Natural for z/OS

System-Architektur

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: NATMF-NNATARC-925-20261013DE

Inhaltsverzeichnis

Vorwort ... v
1 Über diese Dokumentation ... 1
Dokumentationskonventionen ... 2
Online-Informationen und Support ... 2
Datenschutz ... 3
I Natural-Architektur ... 5
2 Natural-Nukleus ... 9
Natural-Laufzeitsystem ... 10
Natural-Compiler ... 13
Natural-Kommando-Interpreter ... 17
Konfiguration ... 17
3 Benutzer-Session-Daten ... 21
4 Natural Buffer Pool ... 25
Objekt in den Buffer Pool laden ... 26
Objekt aus dem Buffer Pool entfernen ... 26
Beispiel für das Laden und Ausführen eines Objekts ... 27
Verwandte Themen ... 29
5 Natural-Editoren und Natural-Utilities ... 31
6 TP/OS-Schnittstelle ... 33
Online-Verarbeitung ... 34
Batch-Verarbeitung ... 36
Natural-TP/OS-Schnittstellen ... 38
7 Benutzerschnittstelle ... 39
8 Druckdateien und Arbeitsdateien ... 41
Objekte mit Hilfe von Arbeitsdateien übertragen ... 42
Druckdateien und Arbeitsdateien, Definition und Zugriff ... 43
9 Natural-Systemdateien ... 45
Systemdatei-Arten ... 46
Libraries in Systemdateien ... 47
10 DBMS-Schnittstelle - Datenbankzugriff ... 49
Von Natural unterstützte Datenbankverwaltungssysteme ... 50
Natural-Datenbankabfragesprache ... 51
Spezielle SQL-Statements ... 52
Natural-DDMs ... 52
II NaturalONE-Architektur ... 55
11 Was ist NaturalONE? ... 57
Allgemeines zu NaturalONE ... 58
Grundlegende Funktionen für die Entwicklung von Natural-Anwendungen ... 59
Natural für Ajax / Ajax-Developer ... 62
Optionale Komponenten ... 62
12 Verschiedene Modi für die Entwicklung von Natural-Anwendungen ... 65
Allgemeine Informationen zu NaturalONE ... 66

Lokaler Modus 66
Natural-Server-Modus (Natural Server Mode) 67
13 Verwendung eines Versionskontrollsystems 69
III Natural-SPoD-Architektur 71
14 Natural-SPoD-Architektur 73

Vorwort

Diese Dokumentation beschreibt die Systemarchitektur folgender Entwicklungsumgebungen:

Natural-Architektur	Hauptbestandteile der Natural-Architektur.
NaturalONE-Architektur	Hauptbestandteile der Eclipse-basierten Entwicklungsumgebung für die Entwicklung und Pflege von Natural-Anwendungen und Einführung in NaturalONE: ■ **Was ist NaturalONE?** ■ **Verschiedene Modi für die Entwicklung von Natural-Anwendungen** ■ **Verwendung eines Versionskontrollsystems** Die Einführung in NaturalONE ist ein übersetztes Exzerpt aus der englischen NaturalONE-Dokumentation (Stand: Mai 2022). Weitere Informationen siehe dort.
Natural-SPoD-Architektur	Hauptbestandteile der Architektur von Natural Single Point of Development. SPoD ermöglicht eine zentralisierte Entwicklung für mehrere Plattformen.

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: Variablen, für die Sie situations- oder umgebungsspezifische Werte angeben müssen. Neue Begriffe, wenn sie erstmals im Text auftreten. Verweise auf andere Dokumentationsquellen.
`Nichtproportionale Schrift`	Kennzeichnet: Text, den Sie eingeben müssen. Meldungen, die vom System angezeigt werden. 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.

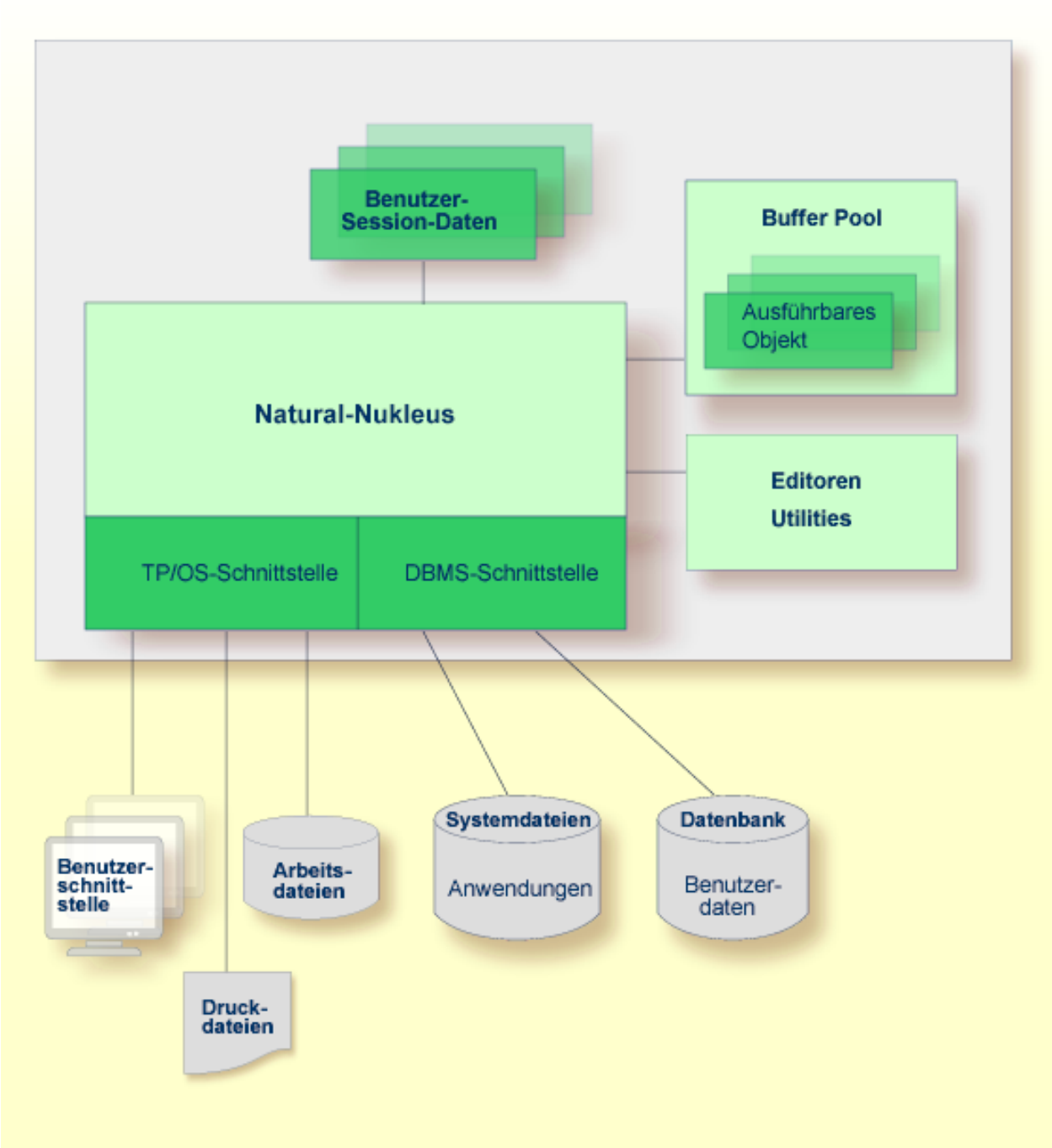
I Natural-Architektur

Der konzeptionelle Aufbau von Natural for z/OS basiert auf dem Prinzip einer offenen Architektur und wird durch die Verwendung von Schnittstellen realisiert. Der Informationsaustausch mit externen Komponenten erfolgt über diese Schnittstellen, und zwar sowohl während der Entwicklung einer Natural-Anwendung als auch bei der Ausführung von Programmen.

Die Natural-Systemfunktionen stehen (soweit relevant) auf allen unterstützten System-Plattformen zur Verfügung. Der Natural-Anwendungsentwickler braucht sich nicht um die komplexen Umgebungszusammenhänge des jeweiligen Systems zu kümmern (Betriebssystem, TP-System, Datenbanksystem, Benutzerumgebung).

Die Natural-Komponenten sind über spezielle systemabhängige Tabellen und Routinen in die Systemumgebung eingebettet und stellen die internen Funktionen mit den erforderlichen Ressourcen zur Verfügung.

Dieses Dokument stellt die wichtigsten Bestandteile von Natural vor und erläutert, wie diese zusammenarbeiten, um die Funktionalität einer Anwendungsentwicklungsumgebung zu realisieren.


Benutzer-
Session-Daten
Buffer Pool
Ausführbares
Objekt
Natural-Nukleus
Editoren
Utilities
TP/OS-Schnittstelle
DBMS-Schnittstelle
Benutzer-
schnitt-
stelle
Arbeits-
dateien
Systemdateien
Anwendungen
Datenbank
Benutzer-
daten
Druck-
dateien

Natural-Nukleus	Die Hauptbestandteile des Natural-Nukleus: Compiler, Laufzeitsystem, Kommando-Interpreter und Konfiguration (Natural-Parameter).
Benutzer-Session-Daten	Temporäre Datenspeicherung in benutzerspezifischen Arbeitsbereichen.
Natural Buffer Pool	Funktionsprinzip des Natural Buffer Pool und das Laden von Objekten.
Natural-Editoren und -Utilities	Editoren und Utilities zum Erstellen und Pflegen einer Natural-Anwendung.
TP/OS-Schnittstellen	Natural-Schnittstellen zum TP-Monitor und zum Betriebssystem.
Benutzerschnittstelle	Von Natural unterstützte Benutzerschnittstellen.
Druckdateien **Arbeitsdateien**	Verwendung von Druckdateien (Print Files) und Arbeitsdateien (Work Files).
Natural-Systemdateien	Speicherung von Objektmodulen in Natural-Systemdateien.
DBMS-Schnittstelle **Datenbankzugriff**	Natural-Schnittstellen zu Datenbankverwaltungssystemen (DBMS). Datenbankzugriffe aus Natural.

2 Natural-Nukleus

- Natural-Laufzeitsystem 10
- Natural-Compiler 13
- Natural-Kommando-Interpreter 17
- Konfiguration 17

Der Natural-Nukleus besteht aus zwei funktionellen Teilen: dem umgebungsabhängigen Nukleus und dem umgebungsunabhängigen Nukleus.

Der umgebungsabhängige Nukleus beinhaltet Komponenten, die von dem in Benutzung befindlichen Großrechner oder TP-System (Online-Schnittstelle) abhängig sind.

Der umgebungsunabhängige Nukleus kann von verschiedenen Großrechnern und TP-Systemen geteilt und von mehreren Benutzern gleichzeitig benutzt werden. Der umgebungsunabhängige Nukleus stellt den Kern von Natural dar, der wichtige Natural-Funktionen wie Kommandointerpretation, Objektkompilierung und –ausführung bereitstellt.

Dieser Abschnitt behandelt die Hauptbestandteile des umgebungsunabhängigen Natural-Nukleus.

Natural-Laufzeitsystem

Das Natural-Laufzeitsystem (Runtime System) funktioniert ähnlich einer virtuellen Maschine, die die zur Ausführung von Objekten in einer mit Natural erstellten Anwendung benötigte Umgebung bereitstellt. Das Natural-Laufzeitsystem interpretiert den Natural-internen Objektcode (binärer Metacode) und führt ihn aus.

Dieser Abschnitt behandelt folgende Themen:

- Objektausführung
- Object Starter und Object Executor

Objektausführung

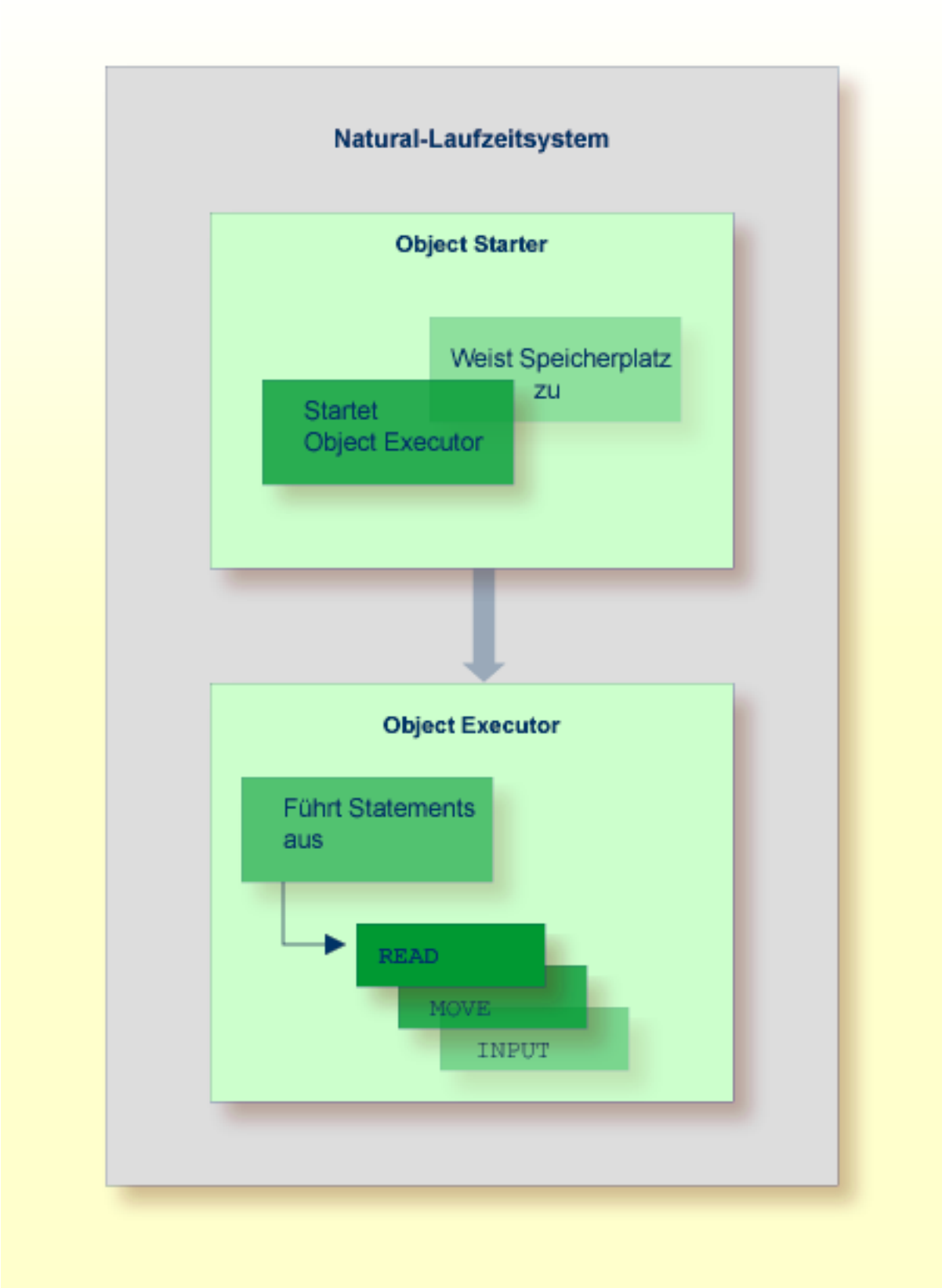
Die Ausführung des Natural-internen Objektcodes erfolgt, wenn die Ausführung eines Natural-Objekts angefordert wird.

Dies geschieht entweder direkt durch einen Benutzer oder indirekt, wenn das Objekt, das zurzeit ausgeführt wird, ein Natural-Statement absetzt, das die Ausführung eines anderen Objekts anfordert. Beispiele: Das Natural-Systemkommando `EXECUTE` bewirkt, dass das mit diesem Kommando angegebene vom Benutzer geschriebene Natural-Programm direkt ausgeführt wird. Das in einem Natural-Programm enthaltene Natural-Statement `CALLNAT` fordert die Ausführung eines Subprogramms an.

Das ***Beispiel für das Laden eines Objekts*** im Abschnitt *Natural Buffer Pool* veranschaulicht den Verarbeitungsablauf bei der Ausführung eines Natural-Programms.

Object Starter und Object Executor

In dem folgenden Diagramm ist die Ausführung eines Objekts durch das Natural-Laufzeitsystem schematisch dargestellt:


Natural-Laufzeitsystem
Object Starter
Weist Speicherplatz zu
Startet Object Executor
Object Executor
Führt Statements aus
READ
MOVE
INPUT

Das Natural-Laufzeitsystem hat zwei Hauptbestandteile: Den Object Starter und den Object Executor.

Der Object Starter ermittelt das auszuführende Objekt im **Natural Buffer Pool** und weist Speicherplatz für die vom Objekt als **Benutzer-Session-Daten** verarbeiteten Daten zu. Schließlich übergibt er die Kontrolle an den Object Executor.

Der Object Executor interpretiert die in dem Objekt enthaltenen Natural-Statements und führt sie der Reihe nach aus. In dem obigen Beispiel verarbeitet der Object Executor zunächst das `READ`-Statement, indem er einen Datenbankaufruf absetzt, die angeforderten Datensätze abruft und dann mit dem `MOVE`-Statement weitermacht.

Verwandte Themen:

- ***Benutzer-Session-Daten***
- ***Natural Buffer Pool***
- ***Beispiel für das Laden eines Objekts*** - *Natural Buffer Pool*

Natural-Compiler

Der Natural-Compiler erzeugt die ausführbare Form des Natural-Quellcode. Beim Natural-Quellcode handelt es sich um einen menschenlesbaren Programmcode, der im Wesentlichen aus einer Abfolge von Natural-Statements besteht. (Informationen zu den Natural-Statements und Hinweise zur Programmierung finden Sie in der *Statements*-Dokumentation bzw. im *Leitfaden zur Programmierung*.)

Der Natural-Compiler liest den Quellcode aus dem Arbeitsbereich. Dieser Quellcode ist Teil der Benutzer-Session-Daten. Er wird mit dem Natural-Systemkommando `READ` oder `EDIT` in den Arbeitsbereich geladen. Der Compiler prüft die Syntax des Quellcode auf Korrektheit und erzeugt dann den Natural-internen Objektcode, der vom **Natural-Laufzeitsystem** interpretiert und ausgeführt wird.

Mit Hilfe des entsprechenden Natural-Systemkommandos kann der Benutzer den Quellcode entweder kompilieren und ausführen oder kompilieren und speichern. Der folgende Abschnitt behandelt die verschiedenen Objektmodularten und die zur Objektkompilierung und –ausführung zur Verfügung stehenden Natural-Systemkommandos.

- Katalogisiertes Objekt
- Source-Objekt
- Kommandos zum Kompilieren

- Beispiel für eine Kompilierung

Katalogisiertes Objekt

Ein katalogisiertes Objekt ist die ausführbare (kompilierte) Form eines Natural-Objekts. Es wird vom Natural-Compiler erzeugt und als Objektmodul in einer Natural-Systemdatei gespeichert. Unter dem Katalogisieren eines Objekts versteht man das Kompilieren des Quellcodes und das Erstellen eines katalogisierten Objekts. Erzeugt wird ein katalogisiertes Objekt mit Hilfe des Natural-Systemkommandos `CATALOG` oder `STOW`.

Zur Ausführungszeit wird das katalogisierte Objekt in den Natural Buffer Pool geladen und vom Natural-Laufzeitsystem ausgeführt. Natural-Objekte können nur ausgeführt werden oder sich gegenseitig referenzieren, wenn sie als katalogisierte Objekte in einer der Natural-Systemdateien gespeichert wurden.

Ein katalogisiertes Objekt kann nicht geändert oder dekompiliert werden.

Source-Objekt

Ein Source-Objekt (oder ein gespeichertes Objekt) enthält die menschenlesbare Form des Natural-Quellcode. Der Quellcode wird mit Hilfe des Natural-Systemkommandos `SAVE` oder `STOW` als Source-Objekt in einer der Natural-Systemdateien gespeichert.

Um den in einem Source-Objekt enthaltenen Quellcode auszuführen, müssen Sie den Quellcode kompilieren, um generierten Objektcode zu erzeugen, der vom Natural-Laufzeitsystem interpretiert und ausgeführt werden kann.

Kommandos zum Kompilieren

Natural bietet verschieden Systemkommandos zum Kompilieren des Quellcodes. Je nach verwendetem Systemkommando wird beim Kompilieren noch eine der folgenden Aktionen ausgeführt:

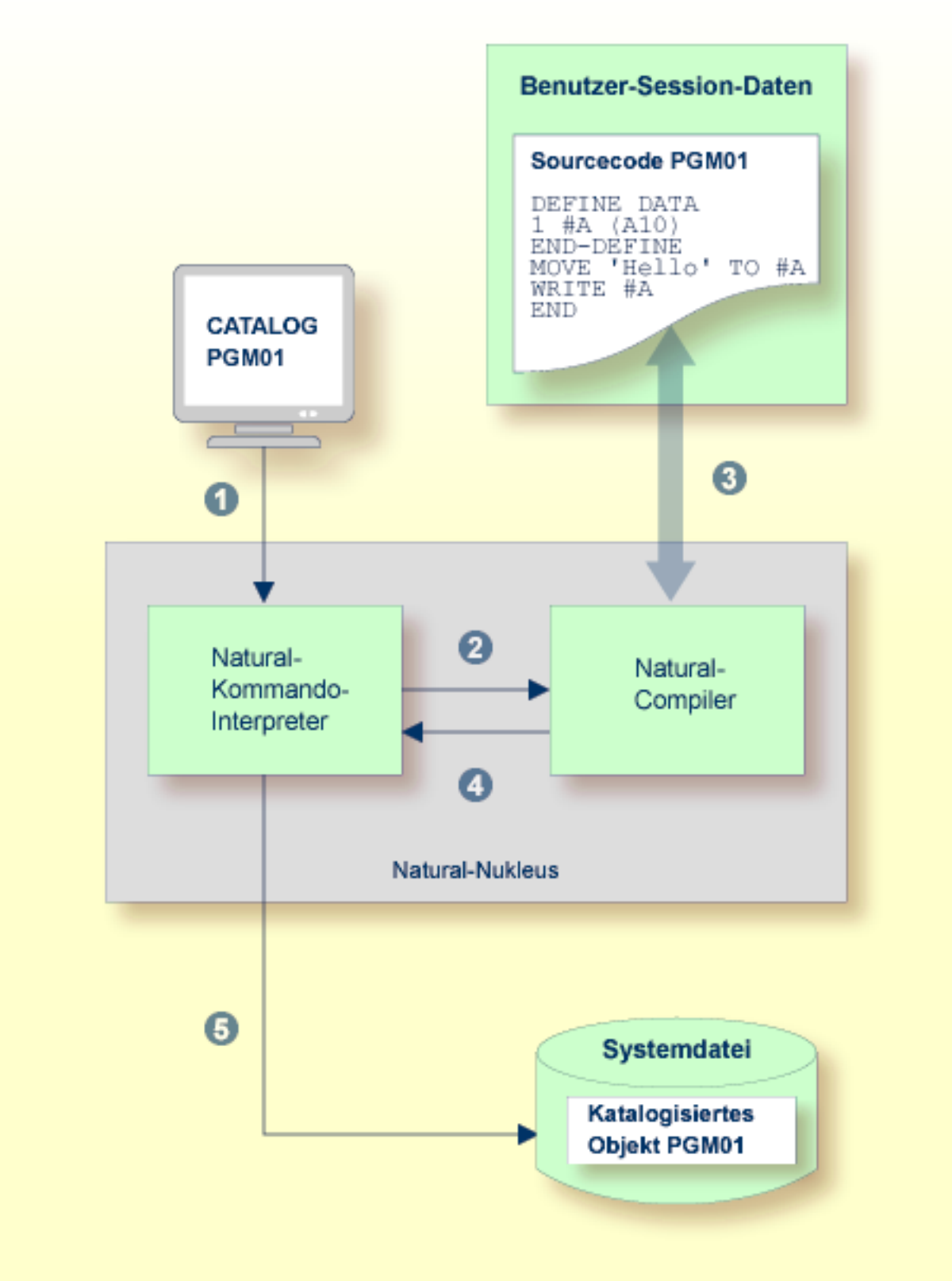
Aktion	Systemkommando
Quellcode kompilieren. Syntaxprüfung durchführen und Objektcode generieren. Der generierte Objektcode wird nicht als Objektmodul in einer Natural-Systemdatei gespeichert.	`CHECK`
Quellcode kompilieren. Nach erfolgreicher Kompilierung den generierten Objektcode als **katalogisiertes Objekt** in einer Natural-Systemdatei speichern.	`CATALOG`
Quellcode kompilieren. Nach erfolgreicher Kompilierung den generierten Objektcode als **katalogisiertes Objekt** in einer Natural-Systemdatei speichern. Außerdem den ursprünglichen Quellcode als separates **Source-Objekt** in einer Natural-Systemdatei speichern.	`STOW`
Quellcode eines Natural-Objekts des Typs Programm kompilieren. Nach erfolgreicher Kompilierung den generierten Objektcode sofort ausführen. Der generierte Objektcode wird nicht als Objektmodul in einer Natural-Systemdatei gespeichert.	`RUN`

Verwandte Themen:

- *Systemkommandos*-Dokumentation
- *Objekte zum Erstellen und Pflegen von Natural-Anwendungen - Leitfaden zur Programmierung*

Beispiel für eine Kompilierung

Das folgende Diagramm veranschaulicht den Verarbeitungsablauf beim Kompilieren des Quellcodes mit Hilfe des Systemkommandos `CATALOG`:


Benutzer-Session-Daten
Sourcecode PGM01
DEFINE DATA
1 #A (A10)
END-DEFINE
MOVE 'Hello' TO #A
WRITE #A
END
CATALOG
PGM01
1
3
Natural-
Kommando-
Interpreter
2
Natural-
Compiler
4
Natural-Nukleus
5
Systemdatei
Katalogisiertes
Objekt PGM01

Legende

1. Der Benutzer setzt das Natural-Systemkommando `CATALOG PGM01` ab, mit dem er die Kompilierung und Speicherung des zurzeit im Arbeitsbereich befindlichen Objektcodes als **katalogisiertes Objekt** mit dem Namen `PGM01` anfordert.
2. Der Natural-Kommando-Interpreter interpretiert das Kommando `CATALOG` und leitet die Anforderung an den Natural-Compiler weiter.
3. Der Natural-Compiler liest die zurzeit im Arbeitsbereich befindlichen Natural-Statements, prüft die Syntax auf Korrektheit und erzeugt dann den Objektcode.
4. Der Natural-Compiler gibt die Kontrolle an den Natural-Kommando-Interpreter zurück.
5. Der Natural-Kommando-Interpreter speichert den generierten Objektcode als **katalogisiertes Objekt** unter dem Namen `PGM01` in der aktuellen Natural-Systemdatei.

Natural-Kommando-Interpreter

Der Natural-Kommando-Interpreter prüft das in einer der Natural-Eingabeaufforderungszeilen eingegebene Kommando und führt es aus.

Wenn das Add-On-Produkt Natural Security installiert ist, kann der Administrator die Ausführung bestimmter Kommandos auf einzelne Benutzer oder eine Benutzergruppe beschränken.

Verwandte Themen:

- *Systemkommandos*-Dokumentation
- *Natural Security*-Dokumentation

Konfiguration

Die Konfiguration einer Natural-Systemumgebung wird mit Natural-Parametern verwaltet.

Diese Parameter dienen zur Standardisierung und Automatisierung der Entwicklungs- und Produktionsvorgänge und zur Anpassung der standardmäßig vorhandenen Einstellungen an die Bedürfnisse einzelner Benutzer. Mittels eines Natural-Parameters können zum Beispiel die Standardeinstellungen für die Report-Erstellung vorgenommen, die Größe eines Reports oder die Größe des erforderlichen Speicherplatzes, z.B. des Arbeitsbereichs eines Editors, festgelegt werden.

Die meisten Eigenschaften einer Natural-Umgebung sind schon vor der Produktauslieferung festgelegt worden. Ihr Natural-Administrator kann verschiedene Standardvorgaben konfigurieren, die für alle Benutzer in Ihrem Unternehmen gültig sind. Jeder Benutzer kann seinerseits die Ein-

stellungen an seine Bedürfnisse anpassen, indem er die Standard-Umgebungseinstellungen mittels eines dynamischen Profilparameters oder eines Session-Parameters überschreibt.

Der folgende Abschnitt behandelt folgende Themen:

- Profilparameter
- Session-Parameter
- Parametrisierungsebenen

Profilparameter

Profilparameter können statisch oder dynamisch angegeben werden.

Statische Parameter werden anlässlich der Natural-Installation im Natural-Parametermodul angegeben. Diese Angaben dienen als Standardvorgaben für jede Natural-Session.

Verwandte Themen:

- *Profilparameter - Parameter-Referenz*-Dokumentation
- *Profile Parameter Usage - Operations*-Dokumentation
- *SYSPARM Utility - Utilities*-Dokumentation

Session-Parameter

Session-Parameter können in einer aktiven Natural-Session und/oder in einem Natural-Objekt angegeben werden. Hauptzweck der Session-Parameter ist die Steuerung der Ausführung von Natural-Programmen.

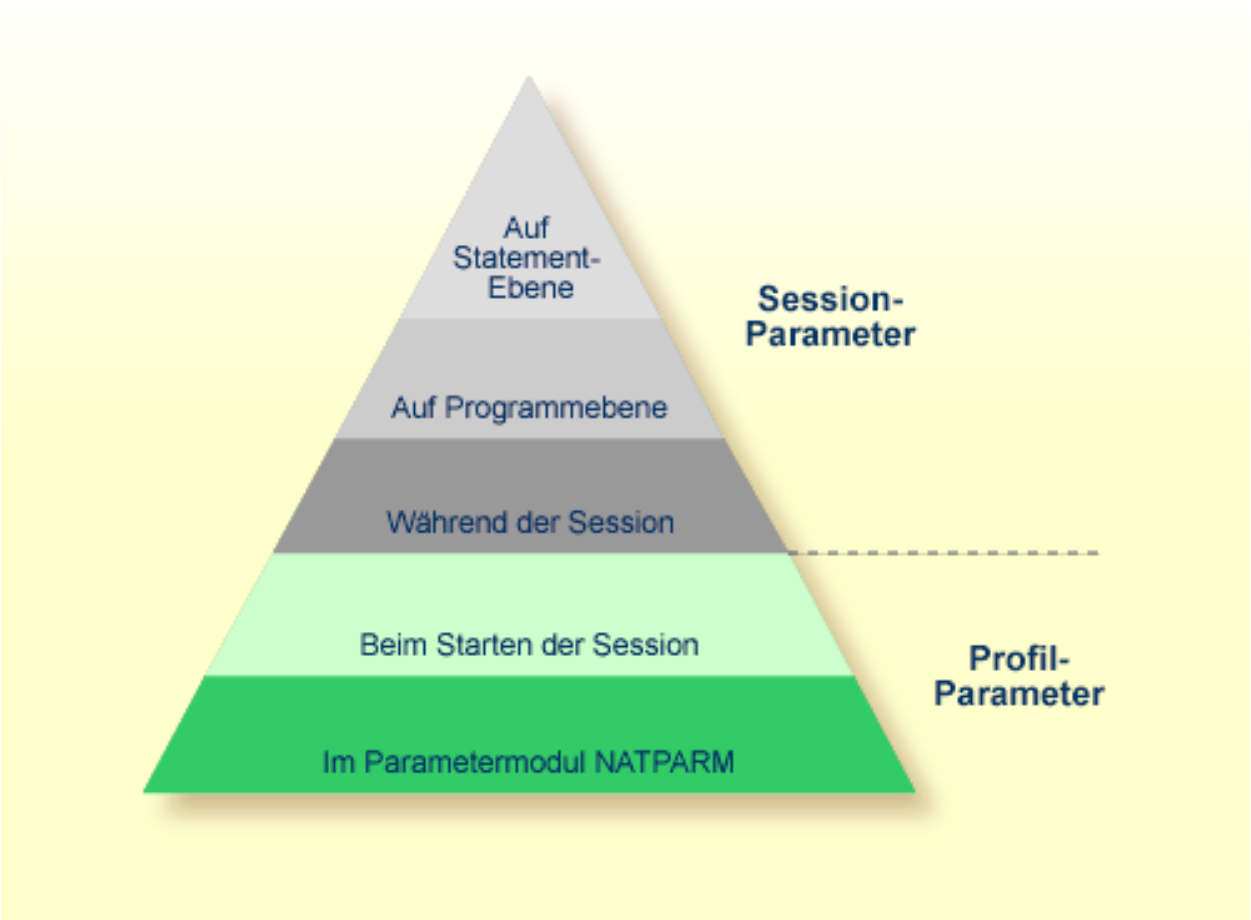
Verwandtes Thema:

Session-Parameter - Parameter Referenz-Dokumentation

Parametrisierungsebenen

Die Ebenen, auf denen Natural parametrisiert werden kann, sind hierarchisch strukturiert. Ein auf einer höheren Ebene gesetzter Parameterwert hat Vorrang vor einem auf einer niedrigeren Ebene gesetzten Wert. Wenn zum Beispiel ein Parameter dynamisch angegeben wird, hat der neue Parameterwert Vorrang vor der entsprechenden statischen Parameterangabe, die im Natural-Parametermodul vorhanden ist.

Das folgende Diagramm veranschaulicht, wann ein Parameter gesetzt werden kann und zeigt die Natural-Parameterhierarchie mit der niedrigsten Ebene an der Basis und der höchsten Ebene an der Spitze der Pyramide:



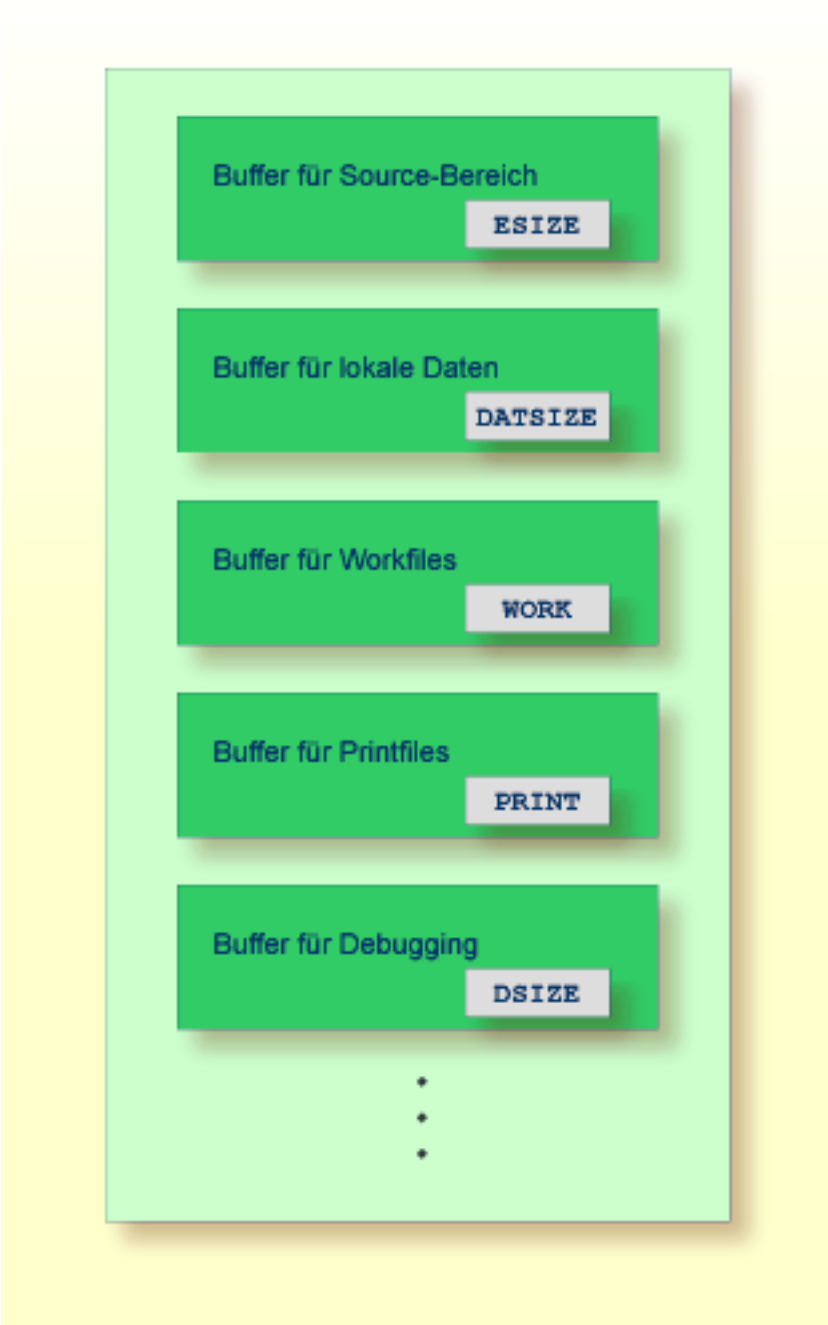
Verwandtes Thema:

Natural Parameter Hierarchy - Operations-Dokumentation

3 Benutzer-Session-Daten

Jeder Natural-Benutzer-Session (Online oder Batch) wird ein separater Arbeitsbereich zugewiesen, der die Daten dieser Session enthält. Die Daten einer Natural-Benutzer-Session werden für die Dauer der Online- oder Batch-Session gespeichert.

Der Arbeitsbereich umfasst mehrere Natural-Buffer. Ein Buffer ist ein Zwischenspeicher, in dem ein bestimmter Datenblock, z. B. in einem Natural-Programm referenzierte Variablendaten, bereit gestellt werden.



Wie in der obigen Abbildung dargestellt, kann die Größe eines Buffers mit einem Natural-Parameter angegeben werden. Eine Ausnahme bilden die internen oder variablen Natural-Buffer. Beispielsweise wird mit dem Natural-Profilparameter `ESIZE` die Größe des Speichers festgelegt, der einem Natural-Editor zum Editieren des Quellcodes zugewiesen ist. Der Profilparameter `DATSIZE` bestimmt zur Ausführungszeit die Größe des Speichers zur Aufnahme lokaler Daten zu einem Natural-Programm (und zu untergeordneten Objekten, die von dem Programm aufgerufen werden).

Natural bietet auch statistische Informationen zur Nutzung der der aktuellen Session zugewiesenen Buffer, zu ihrer Größe und zum tatsächlich genutzten Zwischenspeicherplatz; siehe die entsprechenden Querverweise unter *Verwandte Themen* weiter unten.

Verwandte Themen:

- *Buffer Usage Statistics - BUS*, *Allgemeine SYSTP-Funktionen*, *SYSTP Utility*, *Utilities*-Dokumentation
- *BUS - Systemkommandos*-Dokumentation

4 Natural Buffer Pool

- Objekt in den Buffer Pool laden 26
- Objekt aus dem Buffer Pool entfernen 26
- Beispiel für das Laden und Ausführen eines Objekts 27
- Verwandte Themen 29

Der Natural Buffer Pool mit gemeinsam genutztem Speicher dient als Zwischenspeicher, in den Natural **katalogisierte Objekte** (z.B. Programme) aus einer Natural-Systemdatei zur anschließenden Ausführung durch das Natural-Laufzeitsystem und/oder zur Objektkompilierung durch den Natural-Compiler lädt.

Da Natural ablauf-invarianten Natural-Objektcode generiert, kann eine einzige Kopie eines Naturalobjekts von mehr als einem Benutzer zeitgleich ausgeführt werden. Dazu wird jedes Objekt nur einmal von einer der Natural-Systemdateien in den Natural Buffer Pool geladen, anstatt von jedem Benutzer, der das Objekt anfordert, geladen zu werden.

Dieser Abschnitt behandelt grundlegende Gesichtspunkte für den Betrieb des Buffer Pools und veranschaulicht den Objektladevorgang.

Objekt in den Buffer Pool laden

Das Laden des Objekts in den Buffer Pool wird ausgelöst, wenn ein ausführbares Natural-Objekt zur Ausführung angefordert wird.

Wird die Ausführung eines Objekts angefordert, dann wird das entsprechende **katalogisierte Objekt** von der **Natural-Systemdatei**, in der es gespeichert ist, gelesen und in den Buffer Pool geladen, in dem es durch das **Natural-Laufzeitsystem** ausgeführt werden kann.

Neben ausführbaren Natural-Objekten werden auch nicht ausführbare Objekte des Objekttyps Data Area (Local, Global, Parameter) in den Buffer Pool geladen, wenn ein Natural-Objekt, das eine solche Data Area referenziert, katalogisiert oder neu katalogisiert (kompiliert) wird.

Verwandtes Thema:

Objektausführung - *Natural-Laufzeitsystem, Natural-Nukleus*

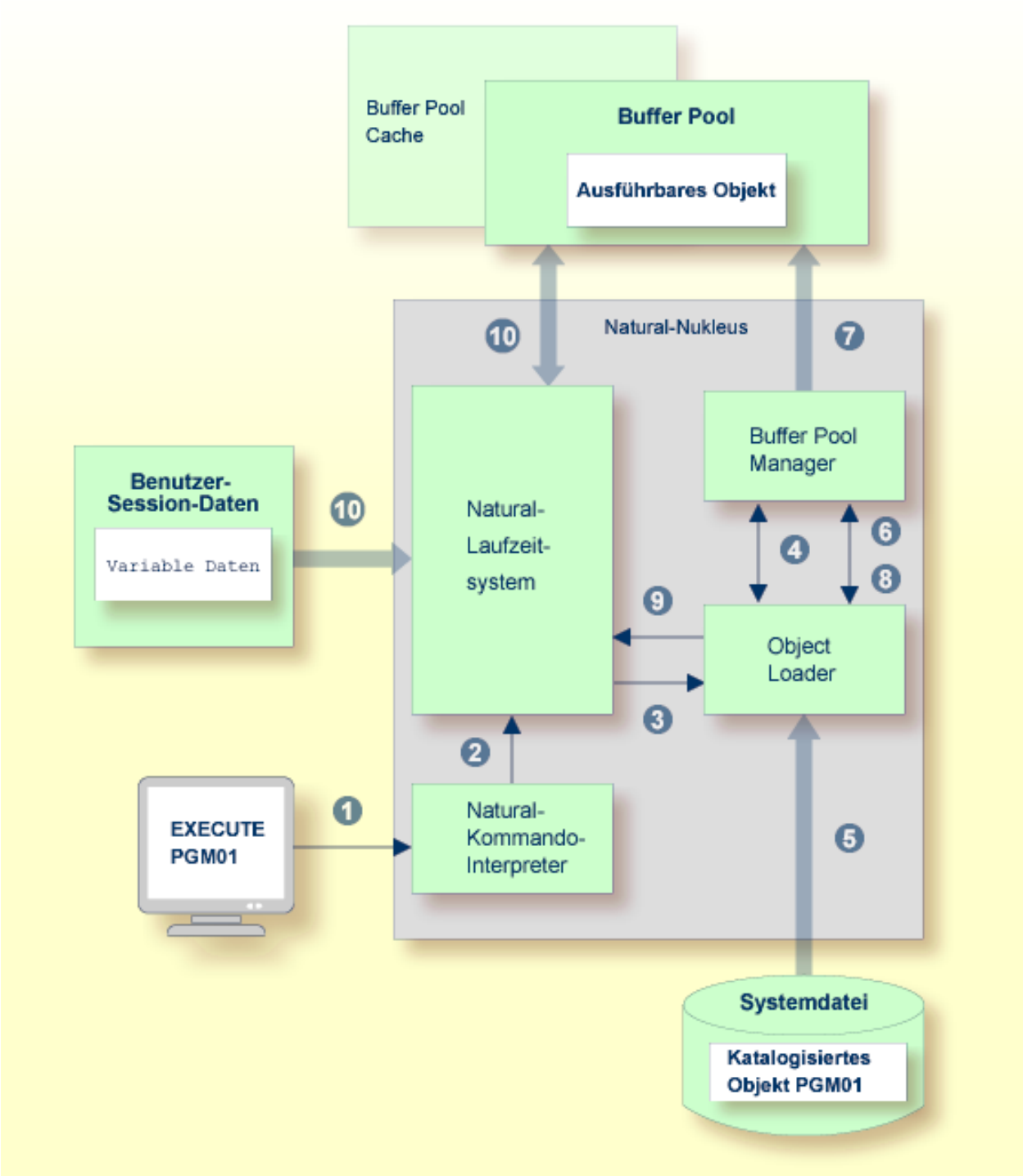
Objekt aus dem Buffer Pool entfernen

Objekte werden aus dem Buffer Pool entfernt, sobald sie nicht mehr von einem Benutzer referenziert werden und der Platz zum Laden neuer Objekte benötigt wird. Dadurch wird die maximale Ausnutzung des im Buffer Pool verfügbaren Speicherplatzes gewährleistet.

Falls ein Buffer Pool Cache vorhanden ist, werden die nicht benutzten Objekte vom Buffer Pool in den Cache-Speicher ausgelagert. Wenn ein nicht benutztes Objekt auch aus dem Cache entfernt wird oder wenn kein solcher Cache existiert, wird das betreffende Objekt erneut von der entsprechenden Natural-Systemdatei geladen, sobald es wieder von einem Benutzer referenziert wird.

Beispiel für das Laden und Ausführen eines Objekts

Das folgende Diagramm veranschaulicht den Vorgang des Ladens und Ausführens eines Natural-Programms:


Buffer Pool
Cache
Buffer Pool
Ausführbares Objekt
Natural-Nukleus
10
7
Benutzer-
Session-Daten
Variable Daten
10
Natural-
Laufzeit-
system
Buffer Pool
Manager
4
6
8
9
Object
Loader
3
2
EXECUTE
PGM01
1
Natural-
Kommando-
Interpreter
5
Systemdatei
Katalogisiertes
Objekt PGM01

Legende

1. Ein Benutzer setzt das Natural-Systemkommando `EXECUTE` ab und fordert die Ausführung eines Natural-Objekts des Typs Programm mit dem Namen `PGM01` an.
2. Der Natural-Kommando-Interpreter interpretiert das Kommando und gibt die Anforderung an das Natural-Laufzeitsystem weiter.
3. Das Natural-Laufzeitsystem übergibt die Anfrage an den Object Loader.
4. Der Object Loader weist den Buffer Pool Manager an, den Buffer Pool (und, falls vorhanden, den Buffer Pool Cache) nach dem Objekt `PGM01` zu durchsuchen und die Adresse festzustellen, unter der das Objekt im Buffer Pool bzw. im Cache gespeichert ist.

   Findet der Buffer Pool Manager die Buffer-Pool-Adresse von `PGM01`, wird die Adresse vom Buffer Pool an den Object Loader übergeben und die Ausführung wird mit Schritt 8 fortgesetzt.
   Findet der Buffer Pool Manager die Buffer-Pool-Adresse von PGM01 nicht, meldet der Buffer Pool Manager das negative Ergebnis der Suche an den Object Loader zurück und die Ausführung des Objekts wird mit Schritt 5 fortgesetzt.
5. Der Object Loader ruft das katalogisierte Objekt von `PGM01` aus der entsprechenden Systemdatei ab.
6. Der Object Loader übergibt das katalogisierte Objekt von `PGM01` an den Buffer Pool Manager.
7. Der Buffer Pool Manager lädt `PGM01` in den Buffer Pool.
8. Der Buffer Pool Manager gibt die Buffer-Pool-Adresse von `PGM01` an den Object Loader zurück.
9. Der Object Loader übergibt die Buffer-Pool-Adresse von `PGM01` an das Natural-Laufzeitsystem.
10. Das Natural-Laufzeitsystem interpretiert den kompilierten Code des katalogisierten Objekts `PGM01` und führt ihn aus.

Verwandte Themen

Die folgenden Abschnitte in der *System-Architektur*-Dokumentation nehmen Bezug auf den Natural Buffer Pool:

- ***Natural-Systemdateien***
- ***Natural-Laufzeitsystem*** - *Natural-Nukleus*
- ***Katalogisierte Objekte*** - *Natural-Compiler*, *Natural-Nukleus*

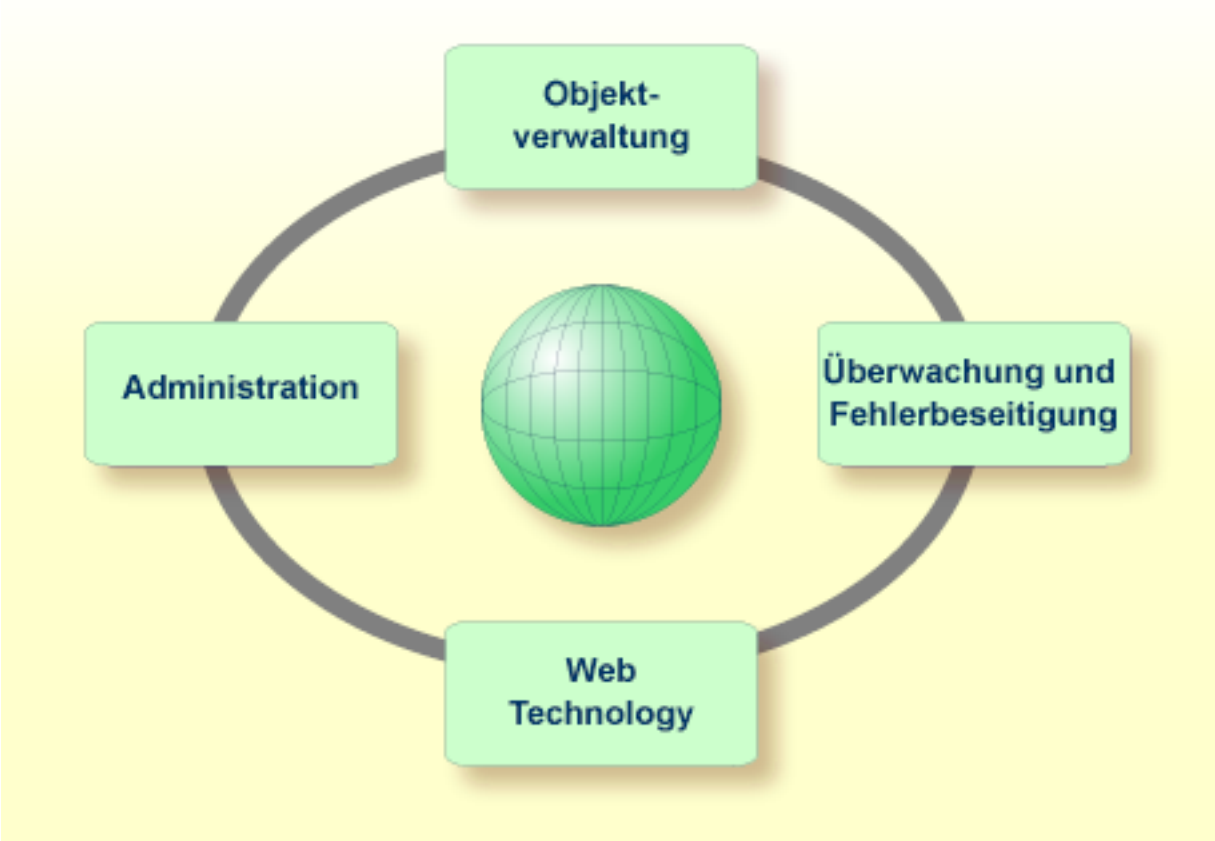
Die folgenden Abschnitte in anderen Natural-Dokumentationen betreffen den Natural Buffer Pool:

- *Natural Buffer Pool - Operations*-Dokumentation
- *Natural Storage Management - Operations*-Dokumentation

5 Natural-Editoren und Natural-Utilities

Natural-Utilities (Dienstprogramme) stellen Funktionen zur Verfügung, mit deren Hilfe Sie eine Entwicklungs-, Administrations- und Operations-Umgebung verwalten können.

Das folgende Diagramm zeigt eine Übersicht über die hauptsächlichen Verwendungszwecke der Natural-Editoren und Utilities:



Natural-Editoren bzw. Utilities stehen für folgende Aufgaben zur Verfügung:

- Erstellen und Pflegen von Natural-Objekten (Programme, Subprogramme usw.), **Natural-DDMs** (Datendefinitionsmodule) und Handhabung von Nicht-Natural-Objekten;
- Ausführung von administrativen Funktionen, z.B. Übertragen von Anwendungen von einer Hardware-Plattform auf eine andere, Steuern/Überwachen von Natural Remote Procedure Calls, Erzeugen von Reports und Pflegen von anwendungsspezifischen Fehlermeldungen;
- Überwachen von Anwendungen und Beseitigen von Fehlern (Debugging);
- Anwenden von Web-Technologie, z.B. Zugriffe auf einen Web Server oder Verarbeitung von XML-Dokumenten.

Verwandte Themen:

- ***Natural-DDMs*** *- DBMS-Schnittstelle - Datenbankenzugriff*
- *Editoren*-Dokumentation
- *Debugger und Dienstprogramme* bzw. *Utilities* (engl.)
- *Natural Web Interface - Web Technology*-Dokumentation
- *XML Toolkit - Web Technology*-Dokumentation

6 TP/OS-Schnittstelle

- Online-Verarbeitung 34
- Batch-Verarbeitung 36
- Natural-TP/OS-Schnittstellen 38

Natural bietet Schnittstellen, über die der Natural-Nukleus-Zugriff auf einen TP-Monitor für die Online-Transaktionsverarbeitung und auf das Betriebssystem (Operating System) für die Batch-Verarbeitung erhält.

Dieser Abschnitt behandelt folgende Themen:

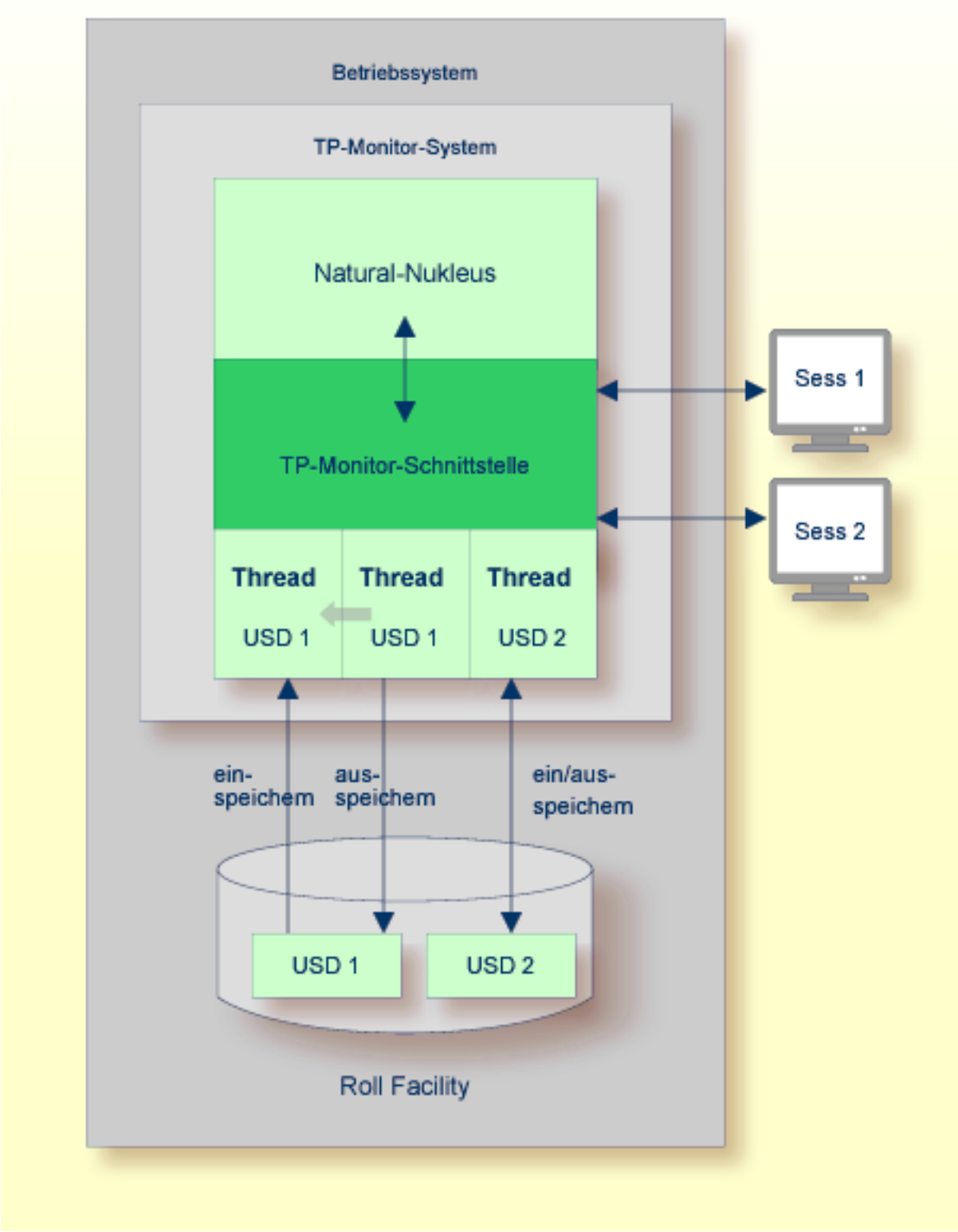
Online-Verarbeitung

In einer TP-Monitor-Umgebung läuft Natural als Standard-TP-Programm. Es arbeitet nach den Regeln, die für Programme gelten, die unter einem TP-Monitor ausgeführt werden.

In einer TP-Monitor-Umgebung bezeichnet man den Arbeitsbereich, in dem **Benutzer-Session-Daten** enthalten sind, als den Natural-Thread. Er wird für die Dauer der Benutzer-Session gepflegt.

Ein Natural-Thread kann, während eine Session auf eine Benutzer-Terminal-Eingabe wartet, in eine Roll Facility (Medium zum Ein-/Ausspeichern) ausgespeichert werden. Durch das Ausspeichern von Threads wird Speicherplatz für andere Benutzer-Sessions freigegeben. Wenn der Benutzer eine Eingabe am Terminal macht, z.B. wenn er EINGABE drückt, wird der Natural-Thread wieder eingespeichert. Ein Thread kann in einen anderen Speicherbereich verlagert werden, wie es im folgenden Diagramm am Beispiel der Benutzer-Session-Daten (`USD`) der Benutzer-Session `Sess 1` gezeigt wird.

Das folgende Diagramm veranschaulicht, wie ein TP-Monitor (hier: Com-plete) die Speicherzuordnung verwaltet, indem er Natural-Threads in eine Roll Facility ein- und ausspeichert.


Betriebssystem
TP-Monitor-System
Natural-Nukleus
TP-Monitor-Schnittstelle
Thread
USD 1
Thread
USD 1
Thread
USD 2
Sess 1
Sess 2
ein-
speichern
aus-
speichern
ein/aus-
speichern
USD 1
USD 2
Roll Facility

Batch-Verarbeitung

Im Batch-Betrieb verarbeitet Natural eine Session, die durch einen Batch Job gestartet wurde. Dabei ist keine Interaktion zwischen dem Rechner und dem Benutzer, der den Batch Job abgeschickt hat, nötig.

Ein Batch Job besteht aus Programmen, die nacheinander ausgeführt werden und dabei sequenziell Eingabedaten erhalten, die von einer Datei gelesen werden. Die Ausgabedaten werden auch in eine Datei geschrieben. Dabei ist der für einen Batch Job eingerichtete Arbeitsbereich in einer Batch-Region (unter z/OS) enthalten. Der Arbeitsbereich enthält Benutzer-Session-Daten, die für die Dauer der Batch-Benutzer-Session verwaltet werden.

Das folgende Diagramm zeigt den Aufbau einer Batch-Verarbeitungsumgebung:

Betriebssystem
Batch-Region/Partition/Task
Natural-Nukleus
Betriebssystem-Schnittstelle
Benutzer-Session-Daten

Natural-TP/OS-Schnittstellen

Informationen zu den bei Natural zur Verfügung stehenden TP-Monitor-Schnittstellen und zur Benutzung von Natural mit einem TP-Monitor finden Sie in den entsprechenden Abschnitten in der *TP-Monitor-Schnittstellen*-Dokumentation:

- *Verwendung von Natural mit TP-Monitoren*
- *Natural unter CICS*
- *Natural unter Com-plete/SMARTS*
- *Natural unter IMS TM*
- *Natural unter TSO*

Mit dem Natural-Dienstprogramm SYSTP können Sie TP-Monitor-spezifische Natural-Eigenschaften überwachen.

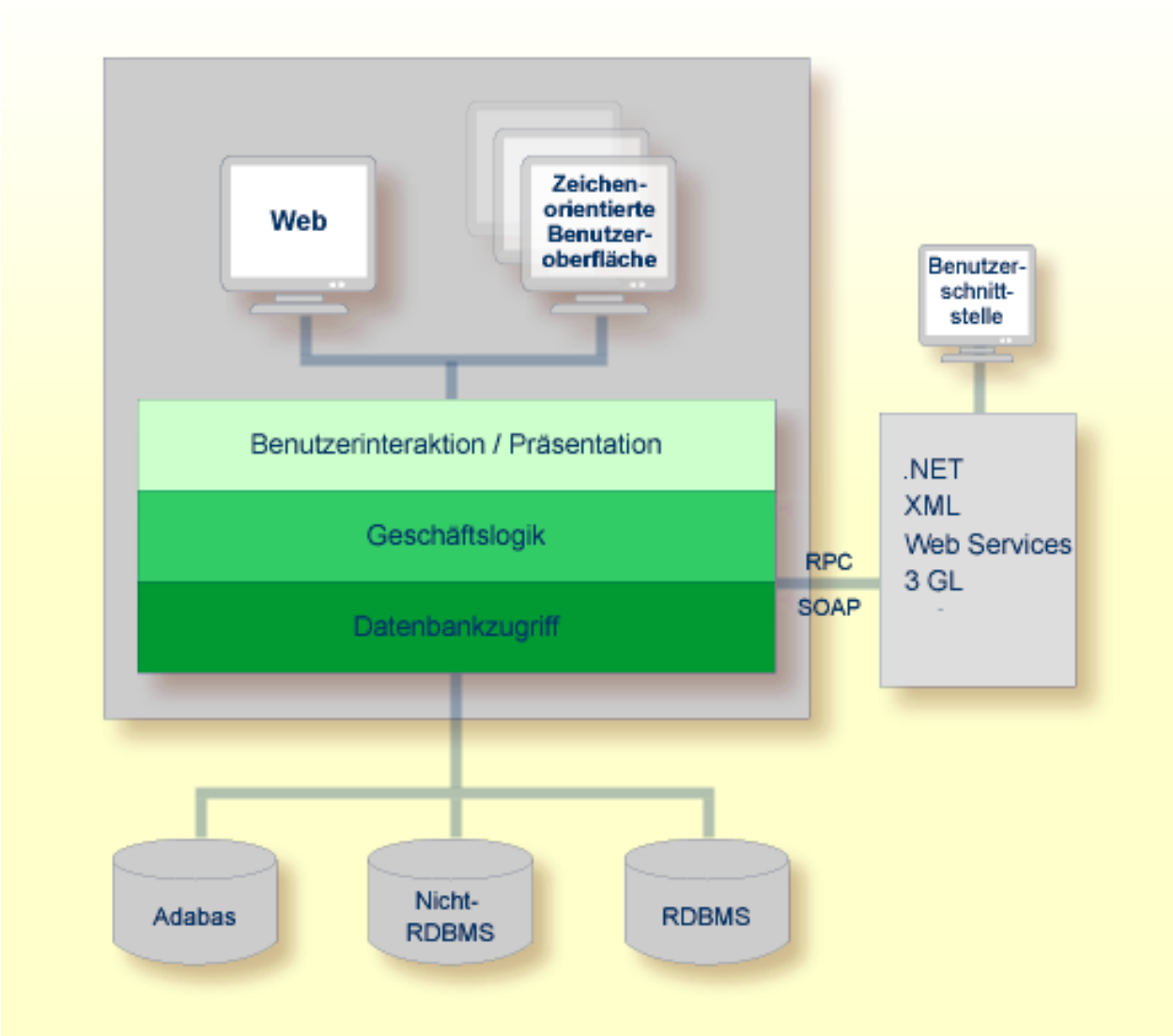
Informationen zu Natural im Batch-Betrieb finden Sie im Abschnitt *Natural im Batch-Modus* in der *Operations*-Dokumentation.

7 Benutzerschnittstelle

Natural unterstützt verteilte Anwendungen mit mehrschichtiger Architektur: Darstellungsschicht, fachliche Anwendungslogikschicht und Datenbankzugangsschicht. Eine Natural-Anwendung ist modular aufgebaut und bietet die Objektarten, die für jede Anwendungsschicht erforderlich sind.

Natural ermöglicht die Programmierung der verschiedenartigsten Benutzerschnittstellen: webbasierte Schnittstellen (z.B. HTML oder XML), prozessgesteuerte Anwendungen mit Zeichenoberflächen (CUIs) und ereignisgesteuerte Anwendungen mit grafischen Benutzeroberflächen (GUIs).

Außerdem kann Natural mit Benutzeroberflächen interagieren, die mit Nicht-Natural-Umgebungen, z.B. J2EE und .NET, verbunden sind. Verbindungen zwischen Natural- und Nicht-Natural-Umgebungen werden beispielsweise mittels Remote Procedure Call (RPC) oder einer SOAP-Anforderung hergestellt, über die es für Programme auf einem Client-Computer möglich ist, ein Natural-Objekt des Typs Subprogramm auf einem Natural-RPC-Server auszuführen.


Web
Zeichen-
orientierte
Benutzer-
oberfläche
Benutzer-
schnitt-
stelle
Benutzerinteraktion / Präsentation
Geschäftslogik
Datenbankzugriff
RPC
SOAP
.NET
XML
Web Services
3 GL
Adabas
Nicht-
RDBMS
RDBMS

8 Druckdateien und Arbeitsdateien

- Objekte mit Hilfe von Arbeitsdateien übertragen 42
- Druckdateien und Arbeitsdateien, Definition und Zugriff 43

Druckdateien (Print Files) und Arbeitsdateien (Work Files) sind „Datenbehälter“ für die dauerhafte oder zwischenzeitliche Speicherung von Daten, die in Natural-Objekten verarbeitet werden. Bei diesen „Datenbehältern“ handelt es sich beispielsweise um physische Dateien, Members von partitionierten Datasets oder Tape Datasets, die entweder vom Betriebssystem oder dem TP-Monitor zur Verfügung gestellt werden. Der Zugriff von Natural auf die Druckdateien und Arbeitsdateien erfolgt sequenziell.

Eine Druckdatei dient dazu, Report-Daten und Druckersteuerzeichen zu speichern, die für die Ausgabe eines Reports auf einem Drucker benötigt werden.

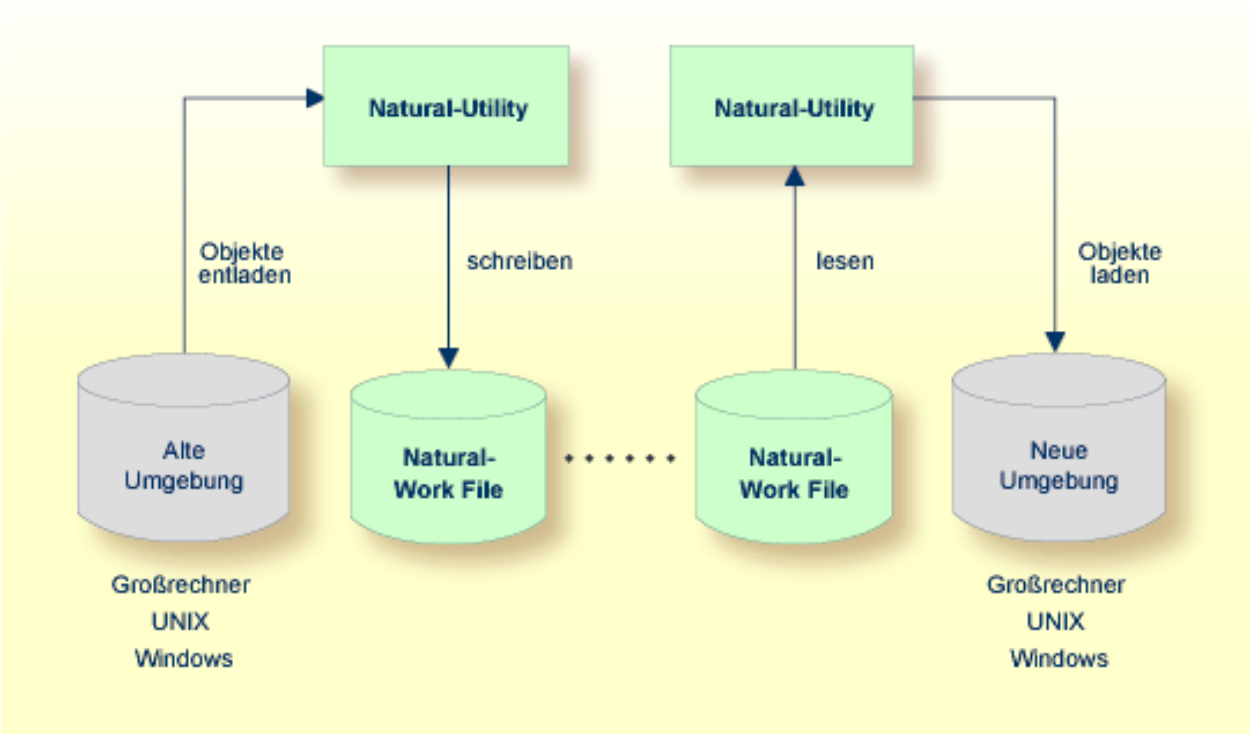
Eine Arbeitsdatei wird verwendet, um Daten zu speichern, die zwischen Natural-Objekten untereinander oder zwischen einem Natural-Objekt und einem in einer anderen Programmiersprache, z.B. COBOL, geschriebenen Objekt ausgetauscht werden.

Druckdateien und Arbeitsdateien können unter Berücksichtigung der Konventionen des installierten Betriebssystems bzw. TP-Monitors zur Batch- oder Online-Verarbeitung verwendet werden.

Der folgende Abschnitt behandelt folgende Themen:

Objekte mit Hilfe von Arbeitsdateien übertragen

Das folgende Diagramm veranschaulicht, wie die in einer Arbeitsdatei (Work File) enthaltenen Daten verwendet werden können, um Natural-Objekte zwischen verschiedenen Natural-Umgebungen auf unterschiedlichen Plattformen zu übertragen. Dies geschieht mit Hilfe eines Dienstprogramms, einer Natural-Utility (z.B. der Object Handler), mit dem die in der Ursprungsumgebung vorhandenen Objekte in ein Work File entladen und von dort in die Zielumgebung geladen werden. Falls erforderlich, wird ein Anwendungsprotokoll, z.B. FTP, benutzt, um Work Files von Ursprungs- in Zielumgebungen zu übertragen.



Druckdateien und Arbeitsdateien, Definition und Zugriff

Druckdateien und Arbeitsdateien werden logisch für eine Natural-Umgebung definiert und können durch entsprechende Angaben in einem Natural-Parameter und/oder in Control-Statements des zugrundeliegenden Betriebs- oder TP-Monitorsystems einer Datei bzw. einem Drucker physisch zugeordnet werden. Die Zuordnungen können für die aktuelle Session zur Laufzeit geändert werden.

Die Daten werden unter Verwendung der entsprechenden Natural-Statements in eine Arbeitsdatei geschrieben und von dort gelesen bzw. in eine Druckdatei geschrieben.

9 Natural-Systemdateien

- Systemdatei-Arten 46
- Libraries in Systemdateien 47

In einer Natural-Systemdatei werden Natural-Objekte und Nicht-Natural-Objekte gespeichert, die entweder zu Benutzeranwendungen oder zu von uns ausgelieferten Natural-Systemanwendungen (z.B. Utilities) gehören.

Zu den in einer Systemdatei gespeicherten Natural-Objekten gehören **katalogisierte Objekte** und **Source-Objekte**, siehe Beschreibung des Natural-Compilers, Abschnitt **Natural-Nukleus**. Natural-Systemdateien werden in Datenbankdateien oder Speichersystemen wie z.B. Adabas und VSAM gespeichert. Je nach Systemumgebung können die Natural-Systemdateien in unterschiedlichen Datenbankdateien oder Speichersystemen untergebracht sein.

Der Zugriff auf eine Natural-Systemdatei erfolgt, wenn ein Benutzer ein Natural-Objekt liest oder ändert. Außerdem wird auf eine Natural-Systemdatei zugegriffen, wenn ein Objekt zur anschließenden Ausführung in den Buffer Pool geladen wird (siehe Abschnitt ***Natural Buffer Pool***).

Dieser Abschnitt behandelt folgende Themen:

Systemdatei-Arten

In der folgenden Tabelle sind diejenigen Natural-Systemdateien aufgeführt, die normalerweise in einer Natural-Umgebung vorhanden sind. Die Verfügbarkeit der Systemdateien und deren Inhalte hängen von den Software AG-Produkten ab, die zusätzlich zu Basis-Natural installiert sind.

Systemdatei	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). Wenn Predict installiert ist, enthält FDIC außerdem noch Daten für das Predict-Datendiktionärsystem. 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.

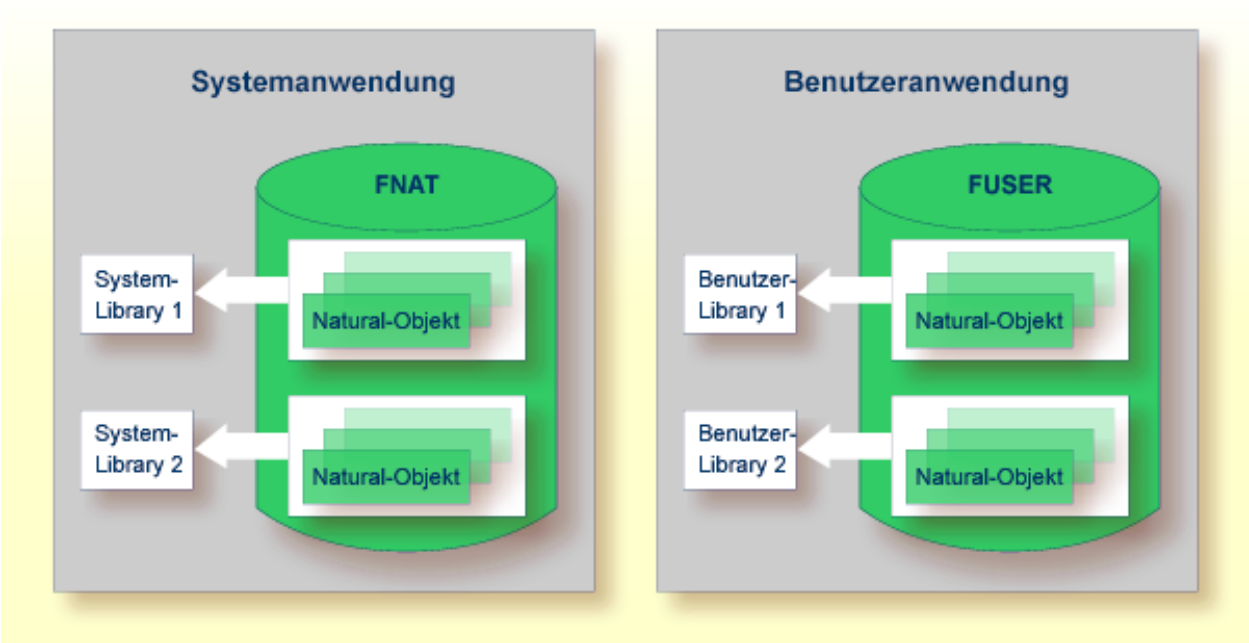
Systemdatei	Verfügbar mit	Datei-Inhalt
FPROF	Basis-Natural	Parameterprofile, die mit dem Profilparameter `PROFILE` angegeben werden, vorausgesetzt dass keine Datenbankinformationen als Subparameter von `PROFILE` geliefert werden.
FREG	Basis-Natural	Registry-Daten, die nicht explizit in einer anderen Systemdatei gespeichert werden.

Verwandtes Thema:

Eine Übersicht über die Profilparameter, die für alle Systemdateien gelten bzw. mit denen die Standardwerte für einzelne Systemdateien verändert werden können, befindet sich im Abschnitt *Natural-Systemdateien* in der *Operations*-Dokumentation.

Libraries in Systemdateien

Die in den Systemdateien `FNAT` und `FUSER` enthaltenen Natural-Objekte sind in logische Konstrukte unterteilt, die als Natural Libraries (Bibliotheken) bezeichnet werden; siehe folgendes Diagramm:



Verwandte Themen:

- ***Katalogisiertes Objekt*** und ***Source-Objekt*** - *Natural-Compiler*, *Natural-Nukleus*
- *Natural Security*-Dokumentation
- *Natural Advanced Facilities*-Dokumentation

- ***Natural-DDMs*** *- DBMS Interface - Database Access*
- *Predict*-Dokumentation
- *Natural Development Server*-Dokumentation

10 DBMS-Schnittstelle - Datenbankzugriff

- Von Natural unterstützte Datenbankverwaltungssysteme 50
- Natural-Datenbankabfragesprache 51
- Spezielle SQL-Statements 52
- Natural-DDMs 52

Natural bietet Schnittstellen zu verschiedenen Datenbankverwaltungssystemen (DBMS), über die der Zugang zu Daten möglich ist, die z.B. in einer Adabas-Datenbank, einem relationalen Datenbankverwaltungssystem (RDBMS) und/oder in einem Nicht-RDBMS gespeichert sind.

Natural-Anwendungen können gleichzeitig auf Benutzerdaten in mehreren DBMS zugreifen. Im Gegensatz zum Zugriff auf **Druckdateien und Arbeitsdateien** (siehe entsprechenden Abschnitt), der sequenziell erfolgt, kann der Zugriff auf die in einem DBMS gespeicherten Daten wahllos erfolgen.

Zu den von Natural unterstützten DBMS gehören Db2 und VSAM.

Dieser Abschnitt enthält Informationen zu den von Natural unterstützten DBMS und beschreibt, wie prinzipiell aus einer Natural-Anwendung heraus auf Daten in einer Datenbank zugegriffen wird.

Von Natural unterstützte Datenbankverwaltungssysteme

Natural bietet Schnittstellen zu den folgenden Datenbankverwaltungssystemen (DBMS):

- **Adabas**
- **Db2**
- **VSAM**

Adabas

Die Natural Adabas-Schnittstelle ermöglicht den Zugriff auf Daten, die in einer Adabas-Datenbank gespeichert sind.

Adabas unterstützt sowohl relationale als auch geschachtelte Datenstrukturen.

Die Natural Adabas-Schnittstelle ist ein integrierter Bestandteil von Natural. Sie gestattet den Zugriff auf Adabas-Datenbanken auf lokalen Rechnern. Für den Remote-Zugang wird zusätzliche Routing- und Kommunikationssoftware benötigt, z.B. Entire Net-Work. Der Typ der Host-Maschine, auf der die Adabas-Datenbank läuft, ist in jedem Fall für Natural immer transparent.

Verwandte Themen:

- *Daten in einer Adabas-Datenbank aufrufen - Leitfaden zur Programmierung*
- *Adabas*-Dokumentation

Db2

Die Natural Db2-Schnittstelle ermöglicht den Zugriff auf Daten, die in einer Db2 UDB von IBM für z/OS-Betriebssysteme gespeichert sind.

Verwandtes Thema:

Natural for Db2-Dokumentation

VSAM

Die Natural VSAM-Schnittstelle ermöglicht den Zugriff auf Daten, die in VSAM-Datasets gespeichert sind.

VSAM ist ein IBM-Zugriffsverfahren, mit dem Datensätze verschiedener Dataset-Organisationsformen verwaltet werden können: Key-Sequenced Data Sets, Entry-Sequenced Data Sets oder Relative-Record Data Sets.

Verwandtes Thema:

Natural for VSAM-Dokumentation

Natural-Datenbankabfragesprache

Die Natural-Datenbankabfragesprache (Data Manipulation Language - DML) bietet eine gemeinsame Datenzugriffssyntax für alle von Natural unterstützten Datenbankverwaltungssysteme (DBMS). Durch die Natural-DML können Natural-Objekte unter Verwendung derselben Sprach-Statements (DML-Statements), z.B. `FIND`, `READ`, `STORE` und `DELETE`, auf verschiedenartige DBMS zugreifen.

Natural bestimmt die DBMS anhand seiner Konfigurationsdateien und übersetzt die DML-Statements in datenbankspezifische Befehle; d.h., Natural erzeugt Direktkommandos für Adabas-Datenbanken, SQL-Statement-Strings und Host-Variablen-Strukturen für RDBMS, die SQL verwenden.

Ein Natural-Objekt, in dem ein DML-Statement verwendet wird, das *keine* datenbankspezifische Klausel enthält, kann gegen unterschiedliche DBMS ausgeführt werden. Ein DML-Statement, das eine datenbankspezifische Klausel enthält, muss geändert warden, bevor es auf einem andersartigen DBMS verwendet werden kann. In der *Statements*-Dokumentation sind datenbankspezifische Hinweise enthalten.

Verwandtes Thema:

Statements-Dokumentation

Spezielle SQL-Statements

Zusätzlich zu den DML-Statements bietet Natural einen Satz spezieller SQL-Statements zur ausschließlichen Verwendung in Verbindung mit RDBMS. Dies sind die Statements `SELECT`, `INSERT`, `UPDATE`, `DELETE`, `COMMIT` und `ROLLBACK`, die in der *Statements*-Dokumentation beschrieben sind.

Der Satz spezieller SQL-Statements wird vervollständigt durch Flexible SQL und Funktionen zum Arbeiten mit Stored Procedures. Flexible SQL gestattet die Benutzung frei wählbarer Syntax.

Verwandte Themen:

- *SQL Statements - Statements*-Dokumentation
- *Flexible SQL - Statements*-Dokumentation

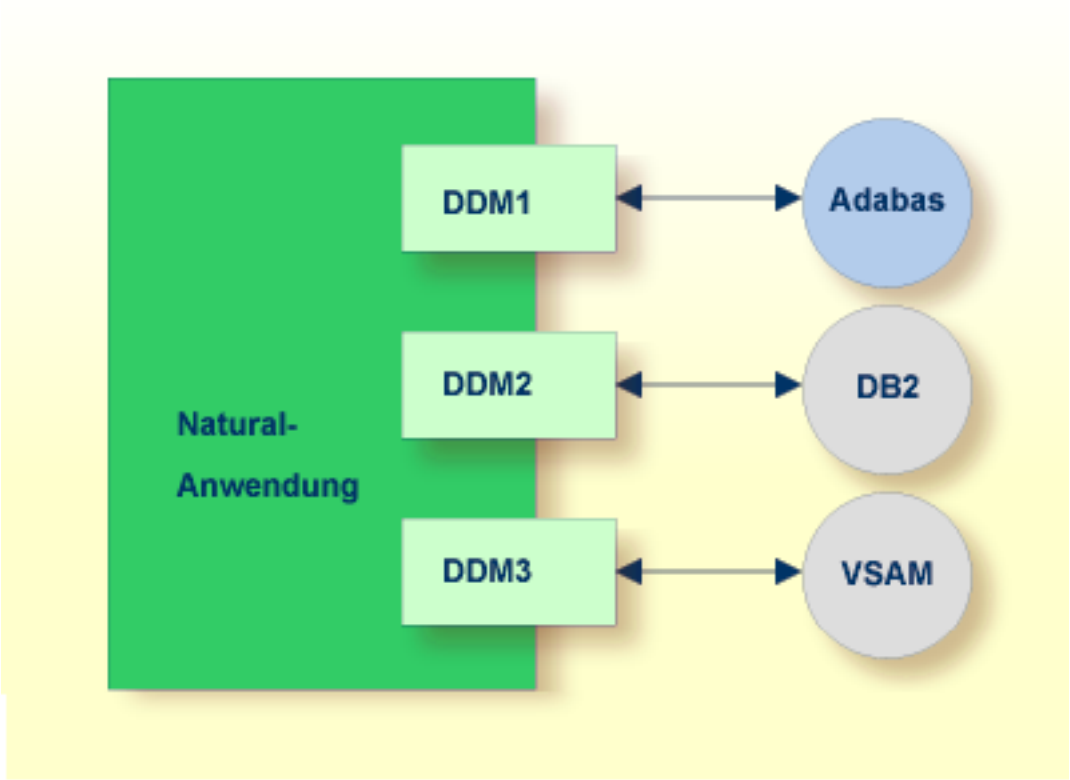
Natural-DDMs

Natural bietet ein Objekt, das als DDM (Datendefinitionsmodul) bezeichnet wird und den komfortablen und transparenten Zugriff auf Datenbankdateien auf unterschiedlichen DBMS ermöglicht.

Ein DDM ist eine logische Sicht auf eine physische Datenbankdatei. Das DDM enthält Informationen zu den einzelnen Feldern der Datei, die für die Verwendung dieser Felder in einem Natural-Objekt relevant sind.

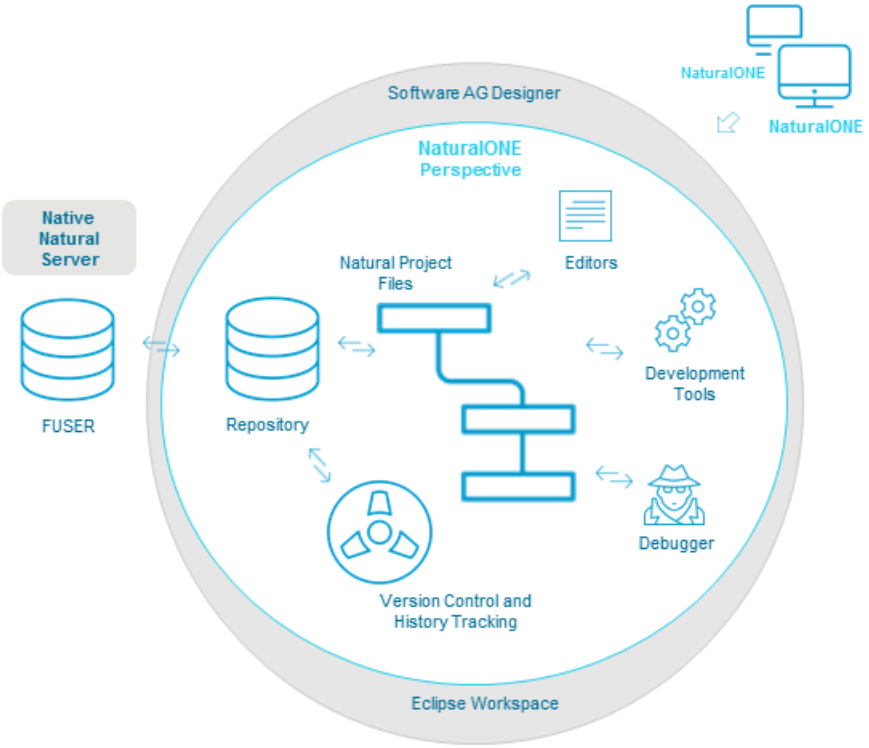
Ein DDM stellt die Verbindung zwischen den Natural-Datenstrukturen und den Datenstrukturen in dem zu verwendenden DBMS her. Bei einer solchen Struktur kann es sich um eine Tabelle in einem RDBMS, eine Datei in einer Adabas-Datenbank oder einen VSAM Dataset handeln. Das DDM verbirgt somit die wirkliche Struktur der Datenbank, auf die zugegriffen wird, gegenüber der Natural-Anwendung.

Das folgende Diagramm veranschaulicht, wie Natural aus einer einzelnen Anwendung heraus auf mehrere, unterschiedliche Datenbanken zugreifen kann, indem es Referenzen auf DDMs verwendet, die die spezifischen Datenstrukturen im jeweiligen DBMS darstellen:


Natural-
Anwendung
DDM1
DDM2
DDM3
Adabas
DB2
VSAM

II NaturalONE-Architektur

NaturalONE ist eine Eclipse-basierte Umgebung für die Entwicklung und Pflege von Natural-Anwendungen mit web-basierten Benutzerschnittstellen und Natural-Diensten (Natural Services). NaturalONE ist integriert in eine Eclipse Workbench und kombiniert Eclipse-Standardfunktionalität mit Natural-spezifischen Funktionen und Tools zu einem einzigen Entwicklungs-Framework.



In NaturalONE werden Natural-Anwendungen in Projekten organisiert. Alle von einer Anwendung referenzierten Natural-Objekte und Natural-Bibliotheken (Libraries) sind in Projektdateien enthalten. Die Projekte sind in einem Arbeitsbereich (Eclipse Workspace) enthalten, der als Schnittstelle zwischen den verschiedenen Tools fungiert, die zum Entwickeln von Anwendungen zur Verfügung stehen.

Natural-Quellcode-Objekte aus der Systemdatei FUSER in einer nativen Natural-Großrechner-, UNIX-, Linux- oder Windows-Umgebung werden in einem Repository gespeichert und verwaltet. Quellcode wird mit lokalen **NaturalONE-Editoren** geändert und automatisch zur Systemdatei FUSER in der Zielumgebung übermittelt. Die Systemdatei FUSER enthält nur katalogisierte Objekte (generierte Programme, GPs), alle Quellcode-Objekte verbleiben im Repository.

Das Repository wird von mehreren Entwicklern benutzt. Änderungen am Quellcode werden in einer Änderungshistorie festgehalten und Anwendungen werden versioniert. Das gestattet parallele Operationen, das Vergleichen und Zusammenführen von Code sowie die Rückkehr zu früheren Versionen einer Anwendung.

NaturalONE stellt verschiedene Entwicklungs-Tools zur Verfügung, die die Entwickler beim Kodieren, bei der Fehlerbehebung (Debugging), bei der Datenanalyse (Profiling), beim Testen und bei der Versionsverwaltung von Anwendungen unterstützen.

11 Was ist NaturalONE?

- Allgemeines zu NaturalONE 58
- Grundlegende Funktionen für die Entwicklung von Natural-Anwendungen 59
- Natural für Ajax / Ajax-Developer 62
- Optionale Komponenten 62

Anmerkung: Die folgende Einführung ist ein übersetztes Exzerpt aus der *NaturalONE*-Dokumentation (Stand: Mai 2022).

Die folgenden Themen werden behandelt:

Allgemeines zu NaturalONE

NaturalONE ist Teil des Software AG Designers. Es handelt sich um eine Eclipse-basierte Entwicklungsumgebung für die Entwicklung und Pflege von Natural-Anwendungen mit webbasierten Benutzeroberflächen und Natural-Services. NaturalONE vereint die Funktionalität mehrerer Werkzeuge in einem einzigen Entwicklungs-Framework und deckt den gesamten Lebenszyklus der Produktentwicklung ab, einschließlich Anwendungsentwicklung, Testen, automatisierter Dokumentation der Quellen in Predict, Versionierung und Bereitstellung der Anwendung in der Produktionsumgebung.

NaturalONE nutzt die Standardfunktionen von Eclipse und erweitert die Eclipse-Workbench um eigene Perspektiven, Ansichten, Editoren und Kommandos. NaturalONE ist nahtlos in die Eclipse-Workbench integriert. Das bedeutet, dass Sie nicht sehen, wo die Standard-Eclipse-Workbench endet und wo die NaturalONE-spezifische Benutzeroberfläche beginnt.

NaturalONE richtet sich an Entwickler, die es gewohnt sind, nativ auf Mainframe-, UNIX-, Linux- oder Windows-Plattformen zu arbeiten. Dank der grafischen Benutzeroberfläche und vieler Funktionen für die schnelle Natural-Quellcode-Entwicklung (z. B. Code Assist) gewinnen Entwickler mit dieser Umgebung an Produktivität. Entwickler, die bereits mit Eclipse vertraut sind, können Natural-Quellcode bequem in einer Umgebung bearbeiten, an die sie bereits gewöhnt sind; die Umgebung verhält sich immer gleich, egal ob Natural- oder Java-Anwendungen entwickelt werden.

Wenn Sie mit Eclipse noch nicht vertraut sind, sehen Sie sich die Eclipse-Online-Hilfe unter ***http://www.eclipse.org/documentation/*** an, oder starten Sie den Software AG Designer und wählen Sie dann **Help > Help Contents**. Allgemeine Informationen zu Eclipse finden Sie dann unter Workbench User Guide. Bei der Arbeit mit dem Software AG Designer bietet die Online-Hilfe auch Hilfe zu den aktuell installierten Software AG-Produkten; diese finden Sie unter Software AG Designer Guides.

Anmerkung: Mit Natural Version 9 für Großrechner, Unix und Linux sind NaturalONE und der Natural Development Server in Natural integriert. Die Natural-Editoren (Programm-Editor, Datenbereich-Editor und Masken-Editor) sind deaktiviert. Da NaturalONE- und Natural Development Server-Lizenzen in Natural integriert sind, können sie mit dem Natural (NAT)-Lizenzschlüssel aktiviert werden. Dies gilt nicht für Natural for Windows.

Grundlegende Funktionen für die Entwicklung von Natural-Anwendungen

Die Grundfunktionalität von NaturalONE wird in *Using NaturalONE* in der *NaturalONE*-Dokumentation beschrieben.

Im Eclipse-Arbeitsbereich sind alle Dateien in Projekten organisiert. Eine direkte Verbindung zu einem Natural-Server ist für die Bearbeitung von Natural-Quellcode nicht erforderlich. Um eine Natural-Anwendung auszuführen und zu debuggen, stellt NaturalONE jedoch eine Verbindung zur entsprechenden Natural-Laufzeitumgebung her.

Die folgenden Themen werden behandelt:

- Leistungsstarkes Tool Set
- Leistungsstarke Natural-Umgebung
- Direkte Entwicklung auf einem Natural Server

Leistungsstarkes Tool Set

Anwendungen können in der Eclipse-Umgebung entwickelt und gepflegt werden.

Editoren

Zur Unterstützung der Entwickler und zur Steigerung der Produktivität während des Entwicklungszyklus stehen spezielle Natural-Editoren zur Verfügung:

- **Quelltext-Editor**
  Der NaturalONE **Source Editor** verwendet einen echten inkrementellen Natural-Parser. Er unterstützt den Entwickler mit Syntaxfärbung des Natural-Quelltextes und Content Assist für schnelles Schreiben von Code. Beide Natural-Programmiermodi (Reporting Mode und Structured Mode) werden unterstützt. Für die Entwicklung internationaler Anwendungen ist die bidirektionale Sprachunterstützung aktiviert.

  NaturalONE nutzt den Source Editor für die Bearbeitung von Datenbereichen (Natural Data Areas). Datenbereiche werden mit dem Statement `DEFINE DATA` definiert. Mit Hilfe des Natural-Parsers lassen sich schnell und einfach syntaktisch korrekte Datenbereiche entwickeln.
- **Masken-Editor**
  Mit dem NaturalONE **Map Editor** können Natural-Masken (Natural-Objekttyp `Map`) grafisch definiert werden. Dazu gehört auch eine bidirektionale Sprachunterstützung. Mit der Gliederungsansicht (**Outline**-Ansicht) können Sie auf grafischer Basis durch die Teile einer Maske navigieren. Es können auch Inline-Regeln oder Predict-Regeln bearbeitet werden.

- **DDM-Editor**
  Datendefinitionsmodule (DDMs), die von Predict generiert wurden, können für Ihre Anwendung übernommen werden. Mit dem NaturalONE **DDM Editor** ist es auch möglich, DDMs von Grund auf neu zu erstellen. Dies ist für alle von Natural unterstützten Typen möglich: Adabas, SQL, Tamino, VSAM und andere. In der Gliederungsansicht (**Outline**-Ansicht) wird die DDM-Struktur in hierarchischer Form visualisiert.

Debuggen und Ausführen

Das Debuggen und Ausführen von Anwendungen ist möglich. Watchpoints, Breakpoints usw. machen die Fehlerbehebung weniger komplex.

XML

Mit dem **XML Toolkit** ist es möglich, Funktionalität für die Verarbeitung von XML-Dokumenten zu generieren. DTDs oder Schemas können zur Generierung von Natural Data Areas, Parser-Implementierungen und Serializern für XML-Dokumente verwendet werden und umgekehrt.

Datenbankabfragen

Der **Data Browser** ermöglicht den schnellen Zugriff auf Adabas- oder SQL-Datenbanken. Mit nur wenigen Mausklicks ist es möglich, eine Datenbankabfrage zu schreiben und die abgerufenen Daten in der Reportdaten-Ansicht (**Report Data**) anzuzeigen. Es ist einfach, den Inhalt der Datenbank zu überprüfen und zu testen, ob die Anwendung korrekt funktioniert.

Leistungsstarke Natural-Umgebung

Das Auslagern der Anwendung in den Eclipse-Arbeitsbereich bietet Ihnen mehrere Vorteile. Der **Natural Builder** behält die Abhängigkeiten der Anwendung bei. Diese werden in der Abhängigkeiten-Ansicht (**Dependencies**) visualisiert. Der Builder behält den Überblick über die Änderungen am Quellcode. Anhand der Label-Dekorationen können Sie leicht erkennen, welche Natural-Objekte noch nicht lokal gespeichert, noch nicht auf dem Natural-Server kompiliert oder noch nicht in Ihrem Versionskontrollsystem versioniert wurden. Wenn die Einstellungen richtig gesetzt sind, rekatalogisiert der Builder die entsprechenden Objekte auf dem Natural-Server anhand der Natural-Parameter, die in der Eclipse-Umgebung definiert sind.

Grundsätzlich unterstützt der Builder zwei verschiedene Workspace-Strukturen: eine, die sehr Natural-bezogen ist (d. h. die Anwendung basiert auf der herkömmlichen Natural-Library-Struktur),und eine andere, die es Ihnen ermöglicht, Ordner zu definieren, die den Namenskonventionen des zugrunde liegenden Dateisystems (Windows oder Linux) folgen. Bei der letztgenannten Workspace-Struktur werden die Ordner auf „reale“ Natural-Libraries abgebildet. So können Sie Ihre Natural-Anwendungen im Arbeitsbereich logischer strukturieren als mit der herkömmlichen Library-Struktur. Ordner können verschachtelt werden.

Neben der Verwendung von Ordnern im Arbeitsbereich ist es auch möglich, alternative (lange) Dateinamen für die Objekte zu verwenden. Auch diese Dateinamen müssen den Regeln des zugrunde liegenden Dateisystems folgen. Es gibt ein Mapping zwischen dem alternativen Dateinamen und dem Natural-Objektnamen, wobei der Natural-Objektname immer den Natural-Namenskonventionen folgt.

Wo es sinnvoll ist, werden Assistenten (z. B. zum Anlegen neuer Projekte oder neuer Objekte) verwendet.

Fehlermeldungen werden mit dem NaturalONE **Error Message Editor** bearbeitet. Sie werden auch im Eclipse-Arbeitsbereich gespeichert.

Um die Produktivität zu steigern, stehen Funktionen zur Verfügung, die häufig wiederholte Aktionen unterstützen. So können Sie beispielsweise Codefragmente in separate Natural-Objekte auslagern. Oder Sie können Natural-Objekte mithilfe der Refactoring-Funktion umbenennen. Wenn Sie einen Natural-Quelltext von Grund auf neu erstellen, wird automatisch ein dem Natural-Objekttyp entsprechendes Skelett generiert.

Sie können die NaturalONE-Umgebung an Ihre speziellen Anforderungen anpassen. In den Natural-Voreinstellungen können Sie das Verhalten des Natural Builders ändern, die Plattform festlegen, für die der Natural-Parser die Natural-Syntax prüfen soll, oder eigene Code-Vorlagen erstellen.

Sie können Ihre Anwendungen in einem Versionskontrollsystem versionieren. Eine Natural-Anwendung besteht aus den Natural-Objekten, Konfigurationsparametern, Fehlermeldungen usw. Mit einem Versionskontrollsystem können Sie die Anwendungsteile von mehreren Entwicklern zusammenführen. Nach der Zusammenführung der Quellen im Repository des Versionskontroll-systems ist die Anwendung bereit für die Bereitstellung auf der gewünschten Plattform. Der Bereitstellungsassistent sorgt dafür, dass die entsprechenden Teile der Anwendung auf dem Natural-Server bereitgestellt werden. Siehe auch *Using a Version Control System* in der *NaturalONE*-Dokumentation.

Es ist möglich, Natural-Anwendungen oder Teile davon direkt auszuführen, zu debuggen oder zu testen. Die lokale Natural-Laufzeitumgebung ist bereits für die Verwendung mit NaturalONE konfiguriert. Standardmäßig ist keine weitere Konfiguration erforderlich. Mit der lokalen Natural-Laufzeit steht eine EntireX-Broker-Umgebung zur Verfügung und ein RPC-Server wird gestartet. So können Sie RPC-basierte Anwendungen entwickeln oder Webservices nutzen

Direkte Entwicklung auf einem Natural Server

Wenn Sie mit Natural Studio und dem Single Point of Development (SPoD)-Konzept von Natural vertraut sind, finden Sie in NaturalONE vergleichbare Funktionen. Die **Natural Server**-Ansicht macht die Informationen von Natural-Servern sichtbar, die sich auf verschiedenen Plattformen befinden können (auf einem Großrechner, UNIX, Linux oder Windows). In dieser Ansicht können Sie Ihren Natural-Quellcode direkt auf einem Natural-Server bearbeiten und katalogisieren oder sogar direkt auf dem Server ausführen. Sie erhalten Informationen über die Serverkonfiguration, ähnlich wie die Ausgabe der Natural-Systemkommandos `SYSPROD`, `SYSPROF`, `SYSFILE` und `UNLOCK`.

Natural für Ajax / Ajax-Developer

Neben den grundlegenden Funktionen für die Entwicklung von Natural-Anwendungen können Sie mit NaturalONE auch Rich-Internet-Anwendungen erstellen, die die Ajax-Technologie (Asynchronous JavaScript und XML) nutzen. Dieses Funktionalität, der **Ajax Developer**, wird immer zusammen mit der Basisfunktionalität installiert und bietet somit eine integrierte Entwicklungs- und Laufzeitumgebung für Natural for Ajax-Anwendungen.

Der Ajax Developer enthält eine Reihe von Werkzeugen zur Erstellung und Pflege komplexer grafischer Benutzeroberflächen. Zentrales Werkzeug ist der **Layout Painter**, mit dem Sie Layouts für HTML-Seiten definieren können.

Ausführliche Informationen finden Sie unter *Natural for Ajax* und *Ajax Developer*.

Optionale Komponenten

Zusätzlich zu den oben genannten Funktionen für die Anwendungsentwicklung, die immer mit NaturalONE installiert werden, können Sie auch optionale Komponenten für NaturalONE installieren. Für einige dieser optionalen Komponenten muss zusätzliche Software auf einem Server installiert werden.

Die folgenden Überschriften entsprechen den Namen, die im Produktauswahlbaum des Software AG Installer verwendet werden. Siehe auch *Installing NaturalONE*.

- Application Testing
- Mainframe-Tools
- Natural Construct
- Predict

- Service Development

Application Testing

Mit der Komponente **Application Testing** können Sie direkt verschiedene Natural-Objekte wie Subprogramme, Maps, Subroutinen und Business Services testen oder Unit-Tests erstellen. Voraussetzung ist, dass EntireX installiert ist.

Wenn Sie Application Testing im Installer auswählen, wird das Kontextmenü **Testing** in der **Project Explorer**-Ansicht verfügbar. Detaillierte Informationen finden Sie in der *Application Testing*-Dokumentation.

Mainframe-Tools

Die Komponente **Mainframe Tools** umfasst die **Mainframe-Navigation**, mit der Sie von Eclipse aus auf Objekte zugreifen und diese bearbeiten können, die auf einem Großrechner gespeichert sind. Zu diesen Objekten gehören Datasets und Members sowie Systemobjekte wie aktive Jobs oder die Konsole unter dem Betriebssystem z/OS. Mit Mainframe Navigation werden die Objekte in einer Baumstruktur angezeigt und können in Eclipse durchsucht und bearbeitet werden. Auf dem Großrechner-Server wird Mainframe Navigation von Natural ISPF unterstützt.

Mainframe Navigation erfordert die Installation zusätzlicher Software auf einem Server. Siehe *Installation and Configuration* in der *Mainframe Navigation*-Dokumentation.

Wenn Sie **Mainframe Tools** im Installationsprogramm auswählen, wird die **Mainframe Navigation**-Ansicht verfügbar. Ausführliche Informationen finden Sie in der *Mainframe Navigation*-Dokumentation.

Natural Construct

Mit der **Natural Construct**-Komponente können Sie Ihre bestehenden Natural Construct-Modelle in NaturalONE verwenden und neue Natural Construct-Modelle oder Codeframes erstellen. Dazu muss zusätzliche Software auf einem Server installiert werden. Siehe *Requirements* im Abschnitt *Using Natural Construct* in der *Code Generation*-Dokumentation.

Wenn Sie Natural Construct im Installer auswählen, ist das Kontextmenü **Code Generation > New Using Construct Model** in der **Project Explorer**-Ansicht verfügbar. Ausführliche Informationen finden Sie unter *Using Natural Construct* in der *Code Generation*-Dokumentation.

Predict

Die Komponente **Predict** wird verwendet, um Natural-Quellcode in Predict über NaturalONE zu dokumentieren, von Predict auf dem Server gespeicherte Daten zu bearbeiten und abzurufen und externe Objekte zu verwalten (Generierung und Verwaltung). Sie erfordert die Installation zusätzlicher Software auf einem Server. Siehe *Setting Up a Predict Environment* in der *Predict Description and Generation*-Dokumentation .

Wenn Sie Predict im Installer auswählen, wird das Kontextmenü für Predict-Beschreibung und -Generierung in der Ansicht Natural Server verfügbar. Ausführliche Informationen finden Sie in der *Predict Description and Generation*-Dokumentation.

Service Development

Wenn Sie im Installer **Service Development** wählen, sind die folgenden Kontextmenüs in der **Project Explorer**-Ansicht verfügbar:

- **Business Services**
  Dient zur Erstellung und Pflege von Business Services. Detaillierte Informationen finden Sie in der *Business Services*-Dokumentation.

  Diese optionale Komponente setzt voraus, dass EntireX installiert ist. Außerdem muss zusätzliche Software auf einem Server installiert sein. Siehe *Voraussetzungen* in der *Business Services*-Dokumentation.
- **Code Generation**
  Dient zur Generierung von Natural-Subprogrammen und Datenbereichen (Data Areas). Ausführliche Informationen finden Sie in der *Code Generation*-Dokumentation.

12 Verschiedene Modi für die Entwicklung von Natural-Anwendungen

- Allgemeine Informationen zu NaturalONE 66
- Lokaler Modus 66
- Natural-Server-Modus (Natural Server Mode) 67

Dieses Dokument behandelt die folgenden Themen:

Allgemeine Informationen zu NaturalONE

NaturalONE bietet zwei Modi für die Entwicklung von Natural-Anwendungen. Der eine, der Natural-Server-Modus, ähnelt dem Natural Studio, wenn es als Entwicklungs-Client in einer SPoD-Umgebung eingesetzt wird. Der andere, der so genannte lokale Modus, ist der bevorzugte Entwicklungsmodus in NaturalONE. In diesem Modus können Sie so arbeiten, wie es ein Eclipse-Benutzer erwartet. Wenn Sie gewohnt sind, nativ auf einem Natural-Server oder mit Natural Studio in einer SPoD-Umgebung zu arbeiten, müssen Sie beim Wechsel in den lokalen Modus einen „Paradigmenwechsel“ vollziehen. Im lokalen Modus werden die Quellcodes nicht mehr direkt auf dem Natural-Server gespeichert oder geändert. Der zentrale Ort für die Speicherung der Quellcodes ist nun der Eclipse-Arbeitsbereich, der mit einem Versionskontrollsystem verbunden ist.

Weitere Informationen zu diesen Entwicklungsmodi finden Sie unter den nachfolgend beschriebenen Themen.

Lokaler Modus

Dies ist der bevorzugte Weg für die Arbeit mit Eclipse.

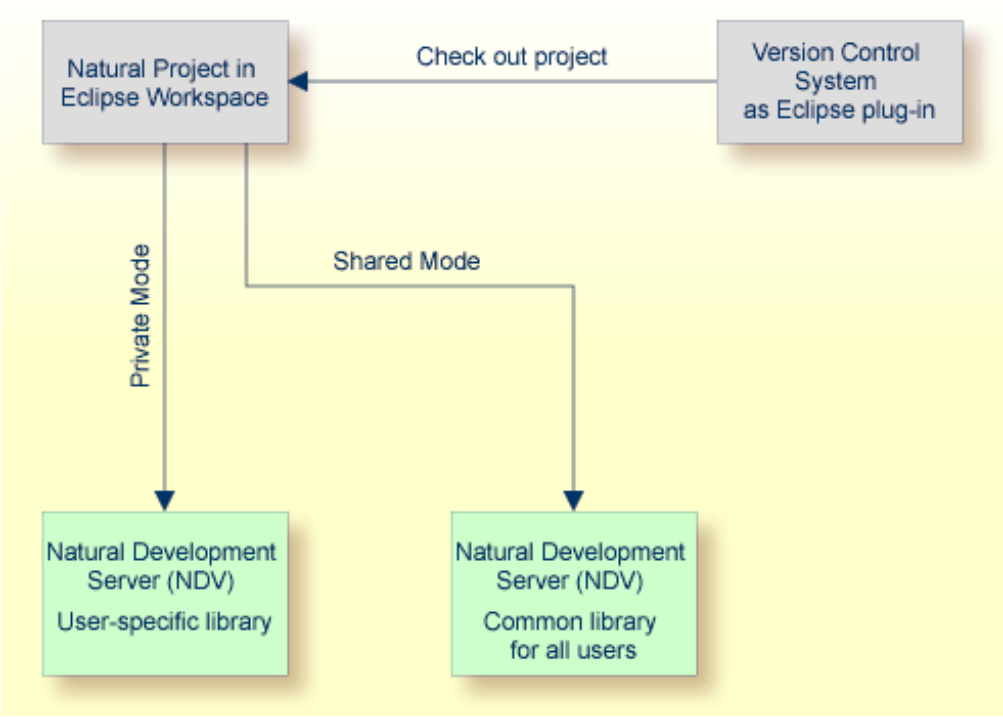
In der Regel laden Sie eine Library von einem Natural-Server in ein Natural-Projekt im lokalen Eclipse-Arbeitsbereich (Eclipse-Workspace) herunter und nehmen dann Ihre Änderungen am Quellcode vor. Um das Projekt zu bauen, müssen Sie den entsprechenden Natural-Server aktualisieren, d. h. Sie laden die Änderungen auf den Server hoch und katalogisieren sie dort.

Wenn mehrere Entwickler an der gleichen Natural-Anwendung arbeiten, ist die Natural-Anwendung nun auf mehrere Eclipse-Workspaces auf verschiedenen PCs verteilt. In den Projekteigenschaften können zwei verschiedene Modi definiert werden, in denen der Build durchgeführt werden soll:

- **Gemeinsamer Modus (Shared Mode)**
  Der Build wird in einer gemeinsamen Library auf dem Natural-Server durchgeführt, die von allen Benutzern gemeinsam genutzt wird. Dies ist der Standardmodus.
- **Privater Modus (Private Mode)**
  Der Build wird in einer benutzerspezifischen privaten Library auf dem Natural-Server durchgeführt. Dadurch wird vermieden, dass ein Entwickler die Änderungen eines anderen Entwicklers überschreibt.

Ausführliche Informationen zu diesen Modi finden Sie unter *Steplibs* in *Changing the Project Properties* (Ändern der Projekteigenschaften) in der Dokumentation *Using NaturalONE*.

Da die Eclipse-Funktionen oder Plug-ins (z. B. ein Plug-in für ein Versionskontrollsystem) erwarten, dass die Quellcodes im Eclipse-Arbeitsbereich gespeichert werden, gewinnen Sie im lokalen Modus zusätzliche Produktivität, da Sie Werkzeuge von Drittanbietern nutzen können.



NaturalONE bietet Teamunterstützung, so dass mehrere Entwickler gleichzeitig und parallel an einer Anwendung arbeiten können. Daher werden die Konfigurationsdateien zusammen mit dem Projekt gespeichert und können so im Versionskontrollsystem versioniert werden. Compiler-Optionen werden immer mit einem Projekt gespeichert.

Bei NaturalONE wird der versionierte Quellcode immer als der ursprüngliche Quellcode betrachtet. Sie müssen darauf achten, dass Sie einen Quellcode nicht auf den Natural-Server kopieren und dann den Quellcode auf dem Server weiterbearbeiten. In diesem Fall schlägt die Teamunterstützung fehl. Siehe auch *Verwendung eines Versionskontrollsystems*.

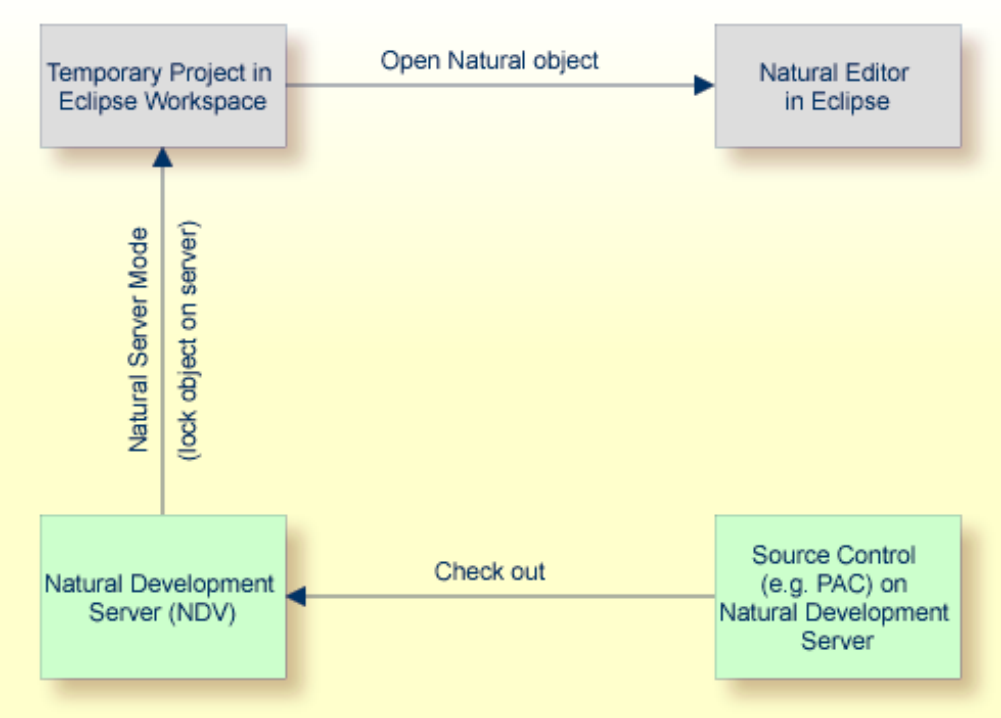
Natural-Server-Modus (Natural Server Mode)

Dies ist der klassische Weg für die Arbeit mit Natural Studio. Dieser Modus ahmt das aus Natural Studio bekannte SPoD-Verhalten nach.

In diesem Fall müssen Sie keine Libraries in den Eclipse-Arbeitsbereich herunterladen. Stattdessen arbeiten Sie virtuell direkt mit den Objekten auf einem Natural-Server. „Virtuell“ deshalb, weil die Objekte zu Bearbeitungszwecken vorübergehend in den Eclipse-Workspace heruntergeladen

werden. Solange Sie einen Quellcode bearbeiten, ist er auf dem Natural-Server gesperrt, um gleichzeitige Änderungen durch verschiedene Benutzer zu verhindern.

Da die Natural-Quellcodes auf dem Natural-Server verbleiben, findet auch die Versionierung auf dem Natural-Server statt.



Anmerkung: Im Natural-Server-Modus gibt es keine Unterscheidung zwischen privatem und gemeinsamem Modus.

13 Verwendung eines Versionskontrollsystems

Mit NaturalONE haben Sie die Möglichkeit, Natural-Anwendungen im Repository eines Versionskontrollsystems zu versionieren. Zu diesem Zweck werden die folgenden Werkzeuge unterstützt:

- **CVS**
  CVS wird mit Eclipse ausgeliefert. Das Workbench-Benutzerhandbuch in der Eclipse-Hilfe (http://www.eclipse.org/documentation/) enthält ausführliche Informationen zum Einrichten eines CVS-Repositorys und zur Team-Programmierung mit CVS.
- **Subversion (SVN)**
  Wenn Sie Subversion verwenden möchten, müssen Sie ein Eclipse-Plug-in wie Subclipse installieren. Subclipse kann von http://subclipse.tigris.org/ heruntergeladen werden. Achten Sie darauf, dass Sie eine Plug-in-Version herunterladen, die Ihre aktuelle Eclipse-Version unterstützt. Subversion gilt als der Nachfolger von CVS.
- **GIT**
  GIT ist ein verteiltes Versionskontrollsystem, das auf https://git-scm.com/ ausführlich beschrieben wird. Eine Eclipse-Plug-in-Version von GIT namens egit ist normalerweise bereits in den Standard-Eclipse-Distributionen enthalten.

CVS und SVN sind Client/Server-Anwendungen. Beide benötigen ein Eclipse-Plug-in als Benutzerschnittstelle und einen Serverteil (das Versionsverwaltungs-Repository), der die Anwendung verwaltet.

GIT ist ein verteiltes (oder dezentrales) Versionierungswerkzeug und als solches ist jedes GIT-Arbeitsverzeichnis ein vollwertiges Repository. Anstelle eines einzigen zentralen Server-Repositorys (wie bei CVS oder SVN) gibt es viele verteilte Repositories auf verschiedenen Rechnern.

Alle Plug-ins haben ihre eigenen Perspektiven. Jede dieser Perspektiven muss mit dem Kommando **Open Perspective** aktiviert werden.

Weitere Informationen finden Sie in der Dokumentation Ihres bevorzugten Versionskontrollsystems.

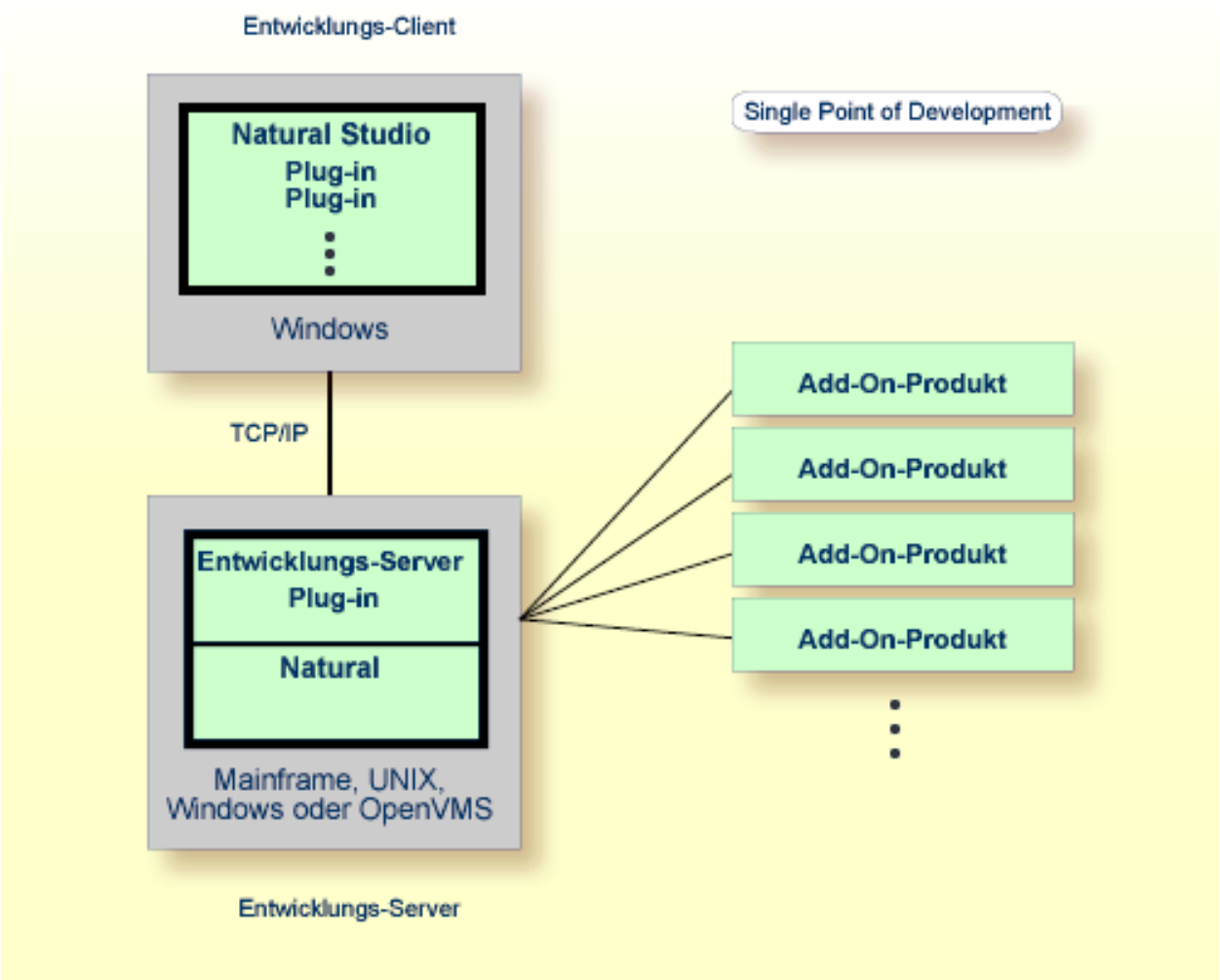
III Natural-SPoD-Architektur

14 Natural-SPoD-Architektur

Dieser Abschnitt beschreibt die Systemarchitektur des Natural Single Point of Development (SPoD).

SPoD ermöglicht eine zentralisierte Anwendungsentwicklung aus einer einzelnen Windows-Umgebung heraus. Sie können die Funktionalität von Natural Studio (Bestandteil von Natural für Windows) benutzen, um Natural-Anwendungen in einer Remote-Umgebung zu entwickeln, die sich auf einer Großrechner- oder UNIX-Plattform befindet.

SPoD basiert auf einem Client/Server-Konzept, das eine einzige Entwicklungsumgebung für alle Plattformen ermöglicht. Das folgende Diagramm veranschaulicht dieses Konzept und die Hauptbestandteile der SPoD-Architektur:



Development Client

Natural für Windows dient als Remote-Development-Desktop-Client für die Zielumgebung auf einer Großrechner oder einer UNIX-Plattform. Zum Desktop-Client gehört auch Natural Studio, die zentrale Workstation mit grafischer Benutzeroberfläche, auf der die Benutzer Anwendungen entwickeln können. Aus Natural Studio können sie alle Natural-Funktionen ausführen, die bei der Remote-Entwicklung benötigt werden.

Development Server

Das Natural-Development-Server-Plug-In ermöglicht die Remote-Entwicklung mit einem Natural, das in einer Zielumgebung auf einer Großrechner- oder UNIX-Plattform installiert ist. Das Natural auf der Zielplattform und das Natural-Development-Server-Plug-In bilden zusammen den Development Server.

Add-On-Produkt

Natural Studio bietet Plug-Ins, die installiert werden können, um ein oder mehrere Natural-Add-On-Produkte (z.B. Predict) in eine SPoD-Umgebung zu integrieren. Um ein Natural Studio Plug-In benutzen zu können, muss das entsprechende Add-On-Produkt in der Development-Server-Umgebung installiert sein.

Security

Um die Natural-Development-Server-Umgebung und die Natural-Base-Applications und Compound-Applications zu schützen, können Sie Natural Security einsetzen. Weitere Informationen siehe *Natural Security*-Dokumentation.

Verwandtes Thema:

Natural Single Point of Development-Dokumentation