sehen:

```
DDNAME   DSID Owner C Dest Rec-Cnt Forms  Wtr   PageDef  FormDef
SYS00002 105   WKK  Y  WK1    2     WOFO  HUGO  PAWOLF   FOWOL
```

Wenn Sie diesen Dataset durchsuchen, können Sie sehen:

```
Page 1
THIS IS A SMART RECORD
```

**Beispiel 3**

Angenommen, Sie verwenden das Standard-Member:

| Parameter | Erläuterung (mögliche Werte) |
|---|---|
| `WSPDFLT NODE=55526,` | Entire System Server (NPR) Zielknoten (Knotennummer) |
| `PROGRAM=*OUTPUT,` | JES-Writer (maximal 8 Zeichen) |

Die anderen Parameter im Standard-Member werden nicht geändert.

Führen Sie das folgende Beispielprogramm aus:

```
DEFINE PRINTER (2) OUTPUT 'KURT'
PRINT (2) ' here comes KURT'
CLOSE PRINTER (2)
```

Danach holt sich der Entire System Server den Wert aus dem Feld `OUTPUT` im Statement `DEFINE PRINTER` und übergibt ihn als JES-Writer-Attribut für den spezifischen Spool-Dataset.

Wenn Sie in TSO/SDSF unter dem Jobnamen von Entire System Server nachsehen, können Sie Folgendes sehen:

```
PREFIX=NPR*  DEST=(ALL)  OWNER=*  SYSNAME=
NP   DDNAME   Time     Forms    FCB  UCS  Wtr      Flash
     SYS00005 10:20:48          **** **** KURT     ****
```

Wenn in JES ein zugehöriges JES-Writer-Programm definiert ist, erhält es die Kontrolle und behandelt diese Ausgabe wie in dem Programm definiert.

# VIII Natural 3GL CALLNAT-Schnittstelle verwenden

Dieser Teil enthält Informationen über die Natural 3GL CALLNAT-Schnittstelle, die es 3GL-Programmen ermöglicht, Natural-Subprogramme aufzurufen und auszuführen.

**Natural 3GL CALLNAT-Schnittstelle - Zweck, Voraussetzungen, Einschränkungen**

**Natural 3GL CALLNAT-Schnittstelle - Verwendung, Beispiele**

# 22 Natural 3GL CALLNAT-Schnittstelle - Zweck, Voraussetzungen, Einschränkungen


- Zweck der 3GL CALLNAT-Schnittstelle ..... 206
- Voraussetzungen ..... 206
- Einschränkungen ..... 208

Dieses Kapitel beschreibt den Zweck der 3GL CALLNAT-Schnittstelle sowie deren Voraussetzungen und Einschränkungen.

## Zweck der 3GL CALLNAT-Schnittstelle

Über die 3GL CALLNAT-Schnittstelle von Natural können 3GL-Programme Natural-Unterprogramme aufrufen und ausführen.

Die 3GL kann eine beliebige Programmiersprache sein, die die Standard-Linkage-Call-Schnittstelle unterstützt. In den meisten Fällen wird es sich um ein COBOL-Programm handeln, aber die Funktionalität kann auch von PL/1-, FORTRAN-, C- oder Assembler-Programmen genutzt werden.

### Verfügbarkeit

Die Schnittstelle ist im Batch-Modus unter z/OS und für die folgenden TP-Monitor-Umgebungen verfügbar:

- CICS
- Com-plete
- IMS TM
- TSO

## Voraussetzungen

In diesem Abschnitt werden die Voraussetzungen für die Ausführung eines Natural-Subprogramms aus einem 3GL-Programm heraus unter Verwendung eines internen `CALLNAT`-Statements beschrieben. Um die gewünschte Funktionalität zu verwirklichen, muss eine Natural-Umgebung eingerichtet werden, bevor Sie die `CALLNAT`-Schnittstelle aus Ihrem 3GL-Programm heraus ausführen.


- Speicherplatzbedarf
- Verlinkung

- Abhängigkeiten von der Umgebung

### Speicherplatzbedarf

Der Ansatz der Parameteradressierung in einem Natural-Programm setzt voraus, dass sich die übergebenen Parameter in einem von Natural zugewiesenen Bereich befinden, d.h. in einer seiner Größen. Das 3GL-Programm hingegen weist den Speicher für seine Variablen irgendwo im Adressraum des Tasks zu. Um die Adressierung dennoch erfolgreich zu gestalten, wird ein Call-by-Value-Verfahren für diejenigen Variablen verwendet, die sich nicht bereits in einem Natural-Bereich befinden. Das bedeutet, dass vor dem Aufruf des Natural-Subprogramms die zu übergebenden Parameter in einen Natural-Bereich, nämlich den `DATSIZE` Buffer, übertragen werden.

Neben dem Speicher für den Inhalt der Variablen wird je nach Anzahl der Parameter zusätzlicher Speicherplatz benötigt. Der gesamte Platzbedarf entspricht in etwa dem Platz, der im `DATSIZE` Buffer benötigt würde, wenn das Programm, das das Unterprogramm aufruft, in Natural kodiert wäre.

### Verlinkung

Um das Natural-Subprogramm aufzurufen, muss das 3GL-Programm die `CALLNAT`-Schnittstelle aufrufen. Je nach Leistungsfähigkeit und Funktionalität der Aufrufschnittstelle der 3GL-Programmiersprache kann die `CALLNAT`-Schnittstelle entweder in einer zugänglichen Load Library zum dynamischen Laden untergebracht oder mit dem 3GL-Programm verlinkt werden.

Es wird empfohlen, die `CALLNAT`-Schnittstelle, wann immer möglich, dynamisch aus einer Natural-Steplib zu laden, da diese Methode sicherstellt, dass immer die neueste Version des Programms verwendet wird.

Die Beispiele `XNATGC2` und `XNATGCP2` sollen die Technik des dynamischen Ladens und Aufrufens der `CALLNAT`-Schnittstelle aus COBOL bzw. PL/I verdeutlichen.

**Anmerkung:** Erkundigen Sie sich bei dem zuständigen Systemprogrammierer nach der besten Lösung in Ihrer Umgebung.

### Abhängigkeiten von der Umgebung

Das fremde 3GL-Modul kann entweder als `CSTATIC`-Modul mit Natural verlinkt und dann über eine Verzweigungs- und Verlinkungsanweisung aufgerufen werden, oder dynamisch geladen und über eine TP-abhängige Link-Methode aufgerufen werden.

Im letzteren Fall wird das 3GL-Modul TP-spezifisch geschrieben und die `CALLNAT`-Schnittstelle muss entsprechend angepasst werden. Zu diesem Zweck werden mehrere TP-spezifische Schnittstellenmodule bereitgestellt:

| Schnittstellenmodul | Zweck |
|---|---|
| NATXCAL | Zu verwenden in den folgenden Fällen:<br>■ wenn das 3GL-Modul entweder dynamisch geladen oder mit Natural verlinkt ist und dann durch eine Verzweigungs- und Verlinkungsanweisung aufgerufen wird (Batch, Com-plete, IMS TM, TSO, %P=S und %P=LS in CICS).<br>■ wenn das 3GL-Modul über die Option INTERFACE4 des CALL-Statement aufgerufen wird. Es stellt die Funktionen INTERFACE4 Natural Callnat Interface sowie die INTERFACE4 Callback zur Verfügung. Weitere Informationen zur INTERFACE4-Funktionalität finden Sie in der Dokumentation des CALL-Statement.<br>**Anmerkung:** Für CALL INTERFACE4-Zwecke kann NATXCAL nicht dynamisch geladen werden, sondern muss mit dem 3GL-Programm verlinkt werden. |
| NCIXCALL | Ist in einer CICS-Umgebung zu verwenden, wenn das 3GL-Modul mit EXEC CICS LINK aufgerufen wurde. NCIXCALL wird im Quellcode geliefert, der mit Ihren CICS-Makros zu kompilieren ist. Siehe auch *Natural CICS-Schnittstelle auf z/OS installieren* in der *Installation für z/OS*-Dokumentation. |
| NCIXCPRM | Ist in einer CICS-Umgebung zu verwenden, um die Parameteradressenliste zu erstellen, die als COMMAREA für das nachfolgende EXEC CICS LINK-Kommando verwendet wird. |

# Einschränkungen

### Beenden eines Natural-Subprogramms

Das aufgerufene Natural-Subprogramm sollte mit einer Rückkehr zum aufrufenden Programm beendet werden.

### Unzulässige Natural-Statements

Die folgenden Statements dürfen nicht verwendet werden.

- FETCH
- RUN
- STOP
- TERMINATE

Wenn sie in dem aufgerufenen Natural-Subprogramm verwendet werden, führen sie zu einem entsprechenden Natural-Laufzeitfehler (NAT0967).

### Übergabe von Parameterwerten durch das 3GL-Programm

Die vom 3GL-Programm übergebenen Parameterwerte dürfen sich nicht in einem schreibgeschützten Speicherbereich befinden.

### Dynamische Arrays

Arrays mit dynamischen Bereichen sind nicht möglich.

### TP-Monitor-spezifische Einschränkungen

- **Unter CICS**
  In CICS-Umgebungen muss das 3GL-Programm, das die Natural 3GL CALLNAT-Schnittstelle benutzt, für den Conversational Mode geschrieben sein. Das 3GL-Programm läuft auf der zweiten CICS-Programmebene. Daher kann die pseudo-konversationelle Programmtechnik nicht verwendet werden.
- **Unter IMS TM**
  In IMS TM-Umgebungen, in denen Natural läuft, kann die 3GL CALLNAT-Schnittstelle nur verwendet werden, wenn sowohl das 3GL-Programm als auch das Natural-Subprogramm keine Terminal-E/A ausgeben. Wenn die Statements `DISPLAY`, `INPUT` und `WRITE` in dem aufgerufenen Natural-Subprogramm verwendet werden, führen sie zu einem entsprechenden Natural-Laufzeitfehler (NAT0967).

# 23 Natural 3GL CALLNAT-Schnittstelle - Verwendung, Beispiele


- Verwendung ..... 212
- Beispielumgebungen ..... 216

Dieses Kapitel beschreibt die Verwendung der 3GL CALLNAT-Schnittstelle und enthält eine Reihe von Beispielen für 3GL CALLNAT-Umgebungen.

# Verwendung

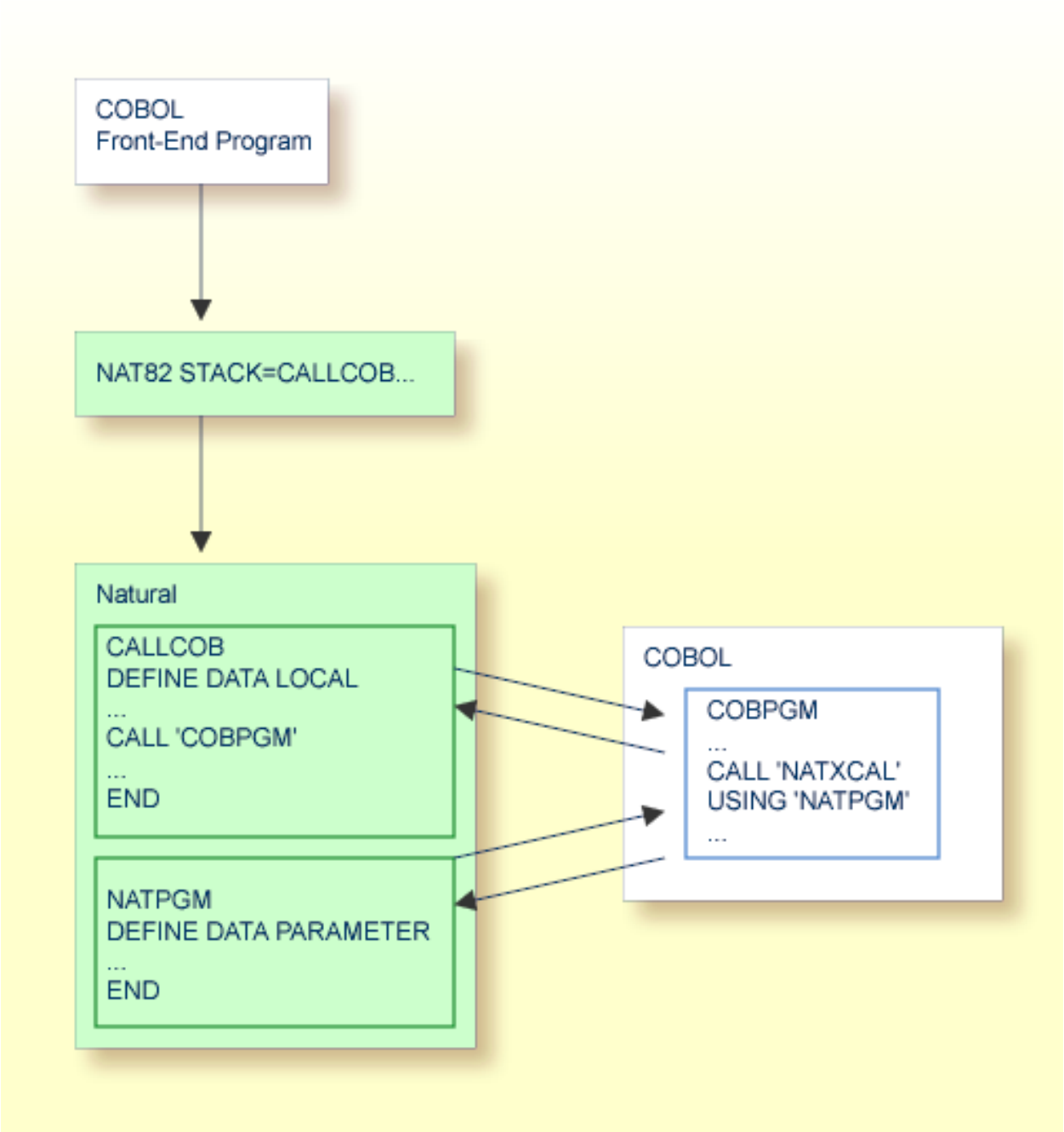
Die folgenden Themen werden behandelt: …

- Übersicht
- Call-Struktur
- Behandlung von Parametern

## Übersicht

Wenn Sie ein Natural-Subprogramm aus einem 3GL-Programm aufrufen, muss eine Natural-Sitzung aktiv sein, d.h. das 3GL-Programm selbst muss von Natural aufgerufen werden.

Daher müssen Sie besondere Vorkehrungen treffen, wenn Sie nicht wollen, dass die Natural-Schicht in Erscheinung tritt. Die folgende Abbildung soll Ihnen einen Überblick darüber geben, wie eine Anwendung, die die Natural-3GL-CALLNAT-Schnittstelle verwendet, in einem solchen Fall gestaltet werden kann:




Die erforderliche Umgebung wird zunächst durch den Aufruf eines Natural-Startup-Programms geschaffen. Mit Hilfe des Natural-Statements `CALL` kann dieses Startup-Programm dann ein 3GL-Programm aufrufen, von dem aus Sie die `CALLNAT`-Schnittstelle aufrufen können.

### Call-Struktur

Das Natural-Hauptprogramm ist sehr einfach. Es ruft z.B. nur ein COBOL-Programm auf:

```
.....
CALL 'COBPGM'
END
.....
```

Das CALL-Statement der 3GL-Programmiersprache (z.B. COBOL) muss Zugriff auf die Natural-3GL-CALLNAT-Schnittstelle haben, die dann das Natural-Subprogramm aufruft:

```
.....
CALL 'interface' USING natpgm p1 ... pn
.....
```

Die Parameter ist *interface* umgebungsabhängig (z.B. NATXCAL) und mit dem aufrufenden Programm verlinkt. Der Parameter *natpgm* muss eine alphanumerische Variable von 8 Bytes sein, die den Namen des aufzurufenden Natural-Subprogramms enthält. Die Parameter *p1 ... pn* werden an das Natural-Subprogramm übergeben.

**Beispiel (für alle Umgebungen außer CICS):**

Das COBOL-Programm COBPGM könnte ein Coding ähnlich dem folgenden enthalten:

```
.....
MOVE 'FINDNPGM' TO natpgm
CALL 'interface' USING natpgm number name
IF natpgm NE 'FINDNPGM'
THEN GOTO error_handling_1
.....
```

Das aufgerufene Natural-Subprogramm FINDNPGM errechnet die Anzahl der Personen in der Datei EMPLOYEES, deren Name dem vom COBOL-Programm übergebenen Wert entspricht:

```
DEFINE DATA
PARAMETER
1 pnumber (P10)
1 pname (A20)
LOCAL
1 emp VIEW OF employees
END-DEFINE
*
RESET presp
FIND NUMBER emp WITH name=pname
MOVE *NUMBER TO pnumber
ESCAPE ROUTINE
```

Tritt bei der Ausführung des Subprogramms ein Fehler auf, werden Informationen über diesen Fehler in der Variablen *natpgm* in der Form `*NAT`*nnnn* zurückgegeben, wobei *nnnn* die entsprechende Natural-Fehlernummer ist.

**Beispiel (nur für CICS):**

Unter CICS sollte der Aufruf eines Natural-Subprogramms aus z. B. COBOL wie folgt aussehen:

```
...
WORKING STORAGE SECTION
...
01 PARM-LIST PIC X(132).
01 NATPGM PIC X(8).
01 NUMBER PIC 9(10) comp-3.
01 NAME PIC X(20).
...
PROCEDURE DIVISION
...
MOVE 'FINDNPGM' TO NATPGM
CALL 'NCIXCPRM' USING PARM-LIST NATPGM NUMBER NAME ...
EXEC CICS LINK PROGRAM('NCIXCALL')
COMMAREA(PARM-LIST) LENGTH(132) END-EXEC.
...
```

Das aufgerufene Subprogramm `NCIXCPRM` baut die Parameteradressenliste auf, die als `COMMAREA` im nachfolgenden `EXEC CICS LINK`-Kommando verwendet wird.

### Behandlung von Parametern

Es erfolgt keine Überprüfung von Format und Länge. Es liegt in der Verantwortung des Aufrufers, eine korrekte Parameterliste zu übergeben. Die Anzahl, das Format und die Länge der Parameter werden durch das aufgerufene Natural-Subprogramm definiert.

Bei der Übergabe von Parametern sollten keine Gruppen-Arrays übergeben werden, da sie als einzelne Arrays aufgelöst werden:

**Beispiel für eine ungültige Syntax:**

```
.....
01 GROUP (1:2)
02 F1
02 F2
.....
.....
CALL ..... F1 F2
.....
```

**Beispiel für eine gültige Syntax:**

```
.....
01 F1 (1:2)
01 F2 (1:2)
....
.....
CALL ..... F1 F2
.....
```

Arrays mit dynamischen Bereichen können nicht als Parameter verwendet werden.

## Beispielumgebungen

Das Ziel der folgenden 3GL-CALLNAT-Beispielumgebungen ist es, zu demonstrieren, wie eine COBOL-Routine ein Natural-Subprogramm unter bestimmten TP-Monitor-Systemen oder im Batch-Modus aufrufen kann, und systemspezifische Anweisungen zur Erstellung solcher Umgebungen zu geben.

Die folgenden Themen werden behandelt:

- Beispielumgebung für CICS
- Weitere Beispiele
- Beispiel für beliebige andere unterstützte Umgebungen

### Beispielumgebung für CICS

Führen Sie die folgenden Schritte aus, um eine Natural 3GL CALLNAT-Beispielumgebung unter CICS zu erstellen:

**Schritt 1: Anlegen der Umgebungsinitialisierung**

- Nehmen Sie das Front-End-Programm in Betrieb, das die 3GL-CALLNAT-Umgebung initialisiert.
- Verwenden Sie das COBOL-Frontend `XNCIFRCX` in der Natural/CICS-Source-Library. Es startet Natural, legt `LOGON YOURLIB` auf den Stack und führt das Programm `TSTCOB` aus, das die Natural-3GL-CALLNAT-Umgebung initialisiert.
- Suchen Sie die Zeichenkette `NC`*`vr`* (wobei *`vr`* für die jeweilige Produktversion steht) im Quellcode und ersetzen Sie sie durch die gültige Transaktionskennung (ID) für Natural.
- Kompilieren und verlinken Sie das COBOL-Programm und definieren Sie das Programm über `CEDA DEFINE PROGRAM` für CICS.

**Schritt 2: Installieren Sie den COBOL-Beispielaufruf**

In der Natural/CICS-Source-Library `NCI.SRCE` befindet sich das Beispiel-Member `XNCI3GC1`, das einen Standardaufruf für das Natural-Subprogramm `MYPROG` enthält.

- Erstellen Sie zu Testzwecken das folgende Programm in der Library SYSTEM und speichern Sie es mit dem Systemkommando STOW als:

```
WRITE 'BEFORE PGM EXECUTION'
CALL 'COBNAT'
WRITE 'AFTER PGM EXECUTION'
END
```

- Sehen Sie sich den XNCI3GC1-Quellcode an und überprüfen Sie die CALL- und LINK-Anweisungen. Kompilieren und verlinken Sie als COBNAT mit den folgenden CICS-INCLUDE-Direktiven oder verwenden Sie Schritt 2 des Beispiel-Jobs NCTI070:

```
INCLUDE CICSLIB(DFHECI)
INCLUDE XNCI3GC1                    <= output from translator and compiler
INCLUDE NCILIB(NCIXCPRM)
ENTRY XNCI3GC1
NAME COBNAT(R)
```

**Schritt 3: Erstellen eines Beispiels für ein Natural-Subprogramm**

Standardmäßig ist das Source Member XNCI3GC1 so eingerichtet, dass es das Natural-Subprogramm MYPROG in der Library YOURLIB aufruft. Das Programm TSTCOB startet, wie oben erwähnt, den Prozess durch den Aufruf von COBNAT, das den eigentlichen Aufruf des Natural-Subprogramms MYPROG enthält.

Erstellen Sie das Subprogramm MYPROG, um das ausführende Natural-Subprogramm zu demonstrieren.

```
DEFINE DATA PARAMETER
  01 PARM1 (A18)
  01 PARM2 (A18)
  01 PARM3 (A18)
END-DEFINE
*
  MOVE 'PARAM01' TO PARM1
  MOVE 'PARAM02' TO PARM2
  MOVE 'PARAM03' TO PARM3
END
```

**Schritt 4: Überprüfen der CICS-Ressourcen**

- Verwenden Sie den Job NCII005 als Richtlinie für die Definition der CICS-Ressourcen (PPT und PCT).
- Definieren Sie die erforderlichen CICS-Ressourcen (PPT und PCT).

**Schritt 5: Testen der Umgebung**

Testen Sie die Umgebung, indem Sie die NCYC-Standardtransaktion verwenden. Verwenden Sie CEDF, um die Programmsteuerung zu überwachen und die dabei verwendeten Datenbereiche zu beobachten.

**Wichtig:** Da Natural an der Spitze der CICS-Programmhierarchie steht, muss jedes COBOL-Subprogramm, das Terminal-Ein-/Ausgaben ausgibt, im Conversational Mode laufen. Pseudokonversationsprogramme müssten geändert werden, und jede Neuentwicklung, die die Natural-3GL-CALLNAT-Schnittstelle verwendet, sollte im Konversationsmodus erfolgen.

### Weitere Beispiele

| Beispiel-Programm | Beispiele |
|---|---|
| XNCI3GC2 | COBOL-Beispiel mit der gleichen Funktionalität wie XNCI3GC1, aber mit der Übernahme von Parametern aus dem aufrufenden Natural-Programm. |
| XNCI3GP1 | PL/I-Beispiel mit der gleichen Funktionalität wie COBOL-Beispiel XNCI3GC1. |
| XNCI3GP2 | PL/I-Beispiel mit der gleichen Funktionalität wie XNCI3GC1, aber mit der Übernahme von Parametern aus dem aufrufenden Natural-Programm. |

### Weitere Nicht-CICS-Beispiele

| Beispiel-Programm | Beispiele |
|---|---|
| XNAT3GC2 | COBOL-Beispiel mit der gleichen Funktionalität wie das CICS-Beispiel XNCI3GC2. |
| XNAT3GP2 | PL/I-Beispiel mit der gleichen Funktionalität wie das CICS-Beispiel XNCI3GP2. |

### Beispiel für beliebige andere unterstützte Umgebungen

Führen Sie die folgenden Schritte aus, um ein Beispiel für Natural 3GL CALLNAT zu erstellen:

**Schritt 1: Assemblieren und Verknüpfen von ASMNAT**

Die Beispiel-Assembler-Routine XNAT3GA1 enthält ein grundsätzliches Beispiel für den Zugriff auf die CALLNAT-Schnittstelle. Die Konventionen für den Registeraufruf finden sich im Quelltext dieses Programms.

Verknüpfen Sie NATXCAL mit XNAT3GA1 mit dem Einstiegspunkt ASMNAT.

**Schritt 2: Starten der Natural-Sitzung**

Starten Sie eine Natural-Sitzung, indem Sie ein Programm auf den Stack legen, das das Programm ASMNAT aufruft, das wiederum das Natural-Subprogramm ASMNAT aufruft.

# IX Betrieb des Software AG Editor

Dieser Teil enthält Informationen zum Betrieb des Software AG Editor.

Der Software AG Editor gehört zur Grundfunktionalität von Natural und wird hauptsächlich von verschiedenen Natural-Teilprodukten und anderen Produkten genutzt.

**Editor-Arbeitsdatei**

**Editor Buffer Pool**

Siehe auch:

- *SYSEDT Utility* - Verwaltungsfunktionen für den Editor Buffer Pool
- *Software AG Editor installieren* in der *Installation für z/OS*-Dokumentation
- *Software AG Editor* in der *Editoren*-Dokumentation

# 24 Editor-Arbeitsdatei


- Struktur der Editor-Arbeitsdatei ..... 222
- Editor-Arbeitsdatei unter z/OS ..... 223
- Verwendung der Software AG Editor Work File Formatting Utility ..... 224
- Formatierung während der Initialisierung ..... 224
- Verwaltung der Editor-Arbeitsdatei ..... 225
- Editor-Arbeitsdatei unter Com-plete/SMARTS ..... 226

Dieses Kapitel beschreibt den Aufbau, die Verwendung und die Verwaltung der Editor-Arbeitsdatei (Work File).

Siehe auch:

- *SYSEDT Utility* - Verwaltungsfunktionen für den Editor Buffer Pool
- *Software AG Editor installieren* in der *Installation für z/OS*-Dokumentation
- *EDBP - Software AG Editor Buffer Pool-Definitionen* in der *Parameter-Referenz*-Dokumentation
- *Software AG Editor* in der *Editoren*-Dokumentation

## Struktur der Editor-Arbeitsdatei

Die Arbeitsdatei des Editors ist ein Dataset mit relativen Datensätzen (Records) fester Länge. Sie ist in drei Teile unterteilt:

- Steuerdatensatz
- Arbeitsdatensätze
- Wiederherstellungsdatensätze

**Anmerkung:** Wenn Sie einen mit dem Profilparameter `EDPSIZE` definierten Editor-Hilfs-Buffer Pool verwenden, ist keine Editor-Arbeitsdatei erforderlich.

### Steuerdatensatz

Der Steuerdatensatz enthält Informationen zur Buffer Pool-Steuerung einschließlich der Buffer Pool-Parameter.

Während der ersten Initialisierung der Arbeitsdatei oder während eines Buffer Pool-Kaltstarts (ausgelöst durch den Editor Buffer Pool-Subparameter `COLD`) werden die im Editor Buffer Pool-Parameter `EDBP` und/oder im entsprechenden Makro `NTEDBP` definierten Werte im Arbeitsdatei-Steuerdatensatz (Work File Control Record) gespeichert. Außerdem werden die aktuelle Betriebssystemkennung (Systemvariable `*HOSTNAME`) und der globale Buffer Pool-Name bzw. der aktuelle Jobname zur späteren Überprüfung gespeichert.

Sie können den Steuerdatensatz mit der Funktion *Generation Parameters - Generierungs-Parameter* des Dienstprogramms SYSEDT ändern.

Bei Puffer Pool-Warmstarts werden die Buffer Pool-Parameter aus dem Steuerdatensatz gelesen.

### Arbeitsdatensätze

Die Arbeitsdatensätze (Work Records) enthalten logische Dateisätze, die aus dem Buffer Pool verschoben wurden, weil nicht genügend freie Buffer Pool-Blöcke vorhanden sind.

Logische Arbeitsdatei-Datensätze (Logical Work File Records) gehen bei einem Neustart des Buffer Pool oder bei einer Zeitüberschreitung für die logische Datei verloren.

### Wiederherstellungsdatensätze

Die Wiederherstellungsdatensätze (Recovery Records) enthalten Checkpoint-Informationen von Editor-Sitzungen. Wenn das System abnormal beendet wird, können diese Informationen von der Editor-Wiederherstellungsfunktion verwendet werden, um logische Dateien wiederherzustellen. Wiederherstellungsdatensätze gehen bei einem Kaltstart des Buffer Pool verloren.

Die Wiederherstellungsfunktion wird nur von Natural ISPF verwendet. Wenn Sie nicht beabsichtigen, dieses Produkt zu verwenden, können Sie ohne die Wiederherstellungsfunktion arbeiten, indem Sie den Editor Buffer Pool-Subparameter `PWORK=100` definieren.

## Editor-Arbeitsdatei unter z/OS

Eine Editor-Arbeitsdatei (Editor Work File) entspricht einem **Editor Buffer Pool**. Wenn Sie einen globalen Editor Buffer Pool verwenden wollen, muss die Editor-Arbeitsdatei von allen Benutzern gemeinsam genutzt werden, die denselben globalen Editor Buffer Pool verwenden. Die zugegriffene Editor-Arbeitsdatei kann nur von Sitzungen innerhalb desselben Betriebssystems (Systemvariable `*HOSTNAME`) und mit demselben globalen Buffer Pool oder demselben Jobnamen für lokale Buffer Pools verwendet werden. Diese Verbindung kann nur durch einen Buffer Pool-Kaltstart beendet werden. Alternativ kann das Dienstprogramm zur Formatierung der Arbeitsdatei des Software AG Editor verwendet werden, um die Arbeitsdatei-Verbindung zurückzusetzen.

Die Editor-Arbeitsdatei muss groß genug sein, um die Editor-Sitzungen aller Benutzer zu beinhalten. Es wird eine Mindestanzahl von 100 Sätzen pro Editor-Benutzer empfohlen. Die Datensatzlänge der Arbeitsdatei muss fest sein, kann von 504 bis 16384 Byte definiert werden und muss ein Vielfaches von 8 sein.

**Anmerkung:** Die Satzlänge von Datasets oder PDS-Members, die mit Natural ISPF bearbeitet werden, kann nicht größer sein als die Satzlänge dieser Editor-Arbeitsdatei (Editor Work File).

Die Größe eines Arbeitsdatei-Datensatzes wird jeweils bei der Zuordnung der Editor-Arbeitsdatei festgelegt (Standardgröße ist 4088).

Die Gesamtzahl der Sätze der Editor-Arbeitsdatei hängt von dem zugeordneten Dataset-Speicherplatz für die Editor-Arbeitsdatei ab.

Es gibt zwei Möglichkeiten, die Editor-Arbeitsdatei zu formatieren:

- Offline mit der **Software AG Editor Work File Formatting Utility**.
- Online **während der Initialisierung des Buffer Pool**.

## Verwendung der Software AG Editor Work File Formatting Utility

Diese Methode ist zu bevorzugen, da kein Online-Benutzer warten muss, bis die Formatierung abgeschlossen ist. Optional kann das **Natural-Parametermodul** assembliert und mit der Software AG Editor Work File Formatting Utility verlinkt werden, um Editor-Buffer-Pool-Parameter mit Hilfe des Makros `NTEDBP` festzulegen. Ansonsten gelten die Standardparameterwerte.

Während der Neuformatierung darf die Arbeitsdatei jedoch nicht in Gebrauch sein, d.h. das System oder die Systeme, die den entsprechenden Buffer Pool verwenden, müssen vor der Neuformatierung beendet worden sein.

## Formatierung während der Initialisierung

Wenn sich der Editor Buffer Pool in einem nicht initialisierten oder beendeten Zustand befindet, wird während der ersten Sitzung, in der der Software AG Editor verwendet wird, ein „Buffer Pool-Kaltstart“ unter einer der folgenden Bedingungen durchgeführt:

1. wenn die Arbeitsdatei noch nicht formatiert worden ist,
2. wenn der **Steuerdatensatz** „Kaltstart“ angibt (dies kann auch mit dem Dienstprogramm `SYSEDT` für die Editor-Buffer-Pool-Verwaltung angegeben werden, siehe *Generation Parameters - Generierungs-Parameter*),
3. wenn der Buffer-Pool-Subparameter `COLD=ON` angegeben wurde.

Andernfalls wird ein Warmstart des Buffer Pool durchgeführt, wenn während der Buffer Pool-Initialisierung ein gültiger Steuerdatensatz gefunden wird. In diesem Fall werden alle Buffer Pool-Parameter aus dem Arbeitsdatei-Steuerdatensatz übernommen und es werden keine Sätze formatiert.

## Verwaltung der Editor-Arbeitsdatei

Wenn Sie die Größe der Editor-Arbeitsdatei ändern wollen (z.B. weil sie zu klein ist), können Sie die `COPY`-Funktion der Software AG Editor Work File Formatting Utility benutzen, um einen Buffer Pool-Kaltstart zu vermeiden, d.h. den Verlust der Wiederherstellungssätze.

Um eine vorhandene Editor-Arbeitsdatei zu kopieren, müssen Sie die folgenden Schritte durchführen:

1. Ändern Sie alle Buffer Pool-Parameter mit dem Dienstprogramm `SYSEDT`, z. B. `PWORK`, wenn Sie den Prozentsatz der Work Records in der Datei ändern möchten.
2. Beenden Sie den Editor Buffer Pool mit den *Administration Facilities - Verwaltungsfunktionen* des Dienstprogramms `SYSEDT` und stellen Sie sicher, dass nach der Beendigung des Buffer Pool keine Natural-Sitzung den Editor verwendet.
3. Schließen Sie (falls erforderlich) die Arbeitsdatei des Editors und geben Sie sie frei.
4. Benennen Sie die Editor-Arbeitsdatei mit dem VSAM-Dienstprogramm `IDCAMS` (Kommando `ALTER`) um.
5. Definieren Sie eine neue Editor-Arbeitsdatei mit dem ursprünglichen Namen und eventuell einer anderen Größe, aber mit derselben Satzlänge (Record Length).
6. Führen Sie die folgenden Schritte durch:
   - Fügen Sie in der `EXEC`-JCL-Karte `PARM=COPY` hinzu.
   - Damit die umbenannte Editor-Arbeitsdatei `CMCOPY` in die neue Arbeitsdatei `CMEDIT` kopiert wird, müssen Sie `//CMCOPY DD....` hinzufügen.
   - Führen Sie die Software AG Editor-Workfile-Formatting Utility mit der neuen Datei aus.
7. Überprüfen Sie das Job-Protokoll auf mögliche Fehler.
8. Ordnen Sie die Editor-Arbeitsdatei neu zu und öffnen Sie sie (falls erforderlich) erneut.
9. Verwenden Sie das Dienstprogramm `SYSEDT`, um zu überprüfen, ob der Buffer Pool und die Arbeitsdatei erfolgreich neu gestartet wurden.

**Wichtig:** Alle Natural-Sitzungen müssen neu gestartet werden, wenn Sie wollen, dass diese den Editor nach dem Neustart des Buffer Pool verwenden können.

## Editor-Arbeitsdatei unter Com-plete/SMARTS

SMARTS-Arbeitsdateien befinden sich im SMARTS Portable File System. Der Pfad muss mit der SMARTS-Umgebungsvariablen `$NAT_WORK_ROOT` angegeben werden. Der Name der Editor-Arbeitsdatei wird mit dem EDBP-Subparameter `DDNAME` des Profilparameters `EDBP` angegeben.

Die Formatierung einer Editor-Arbeitsdatei ist nur während der Buffer-Pool-Initialisierung (online) möglich. Es gibt derzeit unter SMARTS kein Tool, um eine Editor-Arbeitsdatei offline zu formatieren.

# 25 Editor Buffer Pool


- Zweck des Editor Buffer Pool ..... 228
- Freie Blöcke erhalten ..... 229
- Initialisierung des Editor Puffer Pool ..... 229
- Neustart des Editor Buffer Pool ..... 230
- Editor Buffer Pool-Parameter ..... 230
- Buffer Pool-Initialisierung für Mehrbenutzerumgebungen ..... 230

Dieses Dokument beschreibt Zweck, Verwendung und Funktionsweise des Editor Buffer Pool, eines Zwischenspeichers, der vom Software AG Editor verwendet wird.

## Zweck des Editor Buffer Pool

Der Editor Buffer Pool kann als eine Erweiterung des Editor-Arbeitsbereichs angesehen werden (siehe auch Profilparameter `SSIZE`). Er ist ein Zwischenspeicher, der vom Software AG Editor für die Verwaltung seiner logischen Dateien verwendet wird.

Eine logische Datei besteht aus einem oder mehreren logischen Datensätzen und enthält die Daten eines Natural-Quellcodeobjekts oder einer Datei (z. B. eines Jobs, eines PDS-Members oder eines LMS-Elements), die mit dem Editor verwaltete werden. Da ein Benutzer mit mehr als einem Objekt gleichzeitig arbeiten kann, können für jeden Benutzer mehrere logische Dateien gleichzeitig existieren.

Die Anzahl der logischen Dateien (sowie der Prozentsatz der Wiederherstellungssätze in der **Editor-Arbeitsdatei**) wird im Buffer Pool-Parametermakro definiert.

Der Editor Buffer Pool kann als lokaler oder globaler Buffer Pool oder als Hilfs-Buffer Pool (siehe auch Profilparameter `EDPSIZE`) definiert werden. In Mehrbenutzerumgebungen (CICS und IMS TM) wird der Editor Buffer Pool von allen Editorbenutzern entweder derselben Region (lokaler Pool) oder mehrerer Regionen (globaler Pool) gemeinsam genutzt.

Der Editor Puffer Pool enthält verschiedene Steuertabellen und eine Reihe von Datenblöcken:

| Bereich | Größe |
|---|---|
| Hauptsteuerblock | 500 Bytes |
| Tabelle der logischen Dateien | 20 Bytes pro logische Datei |
| Arbeitsdateitabelle | 4 Bytes pro Datensatz |
| Tabelle der Wiederherstellungsdatei | 16 Bytes pro Datensatz |
| Buffer Pool-Blocktabelle | 2828 Bytes pro Block |
| Buffer Pool-Blöcke | siehe Text unten |

Da die Größe eines Buffer Pool-Blocks der Größe eines Arbeitsdateisatzes entspricht, kann ein Buffer Pool-Block einen logischen Dateisatz enthalten.

Der Buffer Pool wird vom ersten Benutzer des Editors initialisiert. Während der Buffer Pool-Initialisierung beim Warmstart werden alle Wiederherstellungsdatensätze überprüft, um die Tabelle der Wiederherstellungsdateien anzulegen.

Für den Zugriff auf den Buffer Pool stehen mehrere Funktionen zur Verfügung (z. B. Funktionen zum Zuordnen, Lesen, Schreiben oder Löschen eines Datensatzes).

## Freie Blöcke erhalten

Wenn der Buffer Pool voll wird, müssen Buffer Pool-Blöcke in einen externen Dataset, die Arbeitsdatei des Editors, verschoben werden, um freie Blöcke zu erhalten.

In einer solchen Situation überprüft der Editor alle logischen Dateien auf ihren Timeout-Wert und löscht jede logische Datei, auf die innerhalb der angegebenen Zeit nicht zugegriffen wurde. Das bedeutet, dass alle Buffer Pool-Blöcke und Arbeitsdateisätze freigegeben werden und die logische Datei verloren ist.

Wenn immer noch kein Buffer Pool-Block verfügbar ist, verschiebt der Editor den ältesten Block in die Arbeitsdatei, und zwar entsprechend den angegebenen Timeout-Parameterwerten (siehe die Funktion *Generation Parameters - Generierungs-Parameter* des Dienstprogramms `SYSEDT`.

## Initialisierung des Editor Puffer Pool

Beim ersten Aufruf des Software AG Editors wird ein nicht initialisierter Editor Buffer Pool initialisiert. Anschließend werden die verschiedenen Kontrollblöcke angelegt. Es gibt zwei verschiedene Modi für die Initialisierung von Buffer Pool und Arbeitsdatei (Workfile): **Kaltstart** und **Warmstart**.

### Buffer Pool-Kaltstart

Ein Buffer Pool-Kaltstart kann durch den Editor Buffer Pool-Subparameter `COLD` oder durch das Dienstprogramm `SYSEDT` oder automatisch (wenn die Editor-Arbeitsdatei unformatiert ist) ausgelöst werden.

Während eines Buffer Pool-Kaltstarts werden die Werte des Editor Puffer Pool-Parameters `EDBP` oder des entsprechenden Makros `NTEDBP` im Arbeitsdatei-Steuerungsdatensatz gespeichert und alle Wiederherstellungssätze der Arbeitsdatei werden gelöscht.

### Buffer Pool-Warmstart

Während eines Buffer Pool-Warmstarts werden die Buffer Pool-Parameter aus dem Arbeitsdatei-Steuerungsdatensatz gelesen, und alle Arbeitsdatei-Wiederherstellungsdatensätze werden gelesen, um die Wiederherstellungsdatentabelle im Buffer Pool anzulegen.

Die Arbeitsdatei, auf die der Editor zugreift, kann nur von Sitzungen innerhalb desselben Betriebssystems (Systemvariable `*HOSTNAME`) und mit demselben globalen Buffer Pool oder demselben Jobnamen für lokale Buffer Pools verwendet werden. Diese Verbindung kann nur durch einen Buffer Pool-Kaltstart beendet werden. Alternativ kann die Software AG Editor Workfile Formatting Utility verwendet werden, um die Arbeitsdatei-Verbindung zurückzusetzen.

## Neustart des Editor Buffer Pool

Mit dem Dienstprogramm `SYSEDT` zur Verwaltung des Editor Buffer Pool können Sie den Editor Buffer Pool beenden, d.h. ihn in den nicht initialisierten Zustand versetzen. Dadurch wird ein Neustart des TP-Systems oder des globalen Buffer Pool vermieden.

Falls `SYSEDT` aufgrund von Buffer Pool-Problemen nicht verfügbar ist, kann das Programm `BPTERM` verwendet werden, um den Buffer Pool zu beenden.

**Wichtig:** Alle Natural-Sitzungen müssen wiederhergestellt werden, wenn sie den Editor nach dem Neustart des Buffer Pool verwenden sollen.

## Editor Buffer Pool-Parameter

Der Editor Buffer Pool-Parameter `EDBP` oder das entsprechende Makro `NTEDBP` im Natural-Parametermodul wird benötigt, um Parameter für den Betrieb des Editor Buffer Pool zu definieren.

Bei der Formatierung der Editor-Arbeitsdatei werden diese Parameter im Arbeitsdatei-Steuerungssatz gespeichert, während alle anderen Sätze gelöscht werden. Die Neuformatierung einer zuvor verwendeten Arbeitsdatei bedeutet also, daß alle Editor-Checkpoint- und Wiederherstellungsinformationen verlorengehen.

Einige dieser Parameter können während der Ausführung des Buffer Pool mit Hilfe des Dienstprogramms `SYSEDT` dynamisch geändert werden.

## Buffer Pool-Initialisierung für Mehrbenutzerumgebungen

Während der Buffer Pool-Initialisierung werden alle Wiederherstellungssätze aus der Arbeitsdatei des Editors gelesen. Daher müssen die ersten Benutzer lange warten oder erhalten sogar eine Zeitüberschreitungsmeldung (Timeout), bis die Editor Buffer Pool-Initialisierung abgeschlossen ist.

Aus diesem Grund wird ein spezielles Natural-Programm ausgeliefert, das die Buffer Pool-Initialisierung auslöst, bevor der erste Benutzer aktiv wird. Dieses Programm kann entweder während des Starts des TP-Monitors aktiviert werden oder durch einen Batch-Job, wenn ein globaler Buffer Pool verwendet wird.

Die Sitzung muss dann mit dem Session-Parameter gestartet werden:

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

**Unter CICS:** Falls die Sitzung asynchron läuft, muss `SENDER=CONSOLE` angegeben werden, um eventuelle Fehlermeldungen während der Initialisierung angezeigt zu erhalten. Das Quellcodeprogramm `FRONTPLT` wird als Beispielprogramm mitgeliefert, um Ihnen zu zeigen, wie Sie eine asynchrone Natural-Sitzung beim CICS-Start über `PLTPI` starten können.

# X Natural als Server

Dieser Teil beschreibt die Verwendung von Natural als Server unter z/OS im Batch-Modus und unter dem TP-Monitor CICS.

| | |
|---|---|
| **Natural als Server unter z/OS** | Beschreibt, wie Natural als Server in einer Client/Server-Umgebung unter z/OS im Batch-Modus eingesetzt werden kann. |
| **Natural als Server unter CICS** | Beschreibt, wie Natural als Server in einer Client/Server-Umgebung unter dem TP-Monitor CICS eingesetzt werden kann. Beschreibt die Funktionalität und die Installation der Natural CICS-Schnittstelle in einer Server-Umgebung und informiert über Einschränkungen, die in einer solchen Umgebung gelten. |

# 26 Natural als Server unter z/OS


- Funktionalität unter z/OS ..... 236
- Natural-Nukleus-Installation in einer Server-Umgebung ..... 237
- Behandlung von Druck- und Arbeitsdateien mit externen Datasets in einer Serverumgebung ..... 237

Dieses Kapitel gilt nur für z/OS.

## Funktionalität unter z/OS

Natural ist nicht nur eine Programmiersprache, sondern kann auch als Server in einer Client/Server-Umgebung fungieren. Er kann Dienste anbieten, wie z. B. die Ausführung von Natural-Subprogrammen. Ein Teil der Serverfunktionalität ist der weiterentwickelte Batch-Treiber.

Der Client/Server-Kommunikation liegen zahlreiche Protokolle zugrunde, z. B. die Ausführung von Stored Procedures für Db2 und die Ausführung von Remote Procedure Calls, siehe die *Natural RPC (Remote Procedure Call)*-Dokumentation..

### Natural Server Stub

Natural als Server läuft in einer separaten Region oder innerhalb der Server-Subsystem-Region, zum Beispiel für Db2 Stored Procedures. Um Natural als Server auszuführen, ist ein dienstspezifischer Server Stub erforderlich. Dieser Server Stub wird als Teil des Server-Produkts geliefert. Er steuert alle Dienstanforderungen und ist die einzige Schnittstelle zum Natural-Server-Frontend.

Es gibt verschiedene Server Stubs für Db2, für Natural RPC und für weitere Dienste.

### Natural Batch-Treiber

Der Natural Batch-Treiber (z. B. `NATOS` unter z/OS) wurde erweitert, um als umgebungsspezifische Schnittstellenkomponente zu fungieren, die die Natural-Server-Sitzungen verwaltet und umgebungsspezifische Dienste für Natural bereitstellt. Er kann mit dem Server Stub-Modul verlinkt werden oder vom Server Stub als separates Modul geladen werden.

Der Batch-Treiber ist in der Lage, mehrere Sitzungen zu erstellen und zu steuern, indem er Speicher-Threads verwendet, einschließlich Funktionen für die Komprimierung und Dekomprimierung von Threads und die Auslagerung (Rollout) auf externe Speichergeräte.

Wenn der Batch-Treiber vom Server Stub zum ersten Mal aufgerufen wird (während der Server-Initialisierung), werden die Speicher-Threads im Hauptspeicher angelegt. Die Anzahl und Größe der Speicher-Threads wird durch den Server Stub bestimmt. Dann wird eine statische Natural-Sitzung initialisiert. Dazu gehören die Auswertung der Profilparameter und die Zuweisung statischer Speicherpuffer. Der so vorinitialisierte Speicher-Thread wird separat im Hauptspeicher abgelegt. Für jede neue Natural-Sitzung wird dieser initiale „Sitzungsklon“ in den Thread kopiert.

Auf Veranlassung des Server Stub kann eine Sitzung ausgelagert werden, um zu einem späteren Zeitpunkt wieder aufgenommen zu werden. Der **Natural Roll Server** wird vom Treiber genutzt, um den komprimierten Thread-Speicher einer Sitzung zu speichern. Alternativ kann auch der Hauptspeicher zum Speichern des komprimierten Thread-Speichers verwendet werden. In diesem Fall ist die Anzahl der Sitzungen im ausgelagerten Zustand durch die Größe der Region begrenzt.

## Natural-Nukleus-Installation in einer Server-Umgebung

Der Natural-Nukleus und sein Batch-Treiber sind sowohl für Server- als auch für Nicht-Server-Umgebungen konzipiert. Die serverspezifischen Definitionen und Anforderungen finden Sie in der entsprechenden Dokumentation, z.B. *Natural RPC (Remote Procedure Call)* oder *Natural for Db2*.

Wenn die Anzahl der Sitzungen nicht auf eine geringe Anzahl beschränkt ist und der Servertyp das Session-Rollout unterstützt, muss der Natural Roll Server installiert und vor der Initialisierung des Servers gestartet werden. Stellen Sie dazu sicher, dass der Parameter `SUBSID` im Natural-Parametermodul auf den richtigen Wert gesetzt ist. Für den Server muss die Adabas-Link-Schnittstelle (`ADALNK`) generiert werden, damit neben dem Server auch `ADALNK` reentrant ist.

Sie können einen lokalen oder einen globalen **Natural Buffer Pool** verwenden. Wenn Sie einen lokalen Buffer Pool definieren, wird er von allen Sitzungen innerhalb der Server-Region gemeinsam genutzt.

Wenn eine logische Druck- oder Arbeitsdateinummer für die Verarbeitung innerhalb einer Servers-itzung verwendet werden soll, muss sie beim Start der Sitzung mit einer Zugriffsmethode verknüpft werden. Dies kann im Natural-Parametermodul mit den Makros `NTWORK` und `NTPRINT` geschehen, wie im folgenden Beispiel, wenn Sie die Gesamtheit aller möglichen Druck- und Arbeitsdateinummern zulassen wollen:

```
NTPRINT (1-31),AM=STD,OPEN=ACC,DEST=*
NTWORK (1-32),AM=STD,OPEN=ACC,DEST=*
```

Der Subparameter `DEST=*` definiert die generische DD-Namensgenerierung während der ersten `OUTPUT`-Klausel eines `DEFINE WORK FILE`- oder `DEFINE PRINTER`-Statement (siehe unten). Der Subparameter `OPEN=ACC` vermeidet das Voraböffnen der Dateien beim Programmstart. Das Öffnen wird beim ersten Zugriff auf die Datei veranlasst.

## Behandlung von Druck- und Arbeitsdateien mit externen Datasets in einer Serverumgebung

Wenn viele gleichzeitige Sitzungen in einer Region laufen, kann es zu Ressourcenkonflikten mit externen Druck- und Arbeitsdateien kommen. Die logischen Namen (DD-Namen) für Druck- und Arbeitsdateien werden durch den Subparameter `DEST` des Makros `NTPRINT` bzw. `NTWORK` oder seine dynamischen Entsprechungen im Profilparameter `PRINT` bzw. `WORK` definiert (Standardwerte `CMPRT`*nn* und `CMWKF`*nn*). Für die normale Natural-Batch-Verarbeitung werden diese Dateien in der JCL durch einen logischen (DD) und einen physischen Dataset-Namen definiert.

DD-Namen sind jedoch vom Betriebssystem für die ausschließliche Verwendung durch eine Task bzw. eine Sitzung reserviert, d.h. wenn `CMWKF01` von einer Sitzung zur Verarbeitung geöffnet wird,

kann keine andere Sitzung diese Datei verwenden, bis sie wieder geschlossen wird. Andere Sitzungen erhalten eine Fehlermeldung, wenn sie versuchen, die Datei zu öffnen.

In einer Serverumgebung werden alle Druck- und Arbeitsdateianforderungen von einer speziellen E/A-Subtask bearbeitet. Dies gewährleistet die Integrität der Datensätze und vermeidet Ressourcenkonflikte. Es ermöglicht die gemeinsame Nutzung von Druck- und Arbeitsdateien über Natural-Sitzungsgrenzen hinweg, d. h. mehrere Sitzungen können gleichzeitig auf dieselbe Datei zugreifen. Dies gilt nur für Druck- und Arbeitsdateien, deren DD-Name mit CM beginnt. Alle anderen Dateien werden als exklusiv betrachtet und können nicht gemeinsam genutzt werden.

Für die exklusive Nutzung von Druck- und Arbeitsdateien bietet Natural die folgenden beiden Funktionen zur Unterstützung von Druck- und Arbeitsdateien in einer Serverumgebung (beide erfordern eine spezielle Implementierung in den Natural-Anwendungsprogrammen für die Serverumgebung):

- Statements `DEFINE WORK FILE` oder `DEFINE PRINTER`, `OUTPUT`-Klausel und
- dynamische Dataset-Zuordnung (Anwendungsprogrammierschnittstelle USR2021N, siehe Dienstprogramm SYSEXT).

Die `OUTPUT`-Klausel der `DEFINE WORK FILE`- und `DEFINE PRINTER`-Statements kann verwendet werden,

- um den logischen DD-Namen für eine Arbeits- oder Druckdatei zu definieren oder
- um den physischen Dataset-Namen zu definieren oder
- um eine Ausgabe-Spool-Klasse zu definieren.

Wenn ein DD-Name angegeben wird, prüft die Zugriffsmethode, ob der Dataset zugeordnet ist. Wenn nicht, wird ein Fehler ausgegeben. Der Dataset kann von jedem Natural-Programm mit dem Subprogramm USR2021N aus der Library `SYSEXT` zugeordnet werden.

Wird ein physischer Dataset-Name oder eine Spool-Datei-Klasse angegeben, weist die Zugriffsmethode den Dataset während der Ausführung des `DEFINE ...`-Statements dynamisch selbst zu. Um sicherzustellen, dass ein eindeutiger DD-Name verwendet wird, sollte `DEST=*` im **Natural-Parametermodul** vordefiniert werden. Dadurch werden DD-Namenskonflikte vermieden.

Wenn die Anwendung die Anwendungsprogrammierschnittstelle USR2021N verwendet, kann sie einen Stern (*) als Wert für die DD-Namensvariable angeben, um einen eindeutigen DD-Namen von der Zugriffsmethode zurückzuerhalten. Dieser DD-Name kann für ein nachfolgendes `DEFINE ...`-Statement verwendet werden.

Standardmäßig werden die Zugriffseigenschaften des Server-Jobs für Druck- und Arbeitsdateien verwendet. Einige Servertypen, z. B. Natural Development Server und Natural RPC, unterstützen Impersonation, d. h. die Zugriffseigenschaften des individuellen Client-Kontos werden für exklusive Druck- und Arbeitsdateien verwendet. Weitere Informationen finden Sie in dem entsprechenden Abschnitt Ihrer Server-Dokumentation.

# 27 Natural als Server unter CICS


- Funktionalität unter CICS ..... 240
- Installation der Natural CICS-Schnittstelle in einer Server-Umgebung ..... 240
- Einschränkungen ..... 241

Dieses Dokument gilt nur unter CICS.

Siehe auch

- *Natural unter CICS* in der *TP Monitor Interfaces*-Dokumentation
- *Natural RPC (Remote Procedure Call)*-Dokumentation

## Funktionalität unter CICS

### Natural als Server

Natural ist nicht nur eine Programmiersprache, sondern kann auch als Server in einer Client/Server-Umgebung fungieren. Es kann Dienste (Services) anbieten, wie z.B. die Ausführung von Natural-Subprogrammen. Der Client/Server-Kommunikation liegen zahlreiche Protokolle zugrunde, beispielsweise die Ausführung von Stored Procedures für Db2 (siehe *Natural for Db2*) und die Ausführung von Remote Procedure Calls (siehe *Natural RPC (Remote Procedure Call)*).

### Natural Server Stub

Natural als Server läuft in einer separaten Region oder innerhalb der Server-Subsystem-Region, zum Beispiel für Db2 Stored Procedures. Um Natural als Server auszuführen, ist ein dienstspezifischer Server Stub erforderlich. Dieser Server Stub wird als Teil des Server-Produkts geliefert. Er steuert alle Dienstanforderungen und ist die einzige Schnittstelle zum Natural-Server-Frontend.

Es gibt unterschiedliche Server Stubs für Db2, für RPC und für andere Dienste.

## Installation der Natural CICS-Schnittstelle in einer Server-Umgebung

Bei der Installation der Natural CICS-Schnittstelle in einer Natural-Server-Umgebung sind keine besonderen Definitionen erforderlich. Es gibt keine Anforderungen an den Thread-Typ oder die Art des Auslagerns (CICS Roll Facilities oder Roll Server).

Tatsächlich können sich Natural-Server-Sitzungen eine Natural-Umgebung unter CICS mit „normalen“, z. B. terminalgebundenen Natural-Sitzungen teilen. Der Unterschied besteht darin, dass im Falle einer Natural-Server-Sitzung die Natural-CICS-Schnittstelle es nicht mit einer Hauptvorrichtung, z. B. einem Terminal oder Drucker, sondern mit einem Server-Stub zu tun hat. In Bezug auf CICS ist eine Natural-Server-Sitzung eine Reihe von asynchronen CICS-Tasks, und der Sitzungskontext (Sitzungsneustartdaten) wird vom Server-Stub unter Verwendung einer eindeutigen 8-Byte-Sitzungskennung verwaltet.

## Einschränkungen

Die folgenden Einschränkungen gelten, wenn Natural als Server unter CICS verwendet wird:

1. Natural-Server-Sitzungen unter CICS können nur im Pseudo-Conversational-Mode laufen. Eine Natural-Server-Sitzung kann nicht im Conversational Mode laufen, da die Natural-CICS-Schnittstelle die Kontrolle immer an den Server Stub zurückgeben muss. Daher wird `PSEUDO=ON` für Natural-Server-Sitzungen unter CICS erzwungen. Aus demselben Grund wird `RELO=ON` für Natural-Server-Sitzungen mit `TYPE=GETM`-Threads erzwungen.
2. Bei 3GL-Programmen, die von Natural aufgerufen werden, sollten die Tatsache berücksichtigt werden, dass Natural-Server-Sitzungen asynchron in CICS laufen, d.h. es ist kein CICS-Terminal (TCTTE) verfügbar.
3. Der Profilparameter `ADAMODE` sollte auf `1` oder `2` gesetzt werden, da Adabas sonst für jeden Dialogschritt der Natural-Server-Sitzung eine andere UQE-ID erzeugt.
4. Der Profilparameter `PROGRAM` oder äquivalente Einstellungen des Backend-Programms durch Natural werden nicht beachtet, da der logische Ablauf bei der Sitzungsbeendigung von der Natural-CICS-Schnittstelle zum Server Stub nicht durch ein mögliches Backend-Programm unterbrochen und/oder verfälscht werden darf.
5. Vorsicht ist geboten bei der Verwendung des Parameters `TERMVAR` mit dem Wert `&TID` im Makro `NTCICSP` bei der Einstellung des Dateinamens für Natural-Druck- und Arbeitsdateien: Da eine Natural-Server-Sitzung asynchron abläuft, gibt es keine (eindeutige) Terminalkennung oder einen anderen eindeutigen vierstelligen Sitzungsidentifikator, der einzufügen wäre. In CICS/TS 1.3 und höher verwendet die CICS-Schnittstelle beim Umgang mit dem CICS-Zwischenspeicher für solche Natural-Druck- und -Arbeitsdateien intern die `QNAME`-Option, d. h. es wird intern ein 16 Byte langer Name für die Warteschlange des Zwischenspeichers verwendet (die 8 Byte lange eindeutige Server-Sitzungskennung wird an die `DEST`-Angabe der Datei angehängt). Dies bedeutet andererseits, dass auf solche CICS-Warteschlangen für den temporären Speicher nur von der veranlassenden Sitzung aus zugegriffen werden kann.

# XI Natural-Ausführung - Verschiedene Themen

Dieser Teil enthält allgemeine Informationen zur Ausführung von Natural.

**Natural 31-Bit Mode-Unterstützung**
**Unterstützung und Nutzung von Natural- und Nicht-Natural-Objekten**
**Eingabe-/Ausgabe-Geräte**
**Doppel-Byte-Zeichensätze**
**Asynchrone Verarbeitung**

Erklärungen zu den in diesem Kapitel verwendeten Begriffen sind im *Glossar* enthalten..

# 28 Natural 31-Bit Mode-Unterstützung

Generell läuft Natural mit den folgenden Einstellungen:

```
AMODE=31
```

```
RMODE=ANY
```

Ausnahmen hiervon sind in der entsprechenden Umgebungsdokumentation beschrieben.

# 29 Unterstützung und Nutzung von Natural- und Nicht-Natural-Objekten


- Unterstützung für Natural-Objekte aus früheren Natural-Versionen ..... 248
- Konventionen für den Aufruf von Backend-Programmen ..... 248
- LE-Subprogramme ..... 250
- Externe Sortierprogramme ..... 253

## Unterstützung für Natural-Objekte aus früheren Natural-Versionen

Natural-Objekte, die in einer früheren Version von Natural erstellt wurden, können in der aktuellen Natural-Version ausgeführt werden, ohne dass Anpassungen an den Objekten oder ein Konvertierungs- oder Migrationsverfahren erforderlich sind. Dies gilt auch für Objekte, die mit dem Natural Optimizer Compiler katalogisiert wurden.

Ausführliche Informationen zu den unterstützten Natural-Versionen finden Sie unter *Von Natural unterstützte Produkt-Versionen* in der aktuellen Natural-*Freigabemitteilung (Release Notes)*.

## Konventionen für den Aufruf von Backend-Programmen

In diesem Abschnitt werden die Konventionen beschrieben, die für den Aufruf eines Backend-Programms gelten.

**Anmerkung:** Außer unter z/OS im Batch-Modus wird ein angegebenes Backend-Programm nicht aufgerufen, wenn die Natural-Sitzung auf einem Natural Development Server ausgeführt wird.

In diesem Abschnitt werden die folgenden Themen behandelt:

- Aufrufkonventionen für Backend-Programme (Batch-Modus)
- Spezielle Aspekte unter CICS
- Spezielle Aspekte unter IMS TM
- Beispiele für Backend-Programme

### Aufrufkonventionen für Backend-Programme (Batch-Modus)

Wenn der Profilparameter `PROGRAM` angegeben ist (oder während einer Natural-Sitzung durch Aufruf des Subprogramms `CMPGMSET` in der Library `SYSEXTP` dynamisch gesetzt wird), wird ein Backend-Programm aufgerufen, und zwar unabhängig davon, ob die Sitzung normal oder abnormal beendet wurde. Das Backend-Programm wird unter Verwendung der Standard-OS-Linkage-Konventionen aufgerufen und muss die Kontrolle an seinen Aufrufer (Caller) zurückgeben.

Wenn ein Backend-Programm verfügbar ist, gibt Natural keine Meldungen zur Sitzungsbeendigung aus. Benutzer-Rückgabewerte ungleich Null, die über *`operand1`* des Natural `TERMINATE`-Statements angegeben werden, werden durch die Natural-Fehlermeldung NAT9987 angezeigt.

Ein Parameterbereich mit den folgenden Informationen wird an das Backend-Programm übergeben:

- ein Vollwort, das den Natural-System- oder Benutzer-Rückgabecode enthält,
- eine Natural-Abbruchmeldung mit 72 Zeichen,

- ein Vollwort, das die Länge der Natural-Abbruchdaten enthält (oder Null),
- die mit *operand2* des TERMINATE-Statements übergebenen Abbruchdaten (falls vorhanden).

Der Backend-Programm-Parameterbereich ist mindestens 80 Bytes lang. Das Makro NAMBCKP, das ein DSECT-Layout des Backend-Programm-Parameterbereichs enthält, ist in der Natural-Source-Library enthalten und kann von Assembler-Backend-Programmen verwendet werden.

### Spezielle Aspekte unter CICS

Unter CICS werden die Daten der Backend-Programmparameter in der COMMAREA und in der TWA übergeben. In der TWA werden nur 80 Bytes übergeben, die den Rückgabecode und die Meldung enthalten, während das Längenfeld eine Adresse enthält, die auf den vollständigen Backend-Programm-Parameterbereich zeigt. Die gleiche TWA wird auch bereitgestellt, wenn Natural über EXEC CICS LINK aufgerufen wurde. Siehe auch *Natural unter CICS*, *Frontend-Aufruf über LINK* in der *TP-Monitor-Schnittstellen*-Dokumentation.

Wenn im Makro NTCICSP der Schlüsselwort-Subparameter BACKRPL=ALL gesetzt ist (abhängig von der installierten Version der Natural-CICS-Schnittstelle), werden nur die Terminierungsdaten in der COMMAREA übergeben.

### Spezielle Aspekte unter IMS TM

Unter IMS TM sind die Aufrufkonventionen für ein Backend-Programm in einer dialogorientierten Umgebung anders. Dort wird das Backend-Programm über einen Programm-zu-Programm-Schalter aufgerufen, und der Name des Backend-Programms wird als IMS TM-Transaktionscode verwendet. In diesem Fall wird die Natural-Umgebung beendet, bevor der Programm-zu-Programm-Wechsel stattfindet. Siehe *Natural under IMS TM*, *Unterstützung des Natural-Profilparameters PROGRAM* in der *TP-Monitor-Schnittstellen*-Dokumentation.

### Beispiele für Backend-Programme

Die folgende Tabelle enthält eine Reihe von Beispielprogrammen:

**Beispiel für ein Backend-Programm für Batch- und TSO-Umgebungen in COBOL:**

```
LINKAGE  SECTION
  01     BACKEND-PARM-AREA.
  02     TERMINATION-RETURN-CODE            PIC S9(8) COMP.
  02     TERMINATION-MESSAGE                PIC X(72).
  02     TERMINATION-DATA-LENGTH            PIC S9(8) COMP.
  02     TERMINATION-DATA                   PIC X(100)
...
PROCEDURE DIVISION USING BACKEND-PARM-AREA
```

**Beispiel für ein Backend-Programm für Batch- und TSO-Umgebungen in Assembler:**

```
BACKPROG CSECT
         SAVE   (14,12)
         LR     11,15
         USING  BACKPROG,11
         L      2,0(1)
         USING  BCKPARM,2
...
         RETURN (14,12)
BCKPARM  NAMBCKP
         END
```

**Beispiel für ein Backend-Programm für CICS in Assembler:**

```
L        2,DFHEICAP
USING    BCKPARM,2
...
BCKPARM  NAMBCKP
         END
```

**Beispiel-Backend-Programm XNATBACK für den Batch-Modus:**

Ein Beispielprogramm für den Batch-Modus ist unter dem Namen `XNATBACK` in der Natural-Source-Library enthalten. Dieses Programm gibt die Natural-Abbruchmeldung sowohl in `SYSPRINT` als auch auf der Bedienerkonsole aus. Eventuelle Abbruchdaten werden in `SYSPRINT` im Dump-Format ausgegeben.

## LE-Subprogramme

Dieser Abschnitt gilt für z/OS-Batch-Modus, CICS, Com-plete, IMS TM und TSO. Er enthält Informationen darüber, wie Natural Subprogramme der IBM Language Environment (LE) unterstützt.

Folgende Themen werden behandelt:

- Unterstützung für IBM LE-Subprogramme
- Natural-Unterstützung für LE-Subprogrammen aktivieren
- Übergabe von LE-Laufzeitoptionen

- LE-Abbruchbehandlung

### Unterstützung für IBM LE-Subprogramme

Um IBM Language Environment (LE)-Subprogramme zu unterstützen, muss Natural so vorbereitet sein, dass das `CALL`-Statement in der Lage ist, LE-Subprogramme aufzurufen. LE-Subprogramme können statische (Profilparameter `CSTATIC` und `RCA`) oder dynamische Subprogramme von Natural sein.

Dynamische LE-Subprogramme von Natural werden über den Dienst CEEFETCH LE geladen und durch den Dienst CEERELES entsprechend der Einstellung des Profilparameters `DELETE` gelöscht.

### Natural-Unterstützung für LE-Subprogrammen aktivieren

Um IBM Language Environment (LE)-Subprogramme von Natural aus aufrufen zu können, ist Folgendes erforderlich:

1. Bei der Installation der Natural CICS-Schnittstelle muss der umgebungsabhängige Nukleus generiert werden. Siehe dazu die entsprechenden Installationsschritte in *Natural CICS-Schnittstelle auf z/OS installieren* in der *Installation für z/OS* -Dokumentation.

   Bezüglich der LE-Befähigung von Natural unter CICS siehe auch die entsprechenden Installationsschritte und den Abschnitt *Natural CICS-Schnittstelle und IBM Language Environment (LE)* in der *TP-Monitor-Schnittstellen*-Dokumentation.

   Für die LE-Befähigung von Natural unter Com-plete muss der Schlüsselwort-Subparameter `LE370` des Makros `NTCOMP` auf `ON` gesetzt werden (siehe *Parameter-Referenz*-Dokumentation). Siehe auch das Kapitel *IBM Language Environment Considerations* in der *Com-plete*-Dokumentation.
2. Die IBM LE-Laufzeitmodule müssen während des Linkage-Editor-Schritts automatisch aus der IBM LE-Bibliothek eingebunden werden. Es dürfen keine unaufgelösten Externals vorhanden sein, die mit „CEE“ beginnen. Setzen Sie nicht die Linkage-Editor-Option `NCAL`.
3. Unter z/OS Batch, IMS TM und TSO kann Natural auch LE-Hauptprogramme aufrufen, allerdings nur als dynamische Subprogramme. Wenn ein LE-Hauptprogramm dynamisch aufgerufen werden soll, muß dies durch die Angabe von `SET CONTROL 'P=L'` vor dem `CALL`-Statement gekennzeichnet werden. Andernfalls wird die von Natural erzeugte LE-Umgebung durch das LE-Hauptprogramm beendet.

### Übergabe von LE-Laufzeitoptionen

**Unter z/OS Batch und TSO:**

Sie haben drei Möglichkeiten:

1. Sie können LE-Laufzeitoptionen durch Verwendung des Parameters `PARM=` in Ihrer JCL übergeben. Dabei gilt Folgendes:
   - Die Laufzeitoptionen, die an die Hauptroutine übergeben werden, müssen von einem Schrägstrich (/) gefolgt werden, um sie von den Natural-Parametern zu trennen.
   - Wenn Sie innerhalb Ihrer Natural-Parameter einen Schrägstrich verwenden wollen, müssen Ihre Natural-Parameter mit einem Schrägstrich beginnen.

   Beispiel:

   ```
   PARM='/ID=/,...'
   ```

2. Sie können LE-Laufzeitoptionen übergeben, indem Sie den `CEEOPTS`-Eingabedatensatz in Ihrer JCL verwenden. Durch die Verwendung von `CEEOPTS` sind die LE-Laufzeitoptionen auch für alle Subtasks verfügbar. Die Verwendung von `CEEOPTS` ist insbesondere bei einem Natural RPC-Server im Batch-Modus erforderlich.

   Beispiel:

   ```
   //CEEOPTS  DD   *
   POSIX(ON)
   /*
   ```

3. Sie können LE-Laufzeitoptionen definieren, indem Sie das mitgelieferte Quellcode-Modul `NATLEOPT` ändern und neu assemblieren. Wenn Sie zum Beispiel Subprogramme haben, die noch im 24-Bit-Modus laufen, setzen Sie `SYSPARM(RMODE24)` als Parameter für den Assembler, anstatt `NATLEOPT` zu ändern.

   Wenn Sie andere spezifische Anforderungen für Ihre LE-Unterprogramme haben, können Sie die gewünschten LE-Optionen für das `CEEXOPT`-Makro im Quellcode-Modul `NATLEOPT` hinzufügen.

**Unter IMS TM:**

Sie können LE-Laufzeitoptionen übergeben, indem Sie das regionsspezifische Laufzeitoptionen-Lademodul `CEEROPT` in Ihrer `STEPLIB`-Verkettung bereitstellen. Darüber hinaus muss die LE Library Routine Retention Initialization Routine `CEELRRIN` in der `PREINIT`-Liste Ihrer Region-JCL vorhanden sein.

Im Folgenden finden Sie ein Beispiel für die Definition eines `CEEROPT-Lademoduls`, das die Ausführung von `AMODE(24)`-Subprogrammen ermöglicht:

```
CEEROPT  CSECT
CEEROPT  AMODE ANY
CEEROPT  RMODE ANY
         CEEXOPT ALL31=((OFF),OVR),                                  X
               STACK=((128K,128K,BELOW,KEEP,512K,128K),OVR)
         END   CEEROPT
```

### LE-Abbruchbehandlung

Natural unterstützt die LE-spezifische Benutzerfehlerbehandlung, d.h. wenn ein LE-Subprogramm einen Benutzerfehler-Handler definiert hat, erhält dieser Handler die Kontrolle, wenn ein Abbruch, eine Programmprüfung oder eine andere LE-Fehlerbedingung im Subprogramm auftritt. Wenn kein LE-Anwenderfehler-Handler definiert wurde, reagiert Natural entsprechend der Einstellung des Profilparameters `DU`.

In diesem Fall wird eine spezielle Fehlermeldung (NAT0950 bei `DU=OFF` oder NAT9967 bei `DU=ON`) ausgegeben, die die LE-Fehlernummer angibt. Außerdem wird die entsprechende LE-Fehlermeldung auf `CEEMSG` ausgegeben und ein LE-Snap-Dump gemäß der LE-Laufzeitoption `TERMTHDACT` in `CEEDUMP` geschrieben.

**Anmerkung:** Im Falle von `DU=FORCE` wird die Abbruchbehandlung von Natural deaktiviert und die LE-Fehlerbehandlung findet statt, auch wenn zum Zeitpunkt des Abbruchs kein LE-Subprogramm aktiv ist. In diesem Fall wird dringend empfohlen, die LE-Laufzeitoption `TERMTHDACT(UAIMM)` anzugeben, um alle erforderlichen Diagnoseinformationen zu erhalten.

## Externe Sortierprogramme

Dieses Dokument enthält Informationen zur Verwendung externer Sortierprogramme mit Natural.

Die folgenden Themen werden behandelt:

- Unterstützung von externen Sortierprogrammen
- Besondere Aspekte

### Unterstützung von externen Sortierprogrammen

Das Natural-Statement `SORT` kann optional ein externes Sortierprogramm aufrufen, das die eigentliche Sortierung durchführt. Ein externes Sortierprogramm wird verwendet, wenn im Natural-Parametermodul der Schlüsselwort-Subparameter `EXT` des Makros `NTSORT` auf `ON` gesetzt ist.

Natural unterstützt alle externen Sortierprogramme, die der in den Handbüchern des jeweiligen Betriebssystems dokumentierten Sortierschnittstelle entsprechen.

Die Anforderungen (z.B. Platz und Datasets) sind identisch mit denen für die Ausführung eines 3GL-Anwendungsprogramms (z.B. COBOL, PL/I), das das Sortierprogramm des Betriebssystems aufruft, und können je nach verwendetem externen Sortierprogramm variieren.

Die Kommunikation mit dem externen Sortierprogramm erfolgt über die User-Exit-Routinen E15 und E35. Folglich benötigt Natural die Datasets `SORTIN` und SORTOUT nicht.

### Besondere Aspekte

Es können alle externen Sortierprogramme verwendet werden, die die erweiterte Parameterliste unterstützen.

# 30 Eingabe-/Ausgabegeräte


- Terminal-Unterstützung ..... 256
- Unterstützung von Lichtstiften (Light Pens) ..... 256
- Drucker-Unterstützung ..... 258

Dieses Kapitel enthält zusätzliche Informationen zu den von Natural unterstützten Ein-/Ausgabegeräten.

## Terminal-Unterstützung

Natural unterstützt eine Vielzahl von Terminaltypen für die Verwendung mit Großrechnern. In TP-Monitor-Umgebungen, in denen die Informationen zum Terminaltyp nicht automatisch an Natural geliefert werden, können Sie den Natural-Profilparameter `TTYPE` verwenden, damit Natural die entsprechende Konvertierungsroutine zum Betrieb eines bestimmten Terminaltyps aktivieren kann.

Links zu verwandten Themen:

- ***NTDVCE - Terminal-Gerätespezifikationstabelle***
- *Terminal-Kommunikation - Profilparameter sortiert nach Kategorien* (*Parameter-Referenz*-Dokumentation)
- ***NATCONFG-Modul*** (verschiedene Themen zur E/A-Umsetzung)
- *Natural-Terminalkommandos*

## Unterstützung von Lichtstiften (Light Pens)

Die Unterstützung von Lichtstiften wurde durch das Terminalkommando `%RM` verbessert. Dieses Kommando bewirkt, dass alle lichtstiftempfindlichen Felder auf dem Bildschirm schreibgeschützt werden, d.h. der Benutzer kann sie mit einem Lichtstift anwählen, aber ihren Inhalt nicht überschreiben.

Damit ein Feld lichtstiftempfindlich ist, muss es intensiviert (Session-Parameter AD=`I`) oder blinkend (`AD=B`) dargestellt werden, und das erste Zeichen des Feldes muss ein Lichtstift-Funktionszeichen sein (siehe unten). Die Auswahl eines Feldes mit einem Lichtstift bewirkt eine Änderung des Funktionszeichens, daher können Sie die Verarbeitung von mit einem Lichtstift ausgewählten Feldern von den Werten der Funktionszeichen abhängig machen.

Die folgenden Lichtstift-Funktionszeichen sind verfügbar:

| Zeichen | Bedeutung |
|---|---|
| ? | Sie können mehrere Felder auswählen, bevor Sie ENTER drücken. |
| > | Es wurde ausgewählt, und wenn es erneut ausgewählt wird, wird es zu einem Fragezeichen ?. Die Zeichen ? und > wechseln sich ab. |
| & | Sie können nur ein Feld auswählen und es wird als ENTER sowohl für das Feld als auch für das MDT (Modified Data Tag) verwendet. |
| ' ' (leer) | Sie können nur ein Feld auswählen, und es wird nur das MDT angezeigt. |

Bei Funktionszeichen müssen Sie zwischen Auswahlfeldern (?, >) und Achtungsfelder (&, leer oder Null) unterscheiden. Auswahlfelder lösen keine sofortige Datenübertragung aus, so dass Sie mehrere Felder auswählen können. Achtungsfelder führen zu einer sofortigen Aktion.

Die Taste SELECT CURSOR emuliert eine Auswahl mit dem Lichtstift. Wenn Sie den Cursor auf das Feld bewegen, das Sie auswählen möchten, und SELECT CURSOR drücken, wird dieses Feld ausgewählt.

### Beispiel für ein Natural-Programm zur Verwendung eines Lichtstiftes

```
RESET #FIELD-1 (A8)
  #FIELD-2 (A8) #FIELD-3 (A8) #CV-1 (C) #CV-2 (C) #CV-3 (C)
SET KEY ALL
/* SET CONTROL 'RM' IS A TOGGLE. AFTER IT IS EXECUTED ONCE MAKE IT A
/* COMMENT, SO THAT YOU DO NOT TOGGLE IT 'OFF'.
**SET CONTROL 'RM'
REPEAT
  IF *PF-KEY NOT = 'ENTR' AND *PF-KEY NOT = 'PEN' ESCAPE BOTTOM
  MOVE (AD=I CD=YE) TO #CV-1
  MOVE (AD=I CD=RE) TO #CV-2
  MOVE (AD=I CD=BL) TO #CV-3
  MOVE ' FIELD-1' TO #FIELD-1
  MOVE '&FIELD-2' TO #FIELD-2
  MOVE '?FIELD-3' TO #FIELD-3
  INPUT (SG=OFF IP=OFF)
    01/01 #FIELD-1 (CV=#CV-1 AD=M)
    03/01 #FIELD-2 (CV=#CV-2 AD=M)
    05/01 #FIELD-3 (CV=#CV-3 AD=M)
  WRITE 'PF-KEY =' *PF-KEY
  IF #CV-1 MODIFIED WRITE '#CV-1 MODIFIED' #FIELD-1
  IF #CV-2 MODIFIED WRITE '#CV-2 MODIFIED' #FIELD-2
  IF #CV-3 MODIFIED WRITE '#CV-3 MODIFIED' #FIELD-3
LOOP
END
```

# Drucker-Unterstützung

Folgende Themen werden behandelt:

- Drucker-Vorschubsteuerzeichen
- Natural-Unterstützung für Laserdrucker

## Drucker-Vorschubsteuerzeichen

In einem Natural-Programm können Sie Steuerzeichen für den Druckervorschub mit dem Statement `DEFINE PRINTER` wie folgt erzeugen:

```
....
DEFINE PRINTER (n) OUTPUT 'name'
DEFINE PRINTER (n+1) OUTPUT 'CCONTROL'
....
```

Beide `DEFINE PRINTER`-Statements wirken zusammen, so dass alle Natural-Ausgaben für den Drucker (*n*) den normalen Natural-Report-Ausgaberegeln folgen und alle Natural-Ausgaben für den Drucker (*n*+1) auch auf den Drucker (*n*) geschrieben werden. Natural erzeugt für diesen Report kein Druckervorschubsteuerzeichen. Daher ist das erste Zeichen in der Ausgabevariablen das Steuerzeichen.

Mit dieser Methode ist es möglich, Steuerzeichen für Laserdruckersysteme und Kanalvorschubzeichen für Zeilendrucker in einem normalen Natural-Ausgabereport zu mischen.

**Natural-Programmbeispiel für Drucker-Vorschubsteuerzeichen**

```
...
DEFINE PRINTER (1) OUTPUT 'CMPRT01'
DEFINE PRINTER (2) OUTPUT 'CCONTROL'
WRITE (1) 'TEST'
WRITE (2) NOTITLE '+TEST'
MOVE H'5A' TO A(A1)
WRITE (2) A '....'
...
```

Die entsprechenden hexadezimalen Daten in der Spooldatei, beginnend mit Spalte `0`, sind:

```
I..I..I..I..I..I..I..I..I..I..I..I..I..I..I..I
F1 E3 C5 E2 E3
1  T  E  S  T
4E E3 C5 E2 E3
+  T  E  S  T
5A ....
```

CCONTROL ist der Name einer speziellen Druckersteuertabelle, die dem Drucker *n*-1 zugeordnet ist. Dieser Name darf nicht geändert werden.

### Natural-Unterstützung für Laserdrucker

Natural unterstützt Laserdruckersysteme des Typs IBM 3800.

Mit dem Statement DEFINE PRINTER können Sie einen Report für das Druckersystem 3800 steuern und zuordnen. Mit diesem Statement können Sie festlegen, dass die Natural-Druckausgabe für Report 1 an ein Druckersystem 3800 weitergeleitet wird.

```
DEFINE PRINTER (1) OUTPUT 'LAS3800'
 I I => 1-31 for CMPRT01 to CMPRT31
 ....
```

Je nach Einstellung des Parameters INTENS wiederholt Natural jede Zeile bis zu viermal und erkennt die Natural-Attribute AD=D, AD=I, AD=C und AD=V (siehe Session-Parameter AD).

Die erste Zeile enthält in der ersten Spalte den ASA-Steuercode und in der zweiten Spalte das 3800-Font-Steuerzeichen (hexadezimal F0) für den ersten Font. Die Spalten 2 bis *nnn* enthalten die Druckdaten, die nicht mit dem Attribut AD=I, AD=C oder AD=V gekennzeichnet sind.

Die zweite Zeile enthält in der ersten Spalte den ASA-Steuercode + (für Druck ohne Zeilenvorschub) und in der zweiten Spalte das 3800-Font-Steuerzeichen (hexadezimal F1) für den zweiten Font. Die Spalten 2 bis *nnn* enthalten die Druckdaten, die mit AD=I gekennzeichnet sind.

Die dritte Zeile enthält in der ersten Spalte den ASA-Steuercode + (für Druck ohne Zeilenvorschub) und in der zweiten Spalte das 3800-Font-Steuerzeichen (hexadezimal F2) für die dritte Schriftart. Die Spalten 2 bis *nnn* enthalten die Druckdaten, die mit AD=C gekennzeichnet sind.

Die vierte Zeile enthält in der ersten Spalte den ASA-Steuercode + (für Druck ohne Zeilenvorschub) und in der zweiten Spalte das 3800-Font-Steuerzeichen (hexadezimal F3) für den vierten Font. Die Spalten 2 bis *nnn* enthalten die Druckdaten, die mit AD=V gekennzeichnet sind.

Wenn INTENS mit einem Wert kleiner als 4 angegeben wird, werden alle nicht unterstützten Schriften mit dem Hexadezimalzeichen F0 gedruckt.

### Beispiel für ein Natural-Programm zur Verwendung von Laserdruckern

```
....
DEFINE PRINTER (1) OUTPUT 'LAS3800'
WRITE (1) 'FIRST' 'SECOND' (AD=I) 'THIRD' (AD=C) 'FOURTH' (AD=V)
....
```

Die entsprechenden hexadezimalen Daten in der Spooldatei, beginnend mit Spalte `0`, sind:

```
I..I..I..I..I..I..I..I..I..I..I..I..I..I..I..I..I..I..I..I..I..I..I
40 F0 C6 C9 D9 E2 E3 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 (hex)
    0  F I R S T
4E F1 40 40 40 40 40 40 E2 C5 C3 E4 D5 C4 C4 40 40 40 40 40 40 40 (hex)
+ 1                      S E C O N D
4E F2 40 40 40 40 40 40 40 40 40 40 40 40 40 E3 C8 C9 D9 D4 40 40 (hex)
+ 2                                           T H I R D
4E F3 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 C5 (hex)
+ 3                                                            F
```

### Beispiel-JCL für die Verwendung von Laserdruckern

```
....
//xxxx JOB xxxxx,....
.
//xxxxx EXEC PGM= XXXXXX;......
.
// PARM='INTENS=4,XXXX,.......
.
.
//OUT1 OUTPUT PAGEDEF=XXXX,FORMDEF=XXXX,TRC=ON
.                      I             I
.                      I             I => 3800 form definition
.                      I
.                      I => 3800 page definition .
//CMPRT01 DD SYSOUT=Y
//        DCB=(RECFM=FBA,LRECL=133),OUTPUT=*,OUT1
//        CHARS=(WWWW,XXXX,YYYY, ZZZZ)
                  I
                  I => IBM font names
...
```

# 31 Doppel-Byte-Zeichensätze


- Natural-Profilparameter SOSI ..... 262
- Angabe des Ausgabeformats ..... 262
- Parameterdefinitionen für DBCS-Unterstützung ..... 262
- Editor-Profil-Optionen ..... 263
- Prüfung der Eingabedaten ..... 264
- Anpassung der Ausgabedaten ..... 264
- Natural Stack-Daten ..... 265
- Anwendungsprogrammierschnittstellen für die DBCS-Behandlung ..... 265
- Alternatives Textmodul NATTXT2U ..... 266

Dieses Kapitel ist für asiatische Länder relevant, die Doppel-Byte-Zeichensätze (Double Byte Character Sets, DBCS) - im Gegensatz zu Ein-Byte-Zeichensätzen (Single Byte Character Sets, SBCS) - verwenden. Es beschreibt alle in Natural implementierten Funktionen zur Unterstützung von DBCS-Terminals und -Druckern.

Folgende Themen werden behandelt:

## Natural-Profilparameter SOSI

In alphanumerischen Feldern mit gemischten SBCS- und DBCS-Zeichen werden die DBCS-Zeichenketten von den SBCS-Zeichenketten durch Shift-Codes mit der Bezeichnung `SO` (Shift-Out) und `SI` (Shift-In) getrennt. Der Natural-Profilparameter `SOSI` wird verwendet, um die Werte der in der aktuellen Umgebung verwendeten Shift-In- und Shift-Out-Codes an Natural zu übergeben.

Es wird dringend empfohlen, intern die IBM-Zeichen `X'0E'` und `X'0F'` zu verwenden. Mit dieser Technik können alle Anwendungen und Daten auf kompatible Weise gehandhabt werden, was bedeutet, dass ein Netzwerk, das verschiedene Großrechnertypen unterstützt, dennoch dieselben Natural-Anwendungen verwenden und dieselben Daten verarbeiten kann.

Weitere Informationen siehe Profilparameter `SOSI` in der *Parameter-Referenz*-Dokumentation.

## Angabe des Ausgabeformats

Der Natural-Session-Parameter `PM=D` wird zur Definition von reinen DBCS-Feldern verwendet. Ein reines DBCS-Feld darf nur gültige DBCS-Zeichen enthalten. Shift-Out/Shift-In-Zeichen (`SO`/`SI`) sind in einem solchen Feld nicht erlaubt. Um ein Feld mit dem Session-Parameter `PM=D` anzuzeigen, wird für IBM-Terminals das Bildschirmattribut `X'43F8'` hinzugefügt.

## Parameterdefinitionen für DBCS-Unterstützung

Die folgenden Parameter müssen bei der Einrichtung von Natural für die Unterstützung von Doppel-Byte-Zeichensätze angegeben werden:

| Parameter | Erläuterung |
|---|---|
| TS=ON | Wenn keine lateinischen Kleinbuchstaben verfügbar sind, wandelt dieser Parameter alle Natural-Systemausgaben anhand der durch das Makro NTTABL im Modul NATCONFG definierten Umsetzungstabelle um. |
| SOSI=(0E,0E,0F,0F,1) | Legt die DBCS Shift-Out- und Shift-In-Werte für IBM-Hardware fest. |
| LC=ON | Setzt nicht alle Eingabedaten in Großbuchstaben um, was wiederum mögliche DBCS-Eingabedaten zerstören würde. |

Zusätzlich zu TS=ON werden weitere Parameter für die Umsetzung von Meldungen in Großbuchstaben von mehreren Natural-Komponenten zur Verfügung gestellt. Ausführliche Informationen finden Sie unter *Weitere Parameter für die Umsetzung in Großbuchstaben* in der Beschreibung des Profilparameters TS.

## Editor-Profil-Optionen

Wenn Sie in einem der Natural-Editoren DBCS- oder Katakana-Zeichen in halber Breite eingeben wollen, sollten Sie im **Editor-Profil** unter *Allgemeine Standardwerte - General Defaults* die folgenden allgemeinen Standardoptionen für den Editor einstellen, um zu vermeiden, dass Zeichenkonstanten oder Feldnamen, die DBCS- oder Katakana-Zeichen in halber Breite enthalten, ungewollt in Großbuchstaben umgewandelt werden:

| Option | Wert | Erläuterung |
|---|---|---|
| Editing in Lower Case | Y | Kleinbuchstaben im Quellcode werden nicht automatisch in Großbuchstaben umgewandelt.<br><br>Diese Option ist erforderlich, wenn Sie DBCS oder Katakana-Zeichen mit halber Breite verwenden. |
| Dynamic Conversion of Lower Case | N | Jeder Quellcode bleibt so, wie Sie ihn eingeben.<br><br>Diese Option ist erforderlich, wenn Sie Katakana-Zeichen mit halber Breite verwenden. |

Ausführliche Informationen zu den allgemeinen Standardoptionen des Editors finden Sie unter *Allgemeine Standardwerte - General Defaults*. Ausführliche Informationen über das Editorprofil finden Sie unter *Editor-Profil* in der *Editoren*-Dokumentation.

Damit Sie diese Optionen nicht für jeden Benutzer ändern müssen, können Sie das Standardprofil für Ihre Installation mit Hilfe der User-Exit-Routine USR0070P ändern, die auch DBCS unterstützt. Siehe ***USR0070P - User Exit für Editor-Profile*** im Kapitel ***Natural konfigurieren***.

## Prüfung der Eingabedaten

Wenn der Session-Parameter `PM=D` für ein Feld gesetzt ist, wird geprüft, ob die Eingabedaten

- eine gerade Anzahl an Bytes enthalten,
- nur gültige DBCS-Zeichen enthalten,
- keine Shift-Out/Shift-In-Zeichen (`SO`/`SI`) enthalten.

Da die Erkennung von Nicht-DBCS-Zeichen ICU erfordert, wird diese Prüfung nicht durchgeführt, wenn ICU nicht verfügbar ist (d.h. wenn der Profilparameter `CFICU=OFF` gesetzt wurde).

## Anpassung der Ausgabedaten

Wenn ein Fenster zur Benutzerinteraktion angezeigt werden soll, kann das Fenster bereits angezeigte DBCS-Zeichen überlagern, oder das Fenster kann selbst DBCS-Zeichen enthalten, die aufgrund der Fenstergröße abgeschnitten sind. Eine Überlagerung kann auch auftreten, wenn bei einem `INPUT`-Statement die Option `NO ERASE` verwendet wird. Um eine Fehlerhaftigkeit des Bildschirminhalts im Falle einer solchen Überlagerung zu verhindern, werden die folgenden Aktionen durchgeführt, um die Ausgabedaten gegebenenfalls anzupassen:

- Wenn der Session-Parameter `PM=D` für ein Feld gesetzt ist, wird ein verwaistes Byte (d.h. ein einzelnes Byte, das infolge einer teilweisen Überlagerung eines DBCS-Zeichens am Anfang oder Ende der anzuzeigenden Daten übrigbleibt) durch ein Attribut ersetzt. Dadurch wird sichergestellt, dass nur gültige DBCS-Zeichen angezeigt werden.
- Wenn der Profilparameter `SOSI` gesetzt ist, wird der Feldinhalt eines alphanumerischen Feldes, für das `PM=D` nicht angegeben ist, auf Shift-Out/Shift-In-Zeichen (`SO`/`SI`) untersucht. Wird ein Shift-Out-Zeichen (`SO`) gefunden, für das das korrelierende Shift-In-Zeichen (`SI`) fehlt, wird entweder das letzte Zeichen der Ausgabedaten durch ein Shift-In-Zeichen (`SI`) ersetzt oder die letzten beiden Zeichen werden durch ein Shift-In-Zeichen (`SI`) gefolgt von einem Leerzeichen ersetzt. Wird ein Shift-In-Zeichen (`SI`) gefunden, für das das korrelierende Shift-Out-Zeichen (`SO`) fehlt, wird entweder das erste Zeichen der Ausgabedaten durch ein Shift-out-Zeichen (`SO`) ersetzt oder die beiden führenden Zeichen werden durch ein Leerzeichen gefolgt von einem Shift-Out-Zeichen (`SO`) ersetzt. Dadurch wird sichergestellt, dass DBCS-Zeichen ordnungsgemäß von Shift-Out-/Shift-In-Zeichen (`SO`/`SI`) umschlossen werden.

## Natural Stack-Daten

Um eine unbeabsichtigte Interpretation von DBCS-Zeichen als Begrenzungs- oder Steuerzeichen zu vermeiden, sollte die Option `FORMATTED` des Statements `STACK` verwendet werden, wenn die Daten, die auf dem Natural Stack abgelegt werden sollen, DBCS-Zeichen enthalten.

Weitere Informationen:

- Statement `STACK` in der *Statements*-Dokumentation
- Natural Stack im *Leitfaden zur Programmierung*

## Anwendungsprogrammierschnittstellen für die DBCS-Behandlung

Die folgenden Anwendungsprogrammierschnittstellen (API) sind zur Unterstützung der DBCS-Behandlung verfügbar:

- USR4211N - Abrufen von DBCS-Zeichen
- USR4213N - String-Behandlung für DBCS-Unterstützung

Diese APIs sind als Unterprogramme in der Natural Library `SYSEXT` enthalten. Ausführliche Informationen zur Verwendung einer API finden Sie in dem zugehörigen Textobjekt (`USR`*xxxx*`T`). Siehe auch Dienstprogramm SYSEXT.

### USR4211N - Abrufen von DBCS-Zeichen

Die Anwendungsprogrammierschnittstelle USR4211N kann verwendet werden, um Informationen über die Verfügbarkeit von DBCS-Unterstützung und die definierten SOSI-Zeichen zu erhalten.

### USR4213N - String-Behandlung für DBCS-Unterstützung

Mit der Anwendungsprogrammierschnittstelle USR4213N können die folgenden Funktionen ausgeführt werden:

- Konvertierung einer normalen lateinischen Zeichenkette in die entsprechende DBCS-Zeichenkette.
- Konvertierung einer DBCS-Zeichenkette, die nur lateinische Daten enthält, in eine Ein-Byte-Zeichenkette.
- Hinzufügen der aktuellen Shift-Codes am Anfang und am Ende einer Zeichenkette.
- Entfernen von einleitenden und abschließenden Shift-Codes aus einer Zeichenkette.

Die letzten beiden Funktionen können verwendet werden, um entweder native DBCS-Zeichenketten zu erzeugen oder Mixed-Mode-Daten aus nativen DBCS-Zeichenketten zu generieren.

## Alternatives Textmodul NATTXT2U

Der alternative Textmodul `NATTXT2U` enthält bestimmte Schlüsselwörter für die englische Sprache in Großbuchstaben, die im Textmodul `NATTXT2` in gemischter Großschreibung enthalten sind.

`NATTXT2U` sollte in Umgebungen, in denen die Kleinbuchstaben-Codepoints `H'81'` bis `H'A9'` zur Darstellung nationaler Zeichen verwendet werden, anstelle von `NATTXT2` mit dem Natural Nukleus verlinkt werden.

# 32 Asynchrone Verarbeitung


- Kennzeichnung asynchroner Natural-Sitzungen ..... 268
- Behandlung der Ausgabe einer asynchronen Natural-Sitzung ..... 268
- Behandlung unerwarteter oder unerwünschter Eingaben ..... 269
- Weitere Überlegungen zu Profilparametern ..... 269

Dieses Kapitel beschreibt die asynchrone Natural-Verarbeitung, ein Verfahren, das unter allen von Natural unterstützten TP-Monitoren verfügbar ist.

Eine asynchrone Natural-Sitzung ist eine Sitzung, die mit keinem Terminal verbunden ist und daher nicht mit einem Terminalbenutzer interagieren kann. Sie kann verwendet werden, um eine zeitaufwändige Aufgabe „im Hintergrund“ (Background Task) auszuführen, ohne dass der Benutzer auf die Beendigung der Aufgabe warten muss.

Verwandte Themen:

- *Asynchrone Natural-Verarbeitung unter CICS*
- *Asynchrone Natural-Verarbeitung unter Com-plete/SMARTS*

## Kennzeichnung asynchroner Natural-Sitzungen

Um eine Sitzung als asynchron zu kennzeichnen, wird der Natural-Systemvariablen `*DEVICE` der Wert `ASYNCH` zugewiesen.

**Anmerkung:** Der Wert von `*DEVICE` kann durch den Natural-Profilparameter `TTYPE` und durch das Statement `SET CONTROL 'T=xxxx'` geändert werden.

**Anmerkung:** Siehe auch Profilparameter `TTYPE` in der *Parameter-Referenz*- Dokumentation und Terminalkommando `%T=` in der *Terminalkommandos*-Dokumentation.

## Behandlung der Ausgabe einer asynchronen Natural-Sitzung

Da es sich bei einer asynchronen Sitzung um eine Sitzung handelt, die mit keinem Terminal verbunden ist, bedeutet dies, dass die von der Sitzung erzeugten Ausgaben nicht einfach auf dem Bildschirm angezeigt werden können, sondern dass Sie explizit ein Ausgabeziel angeben müssen.

Dieses Ziel geben Sie mit dem Natural-Profilparameter `SENDER` beim Aufruf von Natural an. Die `SENDER`-Zielangabe (Destination) gilt für Hardcopy-Ausgaben und Primär-Reports. Alle weiteren Reports werden so wie bei einer synchronen Online-Sitzung an die mit dem Statement `DEFINE PRINTER` angegebenen Ziele (Destinations) gesendet.

Da auch eine asynchrone Sitzung einen Natural-Fehler auslösen kann, muss das Ziel (Destination), an das eine Natural-Fehlermeldung gesendet werden soll, ebenfalls angegeben werden. Dies geschieht mit dem Natural-Profilparameter `OUTDEST`. Dieser Parameter bietet auch die Möglichkeit, Fehlermeldungen an die Bedienerkonsole zu senden. Nachdem eine Fehlermeldung gesendet wurde, beendet Natural die asynchrone Sitzung.

Die Profilparameter SENDER und OUTDEST sollten entsprechend eingestellt werden, um auf unerwartete Ausgaben der asynchronen Natural-Sitzung vorbereitet zu sein, da sonst die asynchrone Natural-Sitzung in einem solchen Szenario abgebrochen werden kann.

## Behandlung unerwarteter oder unerwünschter Eingaben

Eine asynchrone Natural-Sitzung verfügt nur über den Natural-Stack, um die Namen der auszuführenden Natural-Programme und Natural-Systembefehle einzugeben. Wenn ein Natural-Programm oder ein Natural-Systemkommando mit einem unbehandelten Natural-Fehler fehlschlägt oder wenn der gesamte Natural-Stack voll ist und in den NEXT-Modus wechseln würde, wird die asynchrone Natural-Sitzung mit der Abbruchmeldung NAT9943 beendet.

Je nach verwendetem TP-Monitor und abhängig von der TTYPE-Einstellung wird bei einer INPUT-Anforderung standardmäßig entweder die CLEAR-Taste oder der EOF-Anzeiger an Natural zurückgegeben. Diese Maßnahme hilft, Fehlerschleifen zu vermeiden, wenn ein Programm unbeabsichtigt ein INPUT-Statement ausführt. Die Rückgabe des ENTER-Tastenkennzeichens kann durch ein SET CONTROL 'N'-Statement vor dem INPUT-Statement erreicht werden.

**Tipp:** Sie können Ihre Anwendung mit asynchronen Sitzungen kompatibel machen, indem Sie die Systemvariable *SCREEN-IO entsprechend auswerten.

## Weitere Überlegungen zu Profilparametern

Die folgenden Natural-Profilparameter sollten im Falle einer asynchronen Natural-Sitzung berücksichtigt werden:

| Profilparameter | Kommentar |
|---|---|
| AUTO | Asynchrone Sitzungen können nicht-alphabetische Benutzerkennungen haben. In diesem Fall wird AUTO=ON fehlschlagen. |
| CM | Eine unerwünschte Eingabesituation kann entstehen, wenn die Natural-Sitzung unbeabsichtigt auf die NEXT-Ebene wechselt. Durch die Einstellung CM=OFF wird die Sitzung in diesem Fall sofort beendet. |
| ENDMSG | Die Fehlermeldung NAT9995 (normale Beendigung) kann durch Angabe von ENDMSG=OFF unterdrückt werden. |
| IMSG | Natural-Initialisierungsfehlermeldungen und -Warnungen können durch die Angabe von IMSG=OFF unterdrückt werden. |
| MENU | Asynchrone Sitzungen verfügen nur über den Natural-Stack für Kommandoeingaben. Es wird daher empfohlen, MENU=OFF anzugeben und mit Direktkommandos durch Natural zu navigieren. |

| Profilparameter | Kommentar |
|---|---|
| PLOG | Die dynamische Parameterprotokollierung wird durchgeführt, indem alle Parameter Zeile für Zeile an das SENDER-Ziel (Destination) gesendet werden. |
| PROGRAM | Wenn in Ihrer Installation ein Standard-Backend-Programm oder eine Standard-Backend-Transaktion definiert ist, sollte geprüft werden, ob dieses Programm asynchron laufen kann oder ob es nur mit terminalgebundenen Sitzungen arbeiten soll. Durch die Angabe von PROGRAM=0 wird die Backend-Logik umgangen. |