Natural for Mainframes

Natural システムアーキテクチャ

バージョン 4.2.5

October 2009

Natural

This document applies to Natural バージョン 4.2.5 and to all subsequent releases.

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

目次

1 Natural システムアーキテクチャ

このドキュメントでは、Natural for Mainframes の基本的なシステムアーキテクチャと、Natural Single Point of Development（SPoD）のクライアント／サーバーアーキテクチャについて説明します。 SPoD を使用すると、複数プラットフォームに対応する開発を一元化できます。

	Natural の基本アーキテクチャの概要	Natural の主なアーキテクチャコンポーネント。
	Natural SPoD アーキテクチャ	Natural Single Point of Development の主なアーキテクチャコンポーネント。

2 Natural の基本アーキテクチャの概要

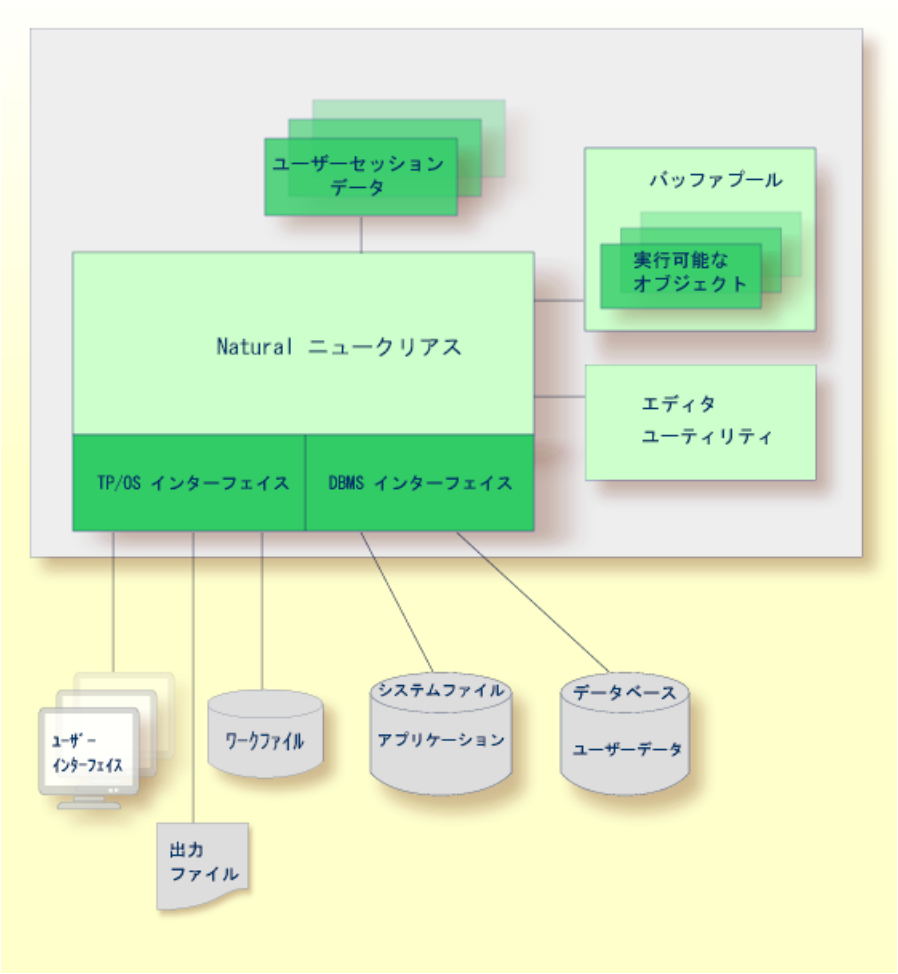
Natural for Mainframes のアーキテクチャの概念は、主にオープンアーキテクチャの原理に基づいており、インターフェイス定義という形で具体化されています。 アプリケーションの開発中も、プログラムの実行中も、このインターフェイスと外部コンポーネントの間で情報が交換されます。

Natural システムの機能は、Natural が実行されるすべてのプラットフォーム上で、適用される場合に使用できます。 Natural では、複雑なシステム環境（オペレーティングシステム、TP システム、データベースシステム、ユーザー環境）がアプリケーション開発環境から隠されています。

特別なシステム依存テーブルおよびルーチンによって、Natural コンポーネントがシステム環境に埋め込まれ、必要なリソースを持つ内部機能が使用可能になります。

このセクションでは、Natural の主なコンポーネントについてと、それらが連携して開発ツールとしての Natural の機能を提供する仕組みについて説明します。

各コンポーネントの詳細については、次の図で項目をクリックするか、以下のメニューから項目を選択してください。



Natural ニュークリアス	Natural ニュークリアスの主なコンポーネント： コンパイラ、ランタイムシステム、コマンドインタープリタ、設定（Natural パラメータ）。
ユーザーセッションデータ	ユーザー固有のワークエリア内の一時データストレージ。
Natural バッファプール	Natural バッファプールのオペレーション原理とオブジェクトのロード。
Natural エディタおよびユーティリティ	アプリケーションの構築と管理に使用される Natural エディタおよびユーティリティ。
TP/OS インターフェイス	Natural が提供する TP モニタおよび OS（オペレーティングシステム）インターフェイス。
ユーザーインターフェイス	Natural がサポートするユーザーインターフェイス。
出力ファイル ワークファイル	出力ファイルおよびワークファイルの使用。
Natural システムファイル	Natural システムファイルにおけるオブジェクトモジュールのストレージ。

DBMS インターフェイス データベースへのアクセス	Natural が提供するデータベース管理システム（DBMS）インターフェイス。 データベースに保存されているデータへの Natural によるアクセス。

3 Natural ニュークリアス

Natural ニュークリアスは、Natural のカーネルを構成するプログラムです。 ニュークリアスは、すべてのメインフレームオペレーティングシステム（z/OS、z/VSE、BS2000/OSD など）およびすべての TP モニタ（Com-plete、CICS、*open*UTM など）で稼働します。

Natural ニュークリアスは、Natural ユーザー（開発者、管理者など）に、コマンドの解釈、オブジェクトのコンパイル、オブジェクトの実行などのすべての Natural 機能を提供します。 Naturalニュークリアスは、複数のユーザーが同時に使用する共有プログラムとしてインストールできます。

このセクションでは、Natural ニュークリアスの主なコンポーネントについて説明します。

Natural ランタイムシステム

Natural ランタイムシステムは、アプリケーション内で Natural オブジェクトを実行するのに必要な環境を提供する仮想マシンと考えることができます。 Natural ランタイムシステムは、Natural 内部オブジェクトコード（バイナリメタコード）を解釈して実行します。

これ以降のセクションでは、次の項目について説明します。

- オブジェクトの実行
- オブジェクトスタータとオブジェクトエグゼキュータ

オブジェクトの実行

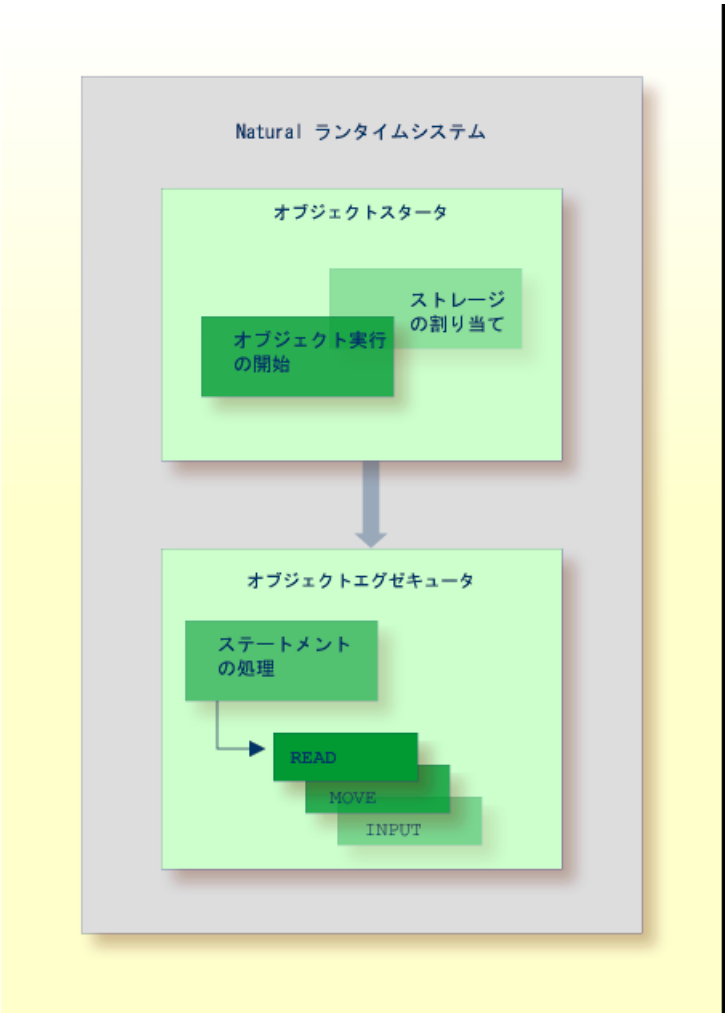
内部 Natural オブジェクトコードは、Natural オブジェクトに実行が要求されると実行されます。

オブジェクトの実行は、ユーザーによって直接要求されるか、または現在実行中のオブジェクトが、別のオブジェクトの実行を要求する Natural ステートメントを発行した場合に間接的に要求されます。 例えば、Natural システムコマンド `EXECUTE` は、このコマンドに指定されたユーザー作成プログラムを直接実行します。 また、Natural プログラムに指定された Natural `CALLNAT` ステートメントは、サブプログラムの実行を要求します。

『*Natural* バッファプール』の「*オブジェクトのロードと実行の例*」では、Natural プログラムの実行時のプロセスフローを説明しています。

オブジェクトスタータとオブジェクトエグゼキュータ

次の図は、Natural ランタイムシステムによるオブジェクトの実行を示しています。



Natural ランタイムシステムは、オブジェクトスタータとオブジェクトエグゼキュータという 2 つの主要コンポーネントで構成されています。

オブジェクトスタータは、実行するオブジェクトを **Natural** バッファプールで探し、そのオブジェクトによって処理される変数データ用のストレージをユーザーセッションデータとして割り当てます。 最後に、制御をオブジェクトエグゼキュータに渡します。

オブジェクトエグゼキュータは、オブジェクトに含まれている Natural ステートメントを解釈し、順番に実行していきます。 例えば上の図では、オブジェクトエグゼキュータはまず `READ` ステートメントを処理して、データベースを呼び出し、要求されたレコードを取得します。それが終わると、`MOVE` ステートメントの処理に移ります。

関連トピック：

- *ユーザーセッションデータ*
- ***Natural** バッファプール*
- *オブジェクトのロードと実行の例*（『*Natural* バッファプール』）

Natural コンパイラ

Natural コンパイラは、Natural ソースコードの実行可能形式を生成します。Natural ソースコードは人間が読み取れる形式のプログラミングコードで、Natural ステートメントのシーケンスで構成されています。Natural ステートメントとプログラミングの詳細については、『ステートメント』ドキュメントと『*プログラミングガイド*』を参照してください。

Natural コンパイラは、ソースエリアからソースコードを読み取ります。ソースエリア内のソースコードは、ユーザーセッションデータの一部です。 ソースコードをソースエリアに読み取るには、Natural システムコマンド `READ` または `EDIT` を使用します。 コンパイラは、ソースコードの構文をチェックし、問題がない場合は**Natural** ランタイムシステムが解釈および実行できる Natural 内部オブジェクトコードを生成します。

適切な Natural システムコマンドを使用することで、ソースコードをコンパイルして実行するかコンパイルして保存することができます。 以下のセクションでは、ストレージとして使用できるオブジェクトモジュールのタイプと、オブジェクトのコンパイルおよび実行に使用される Natural システムコマンドについて説明します。

- **カタログ化オブジェクト**
- **ソースオブジェクト**
- **コンパイルコマンド**
- **コンパイルの例**

カタログ化オブジェクト

カタログ化オブジェクトとは、Natural オブジェクトの実行形式（コンパイルされた形式）のことです。 カタログ化オブジェクトは、Natural コンパイラによって作成され、オブジェクトモジュールとして Natural システムファイルに格納されます。 ソースコードのコンパイルおよびカタログ化オブジェクトの作成を、オブジェクトのカタログ化と呼びます。カタログ化オブジェクトは、Natural システムコマンド `CATALOG` または `STOW` を使用して作成されます。

実行時にカタログ化オブジェクトは Natural バッファプールにロードされ、Natural ランタイムシステムによって実行されます。 Natural オブジェクトは、カタログ化オブジェクトとして Natural システムファイルに格納されている場合にのみ、実行または相互参照が可能となります。

カタログ化オブジェクトを変更または逆コンパイルすることはできません。

ソースオブジェクト

ソースオブジェクト（保存オブジェクト）には、人間が理解できる形式のNaturalソースコードが含まれています。 ソースコードは、Naturalシステムコマンド SAVE または STOW を使用して、Natural システムファイルにソースオブジェクトとして保存されます。

ソースオブジェクトに含まれているソースコードを実行するには、ソースコードをコンパイルして、Naturalランタイムシステムが解釈および実行できる生成済みのオブジェクトコードを作成する必要があります。

コンパイルコマンド

Naturalには、ソースコードをコンパイルするために異なるシステムコマンドが用意されています。 使用する Natural システムコマンドに応じて、コンパイルは次の任意のアクションと組み合わされます。

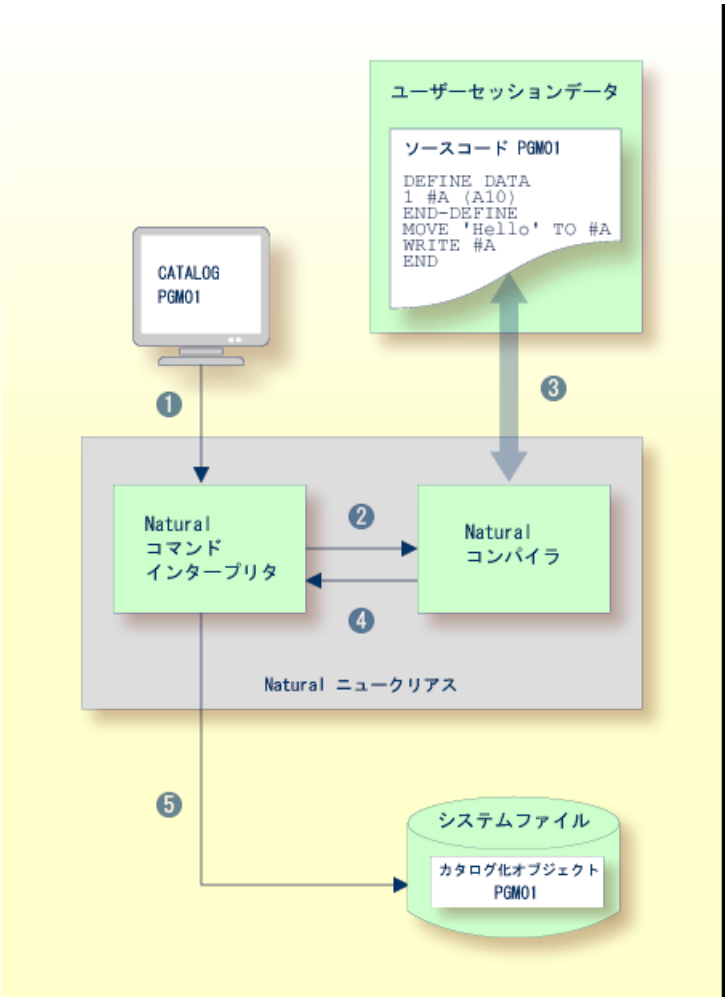
アクション	システムコマンド
ソースコードをコンパイルします。構文チェックを実行し、オブジェクトコードを生成します。 生成されたオブジェクトコードは、Natural システムファイルにオブジェクトモジュールとして格納されません。	CHECK
ソースコードをコンパイルします。成功した場合は、カタログ化オブジェクトとして生成されたオブジェクトコードを Natural システムファイルに格納します。	CATALOG
ソースコードをコンパイルします。成功した場合は、カタログ化オブジェクトとして生成されたオブジェクトコードを Natural システムファイルに格納します。 また、元のソースコードを別のソースオブジェクトとして Natural システムファイルに格納します。	STOW
プログラムタイプのNaturalオブジェクトのソースコードをコンパイルします。 成功した場合は、生成されたオブジェクトコードを直ちに実行します。生成されたオブジェクトコードは、Natural システムファイルにオブジェクトモジュールとして格納されません。	RUN

関連トピック：

- 『システムコマンド』ドキュメント
- *オブジェクトタイプ*（『*プログラミングガイド*』）

コンパイルの例

次の図は、Natural システムコマンド CATALOG を使用してソースコードをコンパイルしたときのプロセスフローを示しています。



説明

1. ユーザーが、Natural システムコマンド CATALOG PGM01 を発行して、ソースエリアに現在保存されているソースコードをコンパイルすること、および生成されたオブジェクトコードを PGM01 という名前のカタログ化オブジェクトとして保存することを要求します。
2. Natural コマンドインタープリタが、コマンド CATALOG を解釈し、要求を Natural コンパイラに渡します。
3. Natural コンパイラが、ソースエリアに現在保存されている Natural ステートメントを読み取り、構文をチェックし、オブジェクトコードを生成します。
4. Natural コンパイラが、制御を Natural コマンドインタープリタに返します。

5 Natural コマンドインタープリタが、生成されたオブジェクトコードを PGM01 という名前のカタログ化オブジェクトとして現在の Natural システムファイルに保存します。

Natural コマンドインタープリタ

Natural コマンドインタープリタは、ユーザーによって Natural コマンドプロンプトから入力されたコマンドをチェックして実行します。

Natural Security がインストールされている場合は、コマンドの実行を 1 人のユーザーまたは1つのユーザーグループに制限できます。

関連トピック：

- 『 システムコマンド』ドキュメント
- 『*Natural Security*』ドキュメント

設定

Natural パラメータは、Natural 環境のコンフィグレーションを管理します。

Natural パラメータを使用すると、開発／生成プロセスの標準化や自動化、または個々のユーザーのニーズに応じた標準設定の調整を行うことができます。例えば、レポート作成時のデフォルト値の設定、レポートサイズの定義、またはエディタのソースエリアなど必要とされるストレージエリアサイズの定義を行う場合に、Natural パラメータを使用します。

Natural 環境の特質の大半はすでに Software AG によって定義されていますが、 Natural 管理者は、すべての Natural ユーザーに対して有効となる別のデフォルト環境設定を構成することができます。 また、個々のユーザーは、デフォルト環境設定にダイナミックプロファイルパラメータまたはセッションパラメータを上書きするという方法で、自身のニーズに応じた設定を適用することができます。

これ以降のセクションでは、次の項目について説明します。

- プロファイルパラメータ
- セッションパラメータ

プロファイルパラメータ

プロファイルパラメータは、スタティックまたはダイナミックに指定されます。

スタティックパラメータは、Natural のインストール中に Natural パラメータモジュール NATPARM で指定されます。 各 Natural セッションのデフォルトとして使用されます。

ダイナミックパラメータは、Natural の起動時に指定されます。 Natural SYSPARM ユーティリティを使用すると、ダイナミックパラメータセットを事前定義することができます。

関連トピック：

- *プロファイルパラメータ*（『パラメータリファレンス』ドキュメント）
- *プロファイルパラメータの使用方法*（『オペレーション』ドキュメント）
- *Natural パラメータモジュールの使用*（『オペレーション』ドキュメント）
- *SYSPARM ユーティリティ*（『ユーティリティ』ドキュメント）

セッションパラメータ

セッションパラメータは、アクティブな Natural セッション内や Natural オブジェクト内で指定されます。 セッションパラメータの主な目的は、Natural プログラムの実行を制御することです。

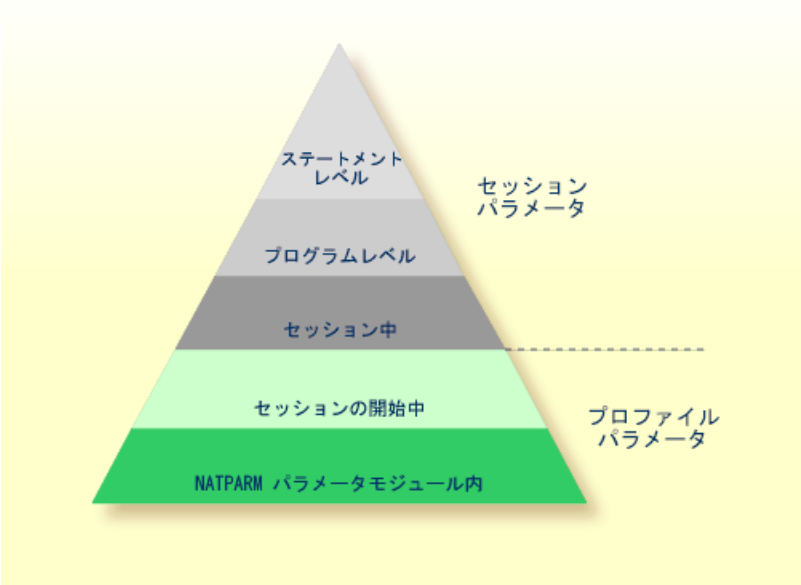
関連トピック：

- *セッションパラメータ*（『パラメータリファレンス』ドキュメント）

パラメータ化のレベル

Natural パラメータの設定レベルは階層構造になっています。 上のレベルで設定されたパラメータ値は、下のレベルで定義された値より優先されます。 例えば、パラメータをダイナミックに指定すると、Natural パラメータモジュール NATPARM で対応するパラメータに対して設定されたスタティックな指定より、新しいパラメータ値が優先されます。

下の図は、Naturalパラメータの設定可能なタイミングとその階層構造を（底辺が最下レベル、頂点が最上レベルのピラミッド方式で）示したものです。



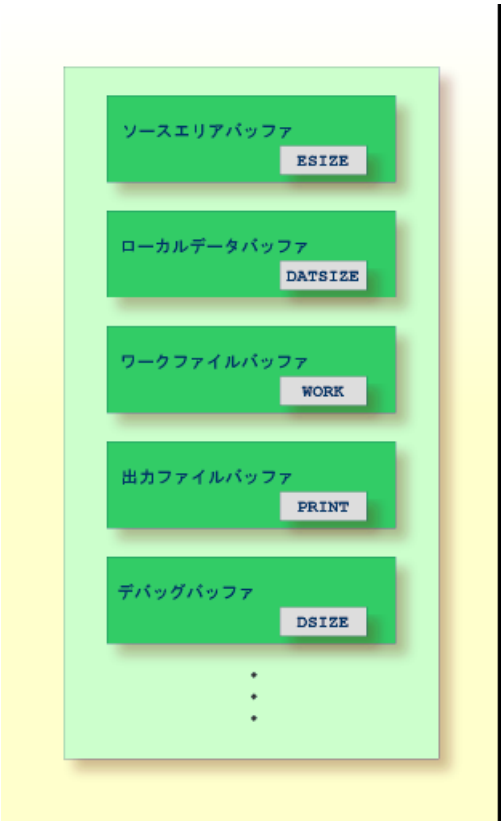
関連トピック：

- *Natural パラメータ階層*（『オペレーション』ドキュメント）

4 ユーザーセッションデータ

各Naturalユーザーセッション（オンラインまたはバッチ）は、このセッションのデータを含む別のワークエリアに割り当てられます。 ユーザーセッションデータは、オンラインセッションまたはバッチセッションの間、保存されます。

ワークエリアは、Natural バッファのセットで構成されます。 バッファとは、Natural プログラムで参照される変数データなどの特定のデータブロックを保持するストレージユニットです。



上の図に示されているように、バッファのサイズは、内部および変数 Natural バッファを除いて、Natural パラメータを使用して指定できます。 例えば、Natural プロファイルパラメータ ESIZE では、ソースコード編集に使用する Natural エディタに割り当てられたソースエリアのサイズを指定します。 プロファイルパラメータ DATSIZE では、Natural プログラム（およびそのプログラムによって下位オブジェクトが呼び出された場合はそのオブジェクト）のローカルデータを保持するために実行時に割り当てられるサイズを指定します。

Naturalは、現在のセッションに割り当てられたバッファ、そのサイズ、および実際に使用されているバッファスペースに関するバッファ使用統計を提供します。詳細については、次の*関連トピック*を参照してください。

関連トピック：

- 『*パラメータリファレンス*』ドキュメント
- *バッファ使用統計 - BUS*（『*ユーティリティ*』ドキュメントの「*一般的な SYSTP 機能*」）
- *BUS*（『*システムコマンド*』ドキュメント）

5 Natural バッファプール

共有メモリである Natural バッファプールは、一時ストレージエリアとして機能し、そこに Natural カタログ化オブジェクト（プログラムなど）が Natural システムファイルからロードされて後で Natural ランタイムシステムによる実行や Natural コンパイラによるオブジェクトコンパイルに使用されます。

Natural ではリエントラントな Natural オブジェクトコードが生成されるので、Natural オブジェクトの 1 つのコピーを複数のユーザーが同時に実行できます。 この目的のため、各オブジェクトは、呼び出し元が呼び出すたびにロードされるのではなく、Natural システムファイルから Natural バッファプールに 1 回だけロードされます。

このセクションでは、バッファプールオペレーションの基本原理とオブジェクトのロード手順について説明します。

オブジェクトのロード

バッファプールへのオブジェクトのロードは、実行可能な Natural オブジェクトの実行が要求されたときに初期化されます。

オブジェクトの実行が要求されると、対応するカタログ化オブジェクトが、該当する **Natural** システムファイルからバッファプールへロードされ、そこで **Natural** ランタイムシステムによる実行が可能になります。

実行可能な Natural オブジェクト以外に、タイプがデータエリア（ローカル、グローバル、パラメータ）である非実行可能オブジェクトも、このようなデータエリアを参照する Natural オブジェクトがカタログ化または再カタログ化（コンパイル）されている場合はバッファプールにロードされます。

関連トピック：

- *オブジェクトの実行*（「*Natural* ニュークリアス」の「*Natural* ランタイムシステム」）

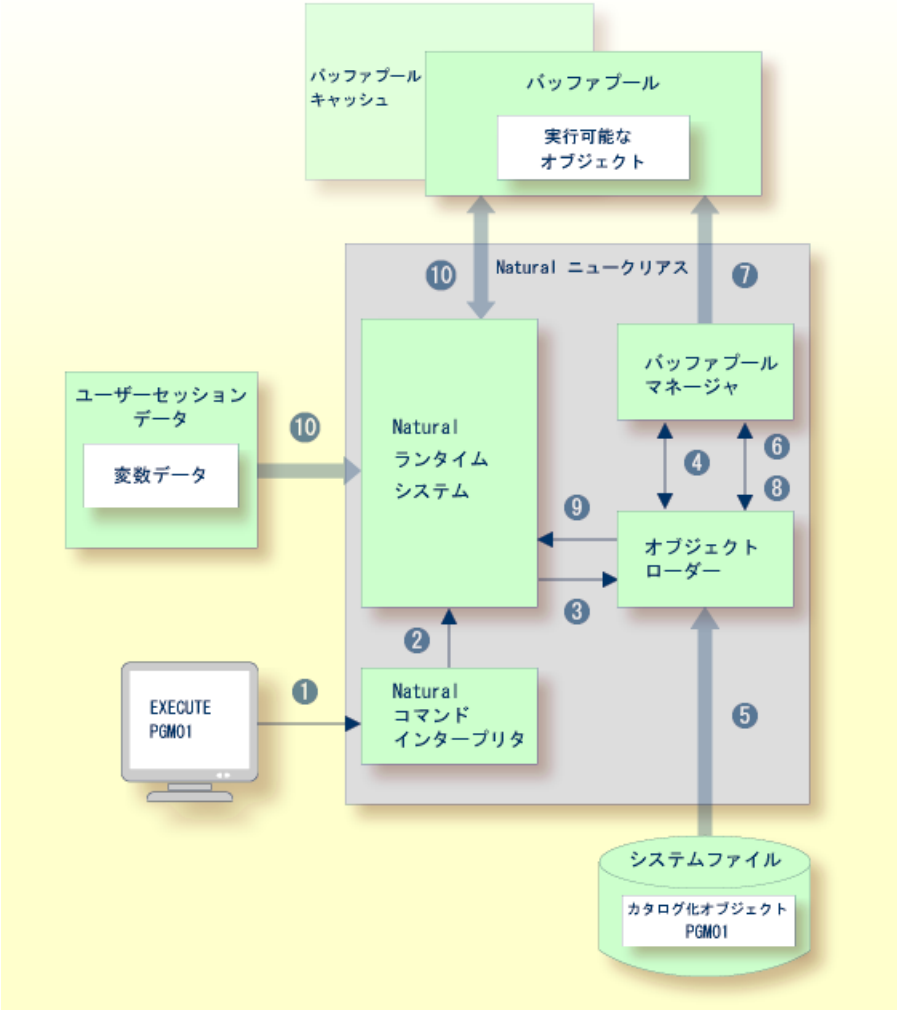
オブジェクトの削除

新しいオブジェクトをロードするためのスペースが必要になった場合は、ユーザーから参照されなくなったオブジェクトがバッファプールから削除されます。 これにより、バッファプールで使用可能なストレージスペースを最大限使用できます。

バッファプールキャッシュがある場合は、使用されていないオブジェクトがバッファプールからバッファプールキャッシュにスワップされます。使用されていないオブジェクトがバッファプールキャッシュからも削除された場合、またはバッファプールキャッシュが存在しない場合に、ユーザーがそのオブジェクトを再び参照すると、オブジェクトは Natural システムファイルから再ロードされます。

オブジェクトのロードと実行の例

次の図は、Natural プログラムをロードして実行するときのプロセスフローを示しています。



説明

1. ユーザーが、Natural システムコマンド EXECUTE を発行して、プログラムタイプで名前が PGM01 の Natural オブジェクトの実行を要求します。
2. Natural コマンドインタープリタが、コマンドを解釈し、要求を Natural ランタイムシステムに渡します。
3. Natural ランタイムシステムが、要求をオブジェクトローダーに渡します。
4. オブジェクトローダーが、バッファプールマネージャに対し、バッファプール（および存在する場合はバッファプールキャッシュ）で PGM01 を探して、バッファプールに保存されている PGM01 のアドレスを特定するよう指示します。

   バッファプールマネージャが PGM01 のバッファプールアドレスを見つけた場合は、バッファプールがそのアドレスをオブジェクトローダーに渡します。8 の処理に進みます。
   バッファプールマネージャが PGM01 のバッファプールアドレスを見つけられなかった場合は、バッファプールマネージャが、否定的な検索結果をオブジェクトローダーに返します。オブジェクトの実行は 5 の処理に進みます。
5. オブジェクトローダーが、該当する Natural システムファイルから PGM01 のカタログ化オブジェクトを取得します。
6. オブジェクトローダーが、PGM01 のカタログ化オブジェクトをバッファプールマネージャに渡します。
7. バッファプールマネージャが、PGM01 をバッファプールにロードします。
8. バッファプールマネージャが、PGM01 のバッファプールアドレスをオブジェクトローダーに返します。
9. オブジェクトローダーが、PGM01 のバッファプールアドレスを Natural ランタイムシステムに渡します。
10. Natural ランタイムシステムが、PGM01 のカタログ化オブジェクトのコンパイル済みコードを解釈して実行します。

関連トピック

Natural バッファプールについて、『*Natural* システムアーキテクチャ』ドキュメントでは次のトピックを参照してください。

- ***Natural* システムファイル**
- ***Natural* ランタイムシステム**（「*Natural* ニュークリアス」）
- *カタログ化オブジェクト*（「*Natural* ニュークリアス」の「*Natural* コンパイラ」）

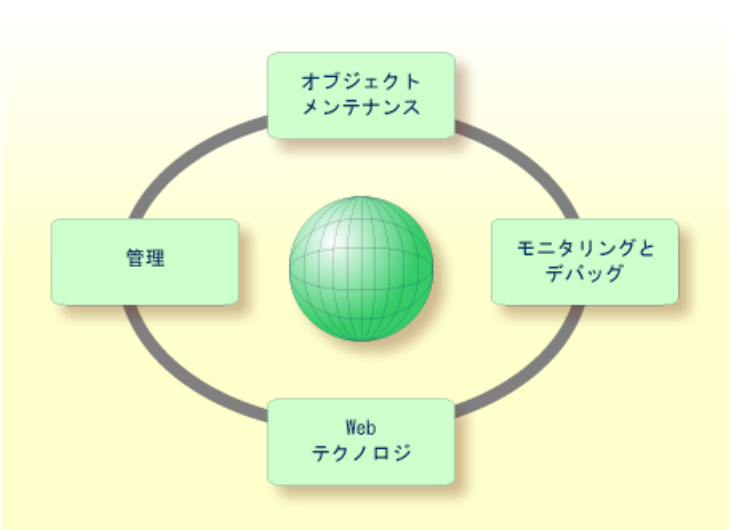
その他の Natural ドキュメントでは、次のトピックを参照してください。

- *Natural* バッファプール（『オペレーション』）
- *Natural* ストレージ管理（『オペレーション』）

6 Natural エディタおよびユーティリティ

Naturalユーティリティは、アプリケーション開発、管理、およびオペレーション環境の管理に使用する機能セットを提供するツールです。

次の図は、Natural エディタおよびユーティリティの主な機能目的を示しています。



Natural エディタまたはユーティリティは、次の目的で使用できます。

- プログラム、サブプログラム、データ定義モジュールなどのNaturalオブジェクト（または外部オブジェクト）の処理。
- ハードウェアプラットフォーム間のアプリケーション転送、リモートプロシージャコールの制御、レポートの生成、アプリケーション固有のエラーメッセージの管理などの管理機能の実行。
- アプリケーションのモニタとデバッグ。
- Web サーバーへのアクセスや XML ドキュメントの処理などの Web テクノロジの使用。

関連トピック：

- *Natural データ定義モジュール*（「*DBMS インターフェイス - データベースアクセス*」）
- 『エディタ』ドキュメント
- 『ユーティリティ』ドキュメント
- *Natural Web インターフェイス*（『*Web テクノロジ*』ドキュメント）
- *XML ツールキット*（『*Web テクノロジ*』ドキュメント）

7 TP/OS インターフェイス

Natural には、Natural ニュークリアスが、オンライントランザクション処理のために TP モニタにアクセスしたり、バッチ処理のためにオペレーティングシステム（OS）にアクセスしたりするためのインターフェイスが用意されています。

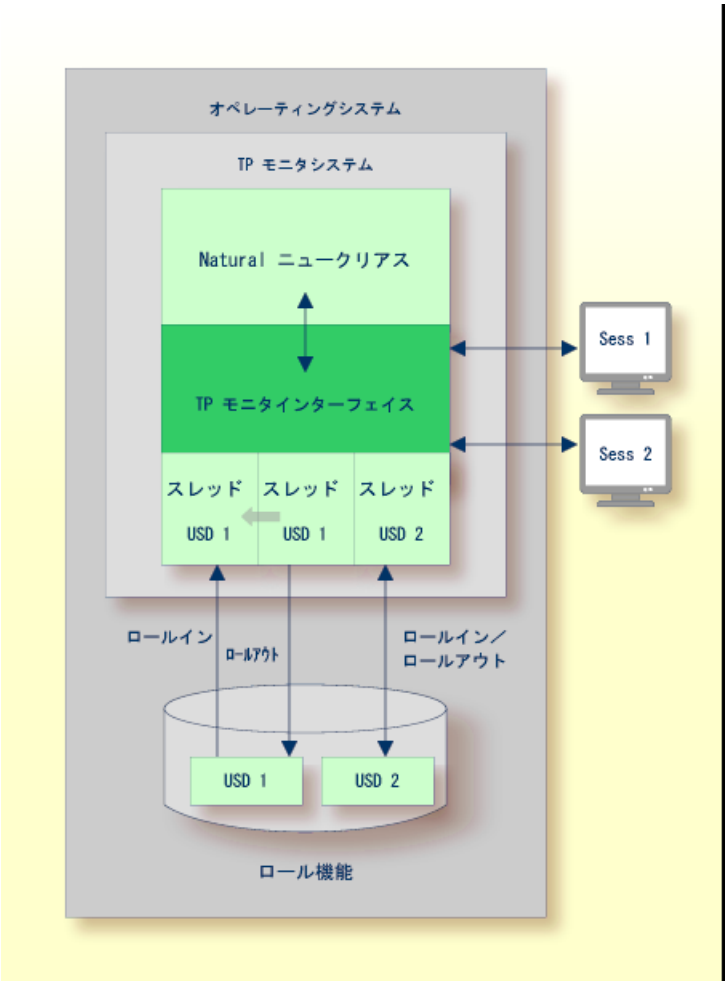
オンライン処理

TP モニタ環境において、Natural は標準 TP プログラムとして動作し、 その TP モニタの制御下で実行されているプログラムに適用されるルールに従います。

TP モニタ環境では、ユーザーセッションデータを含むワークエリアは Natural スレッドと呼ばれます。 これは、ユーザーセッションの間、維持されます。

Naturalスレッドは、セッションがユーザーからの端末入力を待機している間、ロールアウトしてロール機能になることができます。 スレッドをロールアウトすると、ストレージは他のユーザーセッション用に解放されます。Naturalスレッドは、ユーザーが端末入力を行ったとき（例えば Enter キーを押したとき）にロールインされます。 スレッドは別のストレージエリアに再配置できます。次の図の USD1 がこれを示しています。USD1 は、ユーザーセッション Sess 1 のユーザーセッションデータ（USD）を含むスレッド内にあります。

次の図の例では、TP モニタ（ここでは Com-plete）が、Naturalスレッドをロールインしてロール機能にしたりロール機能からロールアウトしたりすることによって、ストレージ割り当てを管理しています。

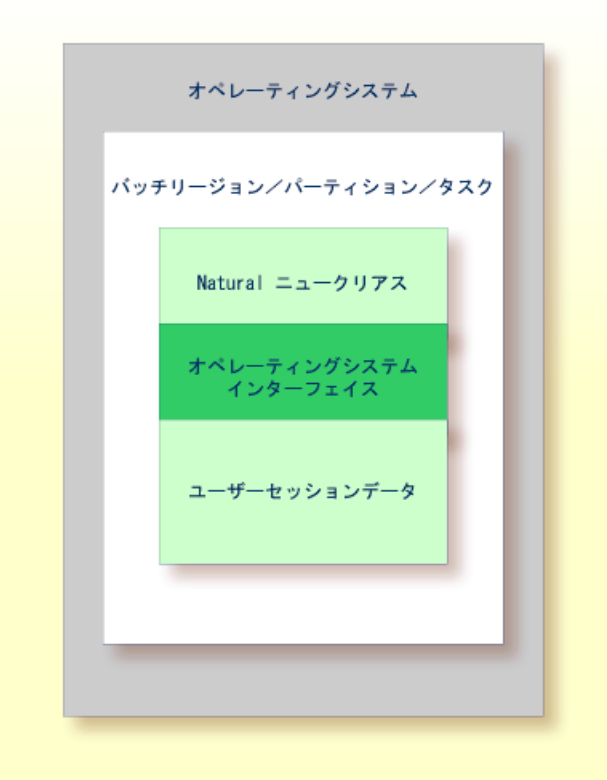


バッチ処理

バッチモードでは、Natural はバッチジョブによって初期化されたセッションを処理します。バッチジョブをサブミットしたユーザーとコンピュータとの対話は不要です。

バッチジョブは、順番の決まった入力データを受け取って順番に実行されるプログラムで構成されます。入力データはファイルから読み取られ、出力はファイルに書き込まれます。バッチジョブ用のワークエリアが存在する場所は、オペレーティングシステムによって異なり、バッチリージョン（z/OS）、パーティション（z/VSE）、またはタスク（BS2000/OSD）です。ワークエリアにはユーザーセッションデータが保存され、バッチユーザーセッションの間、維持されます。

次の図は、バッチ処理環境を示しています。



提供されている Natural TP/OS モニタインターフェイス

Natural が提供する TP インターフェイスと、Natural を TP モニタとともに使用する方法については、『*TP モニタインターフェイス*』ドキュメントで次のセクションを参照してください。

- *Natural での TP モニタの使用*
- *CICS 環境での Natural*
- *Com-plete/SMARTS 環境での Natural*
- *IMS/TM 環境での Natural*
- *TIAM 環境での Natural*
- *TSO 環境での Natural*
- *openUTM 環境での Natural*
- *VM/CMS 環境の Natural*

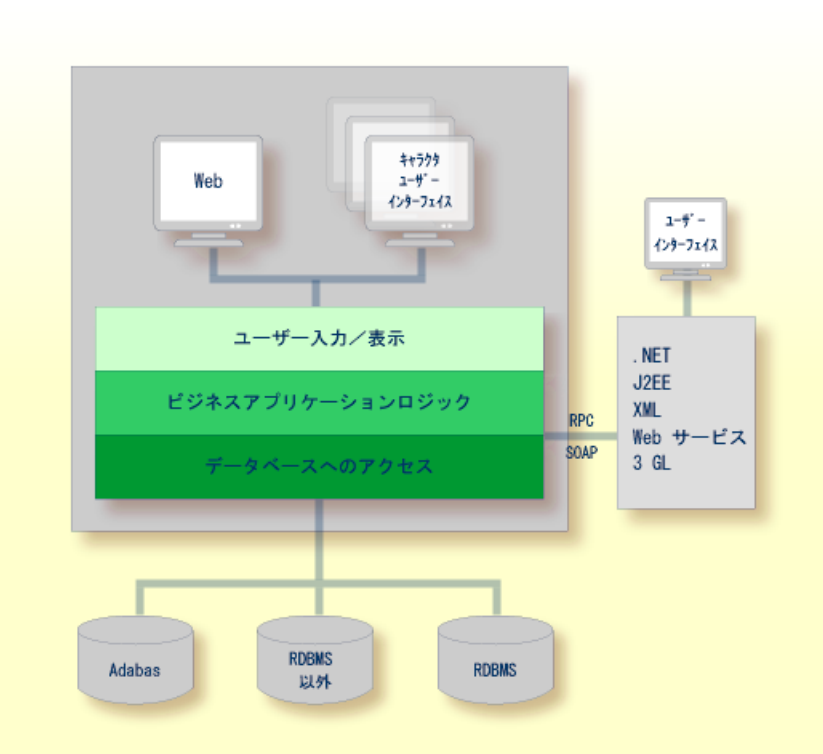
バッチモードでの Natural の実行の詳細については、『*オペレーション*』ドキュメントの「*バッチモードでの Natural*」を参照してください。

8 ユーザーインターフェイス

Naturalでは、プレゼンテーション、ビジネスアプリケーションロジック、およびデータベースアクセスから成る多層アーキテクチャを採用した配布可能なアプリケーションをサポートしています。Naturalアプリケーションはモジュラーであり、各アプリケーション層で要求される適切なタイプのオブジェクトを提供します。

Natural では、Web ベースのインターフェイス（HTML、XML など）、キャラクタユーザーインターフェイス（CUI）を使用するプロセスドリブンアプリケーション、イベントドリブンのグラフィカルユーザーインターフェイス（GUI）など、各種ユーザーインターフェイスによるプログラミングが可能です。

また、Natural アプリケーションは、J2EE や .NET など、非 Natural 環境に接続されたユーザーインターフェイスとの対話も可能です。 Natural 環境と非 Natural 環境との接続の確立には、リモートプロシージャコール（RPC）や SOAP 要求などを使用します。これにより、クライアントコンピュータ上のプログラムが、Natural RPC サーバー上にあるサブプログラムタイプの Natural オブジェクトを実行できます。


Web
キャラクタ
ユーザー
インターフェイス
ユーザー入力／表示
ビジネスアプリケーションロジック
データベースへのアクセス
RPC
SOAP
ユーザー
インターフェイス
.NET
J2EE
XML
Web サービス
3 GL
Adabas
RDBMS
以外
RDBMS

9 出力ファイル - ワークファイル

出力ファイルとワークファイルは、Naturalオブジェクトで処理されるデータの永続ストレージまたは中間ストレージのためのデータコンテナです。このようなデータコンテナには、物理シーケンシャルファイル、パーティション分割されたデータセットのメンバ、オペレーティングシステムまたは TP モニタが提供するテープデータセットなどがあります。 Natural は、出力ファイルとワークファイルに順番にアクセスします。

出力ファイルには、レポートをプリンタに出力するために必要なレポートデータとプリンタ制御文字が保存されます。

ワークファイルには、Natural オブジェクト間で交換可能なデータ、または Natural オブジェクトと COBOL などの他のプログラミング言語で記述されたオブジェクトとの間で交換可能なデータが保存されます。

出力ファイルとワークファイルは、バッチ処理またはオンライン処理において、インストールされているオペレーティングシステムまたは TP モニタの規則に従って使用できます。

ワークファイルを使用したオブジェクトの転送

次の図は、ワークファイルに保存されているデータを使用して、各種プラットフォーム上のさまざまな Natural 環境間で Natural オブジェクトを転送する仕組みを示しています。 このためには、Naturalユーティリティ（オブジェクトハンドラなど）を使用して、ソース環境のオブジェクトをワークファイルにアンロードし、それをワークファイルからターゲット環境にロードします。必要に応じて、FTPなどのアプリケーションプロトコルを使用して、ワークファイルをソース環境からターゲット環境に転送します。

出力ファイルとワークファイルの定義とアクセス

出力ファイルとワークファイルは、Natural環境用に論理的に定義されており、Naturalパラメータや基盤となるオペレーティングシステムの制御ステートメントまたはTPモニタを使用して、ファイルまたはプリンタに物理的に割り当てることができます。 割り当ては、ランタイムに現在のセッション用に変更できます。

適切なNaturalステートメントを使用することで、ワークファイルへのデータの書き込み、ワークファイルからのデータの読み取り、または出力ファイルへのデータの書き込みを行うことができます。

10 Natural システムファイル

Natural システムファイルには、ユーザー定義アプリケーションまたはSoftware AGが提供するNatural システムアプリケーション（ユーティリティなど）に属するNatural オブジェクトが保存されます。

システムファイルに保存されるNatural オブジェクトには、「Natural ニュークリアス」セクションの「Naturalコンパイラ」で説明されているように、カタログ化オブジェクトとソースオブジェクトがあります。Naturalシステムファイルは、データベースファイルまたはストレージシステム（AdabasやVSAMなど）に保存されます。システム環境によっては、Naturalシステムファイルを異なるデータベースファイルまたはストレージシステムに配置できます。

Natural システムファイルは、ユーザーがNatural オブジェクトを読み取ったり変更したりするとアクセスされます。また、システムファイルは、オブジェクトが後続の実行のためにバッファプールにロードされたときにもアクセスされます（「*Natural バッファプール*」セクションを参照）。

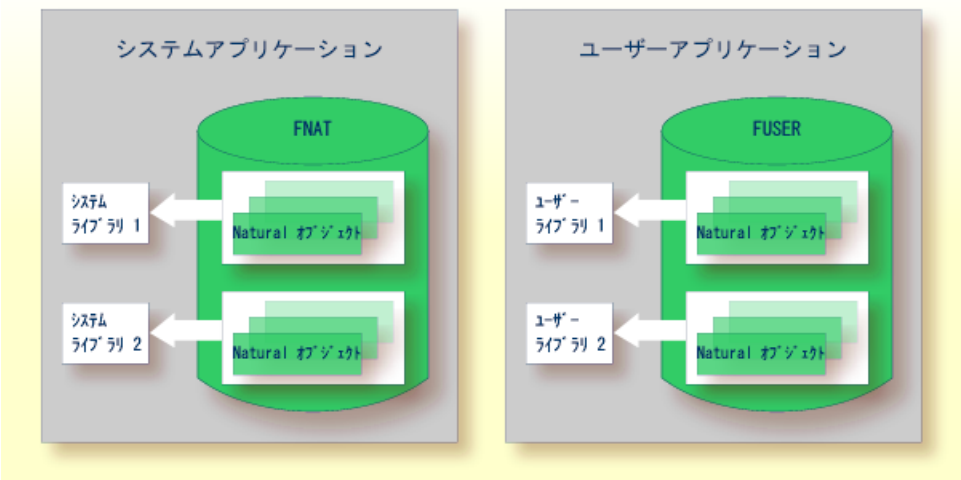
システムファイルのタイプ

次の表に、Natural環境で一般的に使用可能なNaturalシステムファイルの一覧を示します。システムファイルFNATおよびFUSERは、Natural for Mainframesで提供されます。 Natural SecurityなどのNaturalアドオン製品がインストールされている場合は、追加のシステムファイルがNaturalとともに提供されます。

システムファイル	内容	アドオン製品
FNAT	Natural システムアプリケーションに必要なすべてのオブジェクト。	
FUSER	ユーザー定義アプリケーションに必要なユーザー固有オブジェクト。	
FSEC	セキュリティ定義に必要な制御情報。	Natural Security がインストールされている場合に適用されます。
FSPOOL	レポートを画面またはプリンタに出力して出力統計を取得するために必要な制御およびスプーリング情報。	Natural Advanced Facilities がインストールされている場合に適用されます。
FDIC	**Natural** データ定義モジュール。	Predictがインストールされている場合は、FDICに Predict ディクショナリシステムのデータも含まれます。 Natural 開発サーバーがインストールされている場合は、FDIC にアプリケーションデータも含まれ、オブジェクトロック情報も保存されます。

システムファイル内のライブラリ

Natural システムファイルの FNAT と FUSER に保存される Natural オブジェクトは、次の図に示すように、ライブラリと呼ばれる論理構成にグループ化されます。



関連トピック：

- *カタログ化オブジェクト*および*ソースオブジェクト*（「*Natural* ニュークリアス」の「*Natural* コンパイラ」）
- 『*Natural Security*』ドキュメント
- 『*Natural Advanced Facilities*』ドキュメント
- ***Natural** データ定義モジュール*（「*DBMS* インターフェイス - データベースアクセス」）
- 『*Predict*』ドキュメント
- 『*Natural Development Server*』ドキュメント

11 DBMS インターフェイス - データベースアクセス

Natural には、Adabas データベース、リレーショナルデータベース管理システム（RDBMS）、非 RDBMS などに保存されているデータにアクセスするための DBMS（データベース管理システム）インターフェイスが用意されています。

Natural アプリケーションは、複数の DBMS に保存されているユーザーデータに同時にアクセスできます。 出力ファイルやワークファイル（該当セクションを参照）に保存されているデータは順番にアクセスされますが、DBMS に保存されているデータにはランダムにアクセスできます。

サポートされる DBMS には、DB2、DL/I、VSAM データセットなどがあります。

このセクションでは、Natural でサポートされる DBMS についてと、Natural アプリケーションからデータベースのデータにアクセスする原理について説明します。

サポートされているデータベース管理システム

Natural には、次のデータベース管理システム（DBMS）用のインターフェイスが用意されています。

- **Adabas**
- **DL/I**
- **DB2**
- **VSAM**

Adabas

Natural for Adabas インターフェイスは、Adabas データベースに保存されているデータへのアクセスを可能にします。

Adabas は、Software AG のデータベース管理システムです。Adabas では、リレーショナルデータ構造およびネストされたリレーショナルデータ構造をサポートします。

Natural for Adabas インターフェイスは Natural に統合されており、ローカルコンピュータから Adabas データベースにアクセスできます。リモートアクセスの場合は、追加のルーティングおよび通信ソフトウェア（Software AG の Entire Net-Work など）が必要です。いずれにしても、Adabas データベースを実行しているホストマシンのタイプは Natural にとっては常に透過的です。

関連トピック：

- *Adabas データベースのデータへのアクセス*（『プログラミングガイド』）
- 『*Adabas*』ドキュメント

DL/I

Natural for DL/I インターフェイスは、IMS/DB データベースに保存されているデータへのアクセスを可能にします。

DL/I は、IBM の IMS/DB のデータ操作言語です。 IMS は、階層データベースおよびトランザクション管理システムです。

関連トピック：

- 『*Natural for DL/I*』ドキュメント

DB2

Natural for DB2 インターフェイスは、IBM の DB2 UDB for z/OS オペレーティングシステムに保存されているデータへのアクセスを可能にします。

関連トピック：

- 『*Natural for DB2*』ドキュメント

VSAM

Natural for VSAM インターフェイスは、VSAM データベースに保存されているデータへのアクセスを可能にします。

VSAM は、キー順データセット、エントリ順データセット、相対レコードデータセットなどの各種データセット編成のレコードを管理するための、IBM のアクセス方式です。

関連トピック：

- 『*Natural for VSAM*』ドキュメント

Natural データ操作言語

Natural データ操作言語（DML）は、Natural がサポートするすべての DBMS に共通のデータアクセス構文を提供します。 Natural DML を使用することで、Natural オブジェクトはさまざまなDBMSへのアクセスに、`FIND`、`READ`、`STORE`、`DELETE`など、同じ言語ステートメント（DML ステートメント）を使用できます。

Natural は、DBMS の種類をコンフィグレーションファイルから判断し、DML ステートメントをデータベース固有のコマンドに変換します。つまり、Natural が、Adabas データベース用のダイレクトコマンドや、SQL を使用する RDBMS 用の SQL ステートメント文列字とホスト変数構造を生成します。

データベース固有の節を*含まない*DML ステートメントを使用する Natural オブジェクトは、さまざまな DBMS で実行できます。 データベース固有の節を含む DML は、別の DBMS で実行する前に変更が必要です。 データベース固有の考慮事項については、『ステートメント』ドキュメントを参照してください。

関連トピック：

- 『ステートメント』ドキュメント

特殊な SQL ステートメント

DML ステートメントの他に、Natural には RDBMS との連携専用の特殊な SQL ステートメントセットが用意されています。 SQL ステートメントには、`SELECT`、`INSERT`、`UPDATE`、`DELETE`、`COMMIT`、`ROLLBACK` などがあります。 SQL 固有のステートメントについては、『ステートメント』ドキュメントを参照してください。

特殊な SQL ステートメントのセットは、フレキシブル SQL と、ストアドプロシージャを使用するための機能で構成されます。 フレキシブル SQL を使用すると、任意の SQL 構文を使用できます。

関連トピック：

- *SQL ステートメント*（『ステートメント』ドキュメント）
- *フレキシブル SQL*（『ステートメント』ドキュメント）

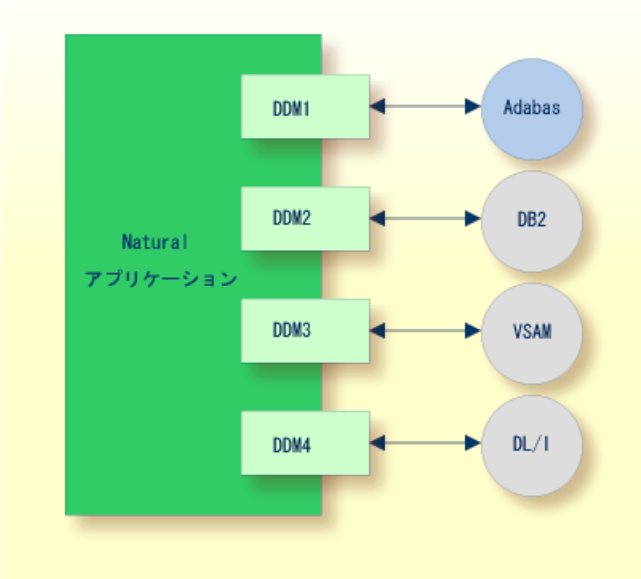
Natural データ定義モジュール

Natural にはデータ定義モジュール（DDM）と呼ばれるオブジェクトが用意されており、これを使用すると各種 DBMS のデータベースファイルに透過的にアクセスできて便利です。

DDM は、物理データベースファイルの論理ビューを構成します。 DDM には、ファイルの各フィールドの情報（Natural オブジェクトでのこれらのフィールドの使用に関連する情報）が保存されます。

DDM は、Natural データ構造と、使用する DBMS のデータ構造との接続を確立します。 このようなデータベース構造としては、RDBMS 内のテーブル、Adabas データベース内のファイル、VSAM データセットなどがあります。 このため、DDM はアクセス対象のデータベースの実際の構造を Natural アプリケーションから隠します。

次の図は、Naturalが、特定のDBMSの特定のデータ構造を表すDDMへの参照を使用することで、単一のアプリケーションから複数のデータベースにアクセスできる仕組みを示しています。

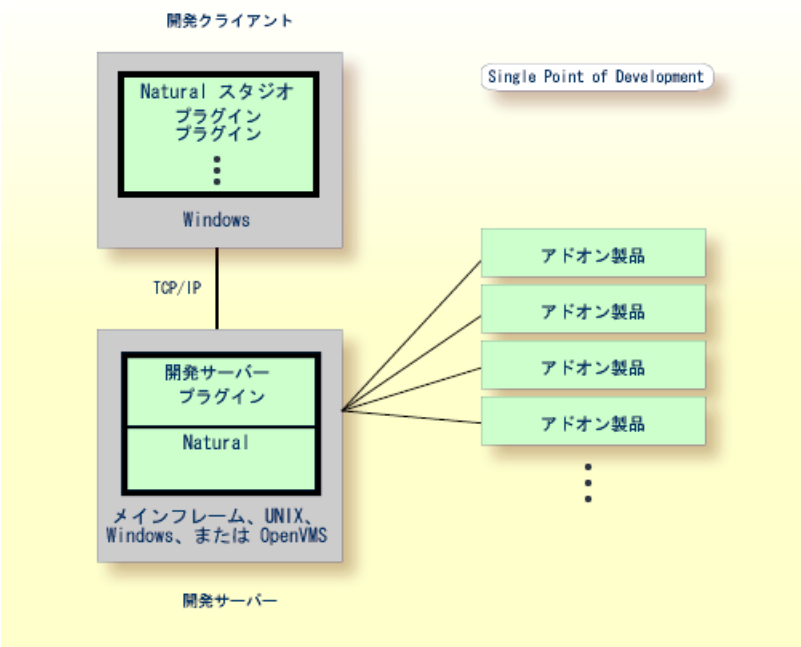


12 Natural SPoD アーキテクチャ

このセクションでは、Natural の Single Point of Development（SPoD）のシステムアーキテクチャについて説明します。

SPoDを使用すると、単一のWindows環境からアプリケーション開発を一元化できます。Naturalスタジオ（Natural for Windows に付属）の機能を使用して、メインフレーム、UNIX、またはOpenVMSプラットフォーム上のリモート環境にあるNaturalアプリケーションを開発およびテストできます。

SPoDはクライアント／サーバー概念をベースとしており、すべてのプラットフォームに対応する単一の開発環境を実現します。次の図は、この概念と、SPoDアーキテクチャの主なコンポーネントを示しています。



開発クライアント

Natural for Windows は、メインフレーム、UNIX、または OpenVMS プラットフォーム上のターゲット環境用のリモート開発デスクトップクライアントとして機能します。 デスクトップクライアントには、Natural スタジオが含まれます。Natural スタジオは、ユーザーがアプリケーションの設計を行う中央ワークステーションです。 リモート開発に必要なすべての Natural 関連機能は、Natural スタジオ内で実行できます。

開発サーバー

Natural 開発サーバープラグインを使用すると、メインフレーム、UNIX、Windows、または OpenVMS プラットフォーム上のターゲット環境の Natural インストールをリモート開発できます。 開発サーバーは、ターゲットプラットフォーム上の Natural と、Natural 開発サーバープラグインで構成されます。

アドオン製品

Natural スタジオでは、1 つ以上の Natural アドオン製品（Predict など）を SPoD 環境に統合するプラグインを使用できます。 Natural スタジオプラグインの前提条件は、開発サーバーに環境に各アドオン製品をインストールすることです。

セキュリティ

Natural Security を使用して、Natural 開発サーバー環境と、Natural ベースのアプリケーションおよび複合アプリケーションを保護できます。 詳細については、『*Natural Security*』ドキュメントを参照してください。

関連トピック：

- 『*Natural Single Point of Development*』ドキュメント