

CA Plex

リリースノート

r7.0



This documentation, which includes embedded help systems and electronically distributed materials, (hereinafter referred to as the "Documentation") is for your informational purposes only and is subject to change or withdrawal by CA at any time.

This Documentation may not be copied, transferred, reproduced, disclosed, modified or duplicated, in whole or in part, without the prior written consent of CA. This Documentation is confidential and proprietary information of CA and may not be disclosed by you or used for any purpose other than as may be permitted in (i) a separate agreement between you and CA governing your use of the CA software to which the Documentation relates; or (ii) a separate confidentiality agreement between you and CA.

Notwithstanding the foregoing, if you are a licensed user of the software product(s) addressed in the Documentation, you may print or otherwise make available a reasonable number of copies of the Documentation for internal use by you and your employees in connection with that software, provided that all CA copyright notices and legends are affixed to each reproduced copy.

The right to print or otherwise make available copies of the Documentation is limited to the period during which the applicable license for such software remains in full force and effect. Should the license terminate for any reason, it is your responsibility to certify in writing to CA that all copies and partial copies of the Documentation have been returned to CA or destroyed.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CA PROVIDES THIS DOCUMENTATION "AS IS" WITHOUT WARRANTY OF ANY KIND, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. IN NO EVENT WILL CA BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE, DIRECT OR INDIRECT, FROM THE USE OF THIS DOCUMENTATION, INCLUDING WITHOUT LIMITATION, LOST PROFITS, LOST INVESTMENT, BUSINESS INTERRUPTION, GOODWILL, OR LOST DATA, EVEN IF CA IS EXPRESSLY ADVISED IN ADVANCE OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE.

The use of any software product referenced in the Documentation is governed by the applicable license agreement and such license agreement is not modified in any way by the terms of this notice.

The manufacturer of this Documentation is CA.

Provided with "Restricted Rights." Use, duplication or disclosure by the United States Government is subject to the restrictions set forth in FAR Sections 12.212, 52.227-14, and 52.227-19(c)(1) - (2) and DFARS Section 252.227-7014(b)(3), as applicable, or their successors.

Copyright © 2012 CA. All rights reserved. All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.

テクニカル サポートへのお問い合わせ

テクニカル サポートへのお問い合わせ

オンライン技術支援、および連絡先の住所、営業時間、電話番号については、テクニカルサポート <http://www.caj.co.jp/support> にお問い合わせください。

ご意見・ご感想

CA 製品ドキュメントについてのご意見またはご質問がございましたら、techpubs@ca.com にご連絡ください。

製品ドキュメントについて簡単なアンケート調査にご協力いただける場合、CA サポート Web サイトでもご利用いただける、次の URL にアクセスし、ご回答ください。
<http://ca.com/docs>

目次

第 1 章: 新機能と更新された機能 9

.NET クライアント アプリケーション ジェネレータ	9
Plex パッケージャ ツール	10
Web サービス インポート	10
WCF プロキシ	11
IBM i Unicode サポート	12
JavaBeans ネイティブの置き換え	13
IBM i ディスパッチャの機能拡張	13
CA Plex r7.0 で削除された機能	13

第 2 章: 旧リリースからのアップグレード 15

アップグレード前に	15
グループ モデルとローカル モデルのアップグレード	15
CA Plex r6.0 と r6.1 からのアップグレード	16
Windows C++ ファンクションのアップグレード要件	16
新しいランタイム .INI の検索パス	16
Java、C#、RPG ファンクションのアップグレード要件	17
.NET サーバ ランタイムのアップグレード考慮事項	17
CA Plex r5.5 からのアップグレード	18
Windows C++ ファンクションのアップグレード要件	18
COM インポートのアップグレード考慮事項	19
Java ファンクションのアップグレード要件	19
Windows アプリケーション サーバの環境設定	22
RPG ファンクションのアップグレード要件	22
デフォルトで切り詰めない長いファイル名	22
CA Plex r5.1 からのアップグレード	23
CA Plex r5.5 での継承の解決方法の変更	23
ケース 1: 複数の継承呼び出しに関する解決方法	23
ケース 2: パラメータの順序に関する変更	27
ケース 3: イベント、サブルーチン、および収集ポイントに関する変更	28
CA Plex r 5.0 からのアップグレード	28
Windows クライアントの MFC ネイティブ コントロール	29
Java ファンクション呼び出し	31
CA Plex r4.5 からのアップグレード	32

継承とプロパティの解決方法の変更	32
継承トリプルの目的語の置き換え	32
複数のレベルで入力された同一プロパティのトリプル	32
RPG 内部ファンクションと「OBASE/現行日付と時刻を設定」のメタ変数	33
Java のメッセージとソース コードでの一重引用符の使用	33
Java の値での円記号の使用	34
CA Plex r4.0 からのアップグレード	34
置き換えとスコープされたオブジェクト	34
CA Plex r3.5 からのアップグレード	34
For Update オプションの動作の変更	34
C++ コードでの CONCAT 演算子の変更	35
System i フィールド値ファイルにアクセスする際のランタイムのレベル チェックの回避	35
ダイナミック アプリケーション パーティショニング API	36
アップグレードしたローカル モデルでのリンカ オプション	36
ランタイム プロパティ DLL のロード	36
ユーザ データ (BSUPPORT) および取引先 (BCONTACT) のパターン	37
CA Plex r3.1 および r3.0 からのアップグレード	37
CA Plex r2.51 あるいはそれ以前のリリースからのアップグレード	37

第 3 章: システム要件 39

PC 開発環境	39
プラットフォーム固有のシステム要件	40
Windows C++ クライアント (WinC ジェネレータ)	40
System i (System i 5250 と System i クライアント/サーバ ジェネレータ)	40
C# サーバ ジェネレータ	41
C# クライアント ジェネレータ	41
Java ジェネレータ	41
EJB オプション	41
Windows C++ サーバ (WinNTC ジェネレータ)	42
配布先のシステム要件	42

第 4 章: インストールについての注意事項 43

Microsoft Help Compiler のインストール	43
---------------------------------------	----

第 5 章: 公開修正モジュール	45
第 6 章: 既知の問題	47
内部テーブルの修正	47
Visual Studio 2005 での C++ 構築パフォーマンス	47
Java クライアント パネルにおける IME Control の制限	47
第 7 章: インターナショナル サポート	49
第 8 章: マニュアル	51

第 1 章: 新機能と更新された機能

本章では、次の CA Plex r7.0 の新機能と更新された機能を説明します。

[.NET クライアント アプリケーション ジェネレータ](#) (9ページ参照)

[Plex パッケージ ツール](#) (10ページ参照)

[Web サービス インポート](#) (10ページ参照)

[WCF プロキシ](#) (11ページ参照)

[IBM i Unicode サポート](#) (12ページ参照)

[JavaBeans ネイティブの置き換え](#) (13ページ参照)

[IBM i ディスパッチの機能拡張](#) (13ページ参照)

[CA Plex r7.0 で削除された機能](#) (13ページ参照)

.NET クライアント アプリケーション ジェネレータ

CA Plex r7.0 の主な機能は、マネージ C# アプリケーションとして Plex UI デザインを生成することです。C# アプリケーションは、Microsoft .NET Framework 4.0 で機能の一つとして提供されている Windows Presentation Foundation (WPF) オペレーティング環境を使用します。CA Plex .NET クライアント ジェネレータは、更新された OBASE と Pattern ライブラリとともに提供され、それらのライブラリは、クライアント サーバ構成の各々をターゲットとする新たに追加されたバリエーションを持ちます。CA Plex .NET ランタイムは、この .NET クライアント ジェネレータをサポートします。

注: 新規開発に対しては、継承元のライブラリ基本セットとして OBASE よりも Patterns を選択することを推奨します。

新しい .NET クライアント ジェネレータについての詳細は、以下を参照してください。

- CA Plex .NET プラットフォーム ガイド
- CA Plex ユーザ ガイド – 「パッケージングとアプリケーションの展開」
- .NET クライアント サンプル モデル

.NET クライアント サンプル モデルは、WinC クライアント/WinNTC サーバ ベース アプリケーションから 100% マネージ コードの .NET アプリケーションへの変換に焦点をあてています。モデルは、¥CA¥Plex¥7.0¥Samples¥Dot Net Client フォルダに用意されています。

- チュートリアル リファレンス モデル

CA Plex チュートリアル リファレンス モデルは、生成構築および実行の準備ができている、すでに移行された状態のモデルです。モデル (TUTREFERCS.mdl) は、¥CA¥Plex¥7.0¥Tutrefer フォルダに用意されています。

Plex パッケージャ ツール

CA Plex は、かなり以前より Java や C# などの最新言語をターゲットとしたアプリケーションを生成できるようになっていますが、これらの言語は、RPG または C++ ベースのジェネレータを使用する多くの Plex 開発者にとってはなじみのない、いくつかの新しい実装概念を伴います。その主な概念的ハードルは、共通の機能をグループ化する名前空間の使用です。

Plex パッケージャ ツールは、Plex 開発者がモデル内でパッケージ情報を自動的に作成するための基準セットを指定することができるようにすることで、そのプロセスを簡素化します。基準セットは、パッケージャ ツールを同じモデルで繰り返し使用できるように保存されます。パッケージ情報は、そのローカル モデルに対して事前に定義された規則をもとに更新することができます。

Plex パッケージャ ツール機能についての詳細は、以下を参照してください。

- CA Plex ユーザ ガイド – 「パッケージングとアプリケーションの展開」
- .NET クライアント サンプル モデル

.NET クライアント サンプル モデルの「Exercise (練習) 2」では、生成と展開のためのアプリケーションの編成を支援するために、パッケージャ ツールを使用します。モデルは、¥CA¥Plex¥7.0¥Samples¥Dot Net Client フォルダに用意されています。

Web サービス インポート

Web サービス インポート機能は、CA Plex ファンクションが標準的な Plex ファンクションとして Web サービスを呼び出すことができるようにすることで、CA Plex でのサービス サポートを完全なものとしします。

Plex Web サービス インポート ツールを使用して、外部のメタ情報をインポートし、Plex ローカル モデル内でサービスを使用することができます。ツールを使用し、パッケージオブジェクトを参照する名前を定義することができます。その名前のもとに、Plex モデル内でサービスが編成されます。

CA Plex は、以前のバージョンよりサーバ ファンクションを EJB または WCF サービスとしてエクスポートする機能を提供しています。

Web サービス インポート機能についての詳細は、以下を参照してください。

- CA Plex ユーザ ガイド – 「コンポーネントのインポートと使用」
- Web サービス インポート サンプル モデル

Web サービス インポート サンプル モデルは、CA Plex アプリケーションで Web サービスをインポートし、使用方法を示します。モデルは、¥CA¥Plex¥7.0¥Samples¥Web Service Import フォルダに用意されています。

WCF プロキシ

CA Plex r6.0 および r6.1 .NET ランタイムは、他のプラットフォームに対する唯一の通信プロトコルとして TCP-IP を使用します。TCP-IP は、イントラネット アプリケーションにとっては適切である一方、Web またはクラウド ベースのサーバアプリケーションデプロイメントには十分ではありません。そのため、CA Plex r6.1 では、Plex が生成したクライアント ファンクションや IIS 内で実行しているサーバ ファンクションを呼び出すことができませんでした。

CA Plex r7.0 では、新たに WCF 呼び出しインターフェースが CA Plex .NET ランタイムに追加されました。CA Plex .NET サーバ ランタイムを適切なホストにデプロイすることによって、IIS のような、以前のバージョンではデプロイできなかった環境にサーバ アプリケーションをデプロイすることができます。WCF 呼び出しインターフェースは、HTTP、TCPIP、名前付パイプのような WCF がサポートする通信プロトコルの使用を可能にします。

新しいプロトコルは、CSHARPWCF という名前で作成され、WCF を使用しなければならないリモート呼び出しを指定することができます。このプロトコルを選択すると、リモート サーバを構成するために必要な追加のパラメータの設定が可能となります。

注: WCF ランタイム プロキシに対してサポートされるオペレーティング環境は、C# クライアントから C# サーバへの呼び出し環境のみです。

WCF プロキシ機能についての詳細は、オンライン ヘルプ「WCF ランタイム プロキシ」セクションを参照してください。

IBM i Unicode サポート

CA Plex は、IBM i オペレーティング環境における文字データの保管や送信のための EBCDIC (Extended Binary-Coded Decimal Interchange Code) をサポートします。しかし、V5R2 から IBM i5 OS は、Unicode をサポートするようになっており、この Unicode サポートによって、複数の言語をサポートしなければならない CA Plex IBM i アプリケーションのデプロイメントを簡素化できます。加えて、Unicode サポートにより、これらのアプリケーションは、より多様なネイティブ言語文字を扱うことができるようになります。

CA Plex は、IBM i プログラムへ接続する次のクライアント ファンクションをサポートします。

- WinC
- Java
- 5250 (グリーン スクリーン)
- .NET

Java と .NET クライアントは、Unicode を使用しますが、CA Plex WinC クライアントは ASCII のみ、5250 は EBCDIC のみをサポートします。CA Plex r7.0 で完全な Unicode クライアント サーバ アプリケーションを実装したい場合は、アプリケーションのクライアントに、Java か .NET のいずれかを選択することを推奨します。

CA Plex r7.0 での Unicode サポートには、以下が含まれます。

- 新しい Unicode データ型でのデータベース スキーマの生成と構築
- RPGIV ジェネレータでの Unicode データ型生成
- RPGIV プログラム (PGM) における Unicode と非 Unicode (EBCDIC) データ型間の相互運用のサポート

IBM i Unicode サポートについての詳細は、オンライン ヘルプ「Unicode サポート」を参照してください。

JavaBeans ネイティブの置き換え

JavaBeans は、多くのサードパーティ ベンダから提供されている再利用可能な GUI ソフトウェア コンポーネントです。CA Plex によって生成される Java クライアントは、JavaBeans を使用します。例えば、CA Plex ACTIVE パターン ライブラリには、CA が提供する JavaBeans セットのいくつかのコンポーネントを実装するパターンが含まれています。

以前のバージョンで提供されていた jSuite コントロールのライセンスは、インフラジスティックス社が Java オペレーティング環境のサポートを中止したことにより、もはや提供されません。そして、インフラジスティックス コントロールによってサポートされていたメソッドは、CA Plex Java ランタイム ネイティブ コントロールに置き換わります。

JavaBeans ネイティブ置き換えコントロールについての詳細は、オンライン ヘルプ「ACTIVE パターン ライブラリでの JavaBeans の使用」セクションを参照してください。

IBM i ディスパッチャの機能拡張

CA Plex IBM i ディスパッチャは、1 クライアントが接続に失敗した場合に、ディスパッチャがハングしてしまうことを防ぐため、クリティカル セクションにソケット タイムアウト パラメータを実装しました。

この IBM i ディスパッチャ機能は、YOBSYTCP と YOBSYTCPDP の両ディスパッチャ機能に実装されます。

IBM i ディスパッチャの機能拡張についての詳細は、オンライン ヘルプ「CA Plex TCP/IP ディスパッチャのタイムアウト」セクションを参照してください。

CA Plex r7.0 で削除された機能

CA Plex r7.0 で削除された機能はありません。

第 2 章: 旧リリースからのアップグレード

本章では、CA Plex の以前のリリースからアップグレードするためのアップグレード要件の概要を説明します。

[アップグレード前に \(15ページ参照\)](#)

[CA Plex r6.0 と r6.1 からのアップグレード \(16ページ参照\)](#)

[CA Plex r5.5 からのアップグレード \(18ページ参照\)](#)

[CA Plex r5.1 からのアップグレード \(23ページ参照\)](#)

[CA Plex r5.0 からのアップグレード \(28ページ参照\)](#)

[CA Plex r4.5 からのアップグレード \(32ページ参照\)](#)

[CA Plex r4.0 からのアップグレード \(34ページ参照\)](#)

[CA Plex r3.5 からのアップグレード \(34ページ参照\)](#)

[CA Plex r3.1 および r3.0 からのアップグレード \(37ページ参照\)](#)

[CA Plex r2.51 あるいはそれ以前のリリースからのアップグレード \(37ページ参照\)](#)

アップグレード前に

アップグレード前に、追加の情報または本書が記載された後に提供された修正のために CA テクニカル サポート (<http://www.ca.com/jp/support/>) の CA Plex ホーム ページを確認してください。

CA Plex の複数のリリースおよびその生成されたアプリケーションは、同じマシン上に導入することができます。例えば、CA Plex r6.1 と CA Plex r7.0 は同じマシン上に導入し、実行することができます。

グループ モデルとローカル モデルのアップグレード

定期的にローカル モデルからグループ モデルへ更新することが求められます。それゆえ、アップグレードを開始する前にグループ モデルを更新し、オフラインですべてのグループ モデルとローカル モデルのバックアップを取り、新しいリリースへグループ モデルをアップグレードした後に新規のローカル モデル (そして、ビルド ファイル) を作成することをお奨めします。

すべてのグループ モデルと関連するライブラリ モデルは、同時にアップグレードされなければなりません。

グループ モデルへログインすることで CA Plex r7.0 にアップグレードされます。アップグレード後は、以前のリリースでそのモデルにアクセスすることはできなくなります。そのため、作業グループのすべての開発者が同時に CA Plex の新しいリリースへアップグレードする必要があります。2 人の開発者が、CA Plex の異なるリリースを使用している場合、その 2 人は同じモデルで作業することはできません。

ローカル モデル アップグレード

最新リリースでローカル モデルを開くことで簡単に以前のリリースからローカル モデルをアップグレードすることができます。この方法は、テスト目的のために役に立ちます。しかし、上記で述べているように正式なリリース アップグレードのためのアプローチとしては推奨されません。

CA Plex r6.0 と r6.1 からのアップグレード

このセクションでは、r6.0 と r6.1 からのアップグレードに関連するアップグレード要件について説明します。

Windows C++ ファンクションのアップグレード要件

CA Plex r6.0 または r6.1 から CA Plex r7.0 へアップグレードする際、既存の C++ ファンクションを再生成構築する必要はありません。

新しいランタイム .INI の検索パス

Windows Vista セキュリティ要件やマルチユーザ展開のサポート向上のため、位置付けおよびランタイム .INI ファイルの作成に WinC ランタイムが使用するアルゴリズムが変更されました。

これらの変更を注意して確認してください。特に、カスタマイズされた .INI ファイル処理を開発している、または、サーバベースのマルチユーザ環境に CA Plex アプリケーションを展開する場合は注意が必要です。

典型例

典型的なエンド ユーザ環境において、.INI ファイルを含む開発したアプリケーションは、**Program Files** フォルダに導入されます。**Program Files** フォルダ内のファイルは、標準ユーザに対しては読み取り専用です。

Plex WinC アプリケーションを開始する際、アプリケーションは、書き込み可能な .INI ファイルを検索します。ファイルが見つからない場合、新しい .INI ファイルがエンドユーザの個人フォルダ (Windows XP では「マイ ドキュメント」、Windows Vista では「ドキュメント」) に作成されます。.INI ファイルは、実行ファイルと同じ名前でサブフォルダ内に置かれます。例えば、MyApp.exe であれば .INI ファイルは「マイ ドキュメント」¥Myapp¥Myapp.ini に置かれます。

それゆえ、この例ではアプリケーションを実行する各標準ユーザは、ユーザ独自のアプリケーション .INI ファイルのコピーを持つことになります。

.INI ファイル アルゴリズム

一連のルールは以下のように設計されており、アプリケーションを実行するユーザのアクセス権限に応じた他の展開シナリオに対する柔軟性を提供します。

もし、Plex が生成した実行ファイル名が **app.exe** であれば、各 **.INI** ファイルは **app.ini** になります。

実行時、**app.exe** アプリケーションは、次のように **app.ini** ファイルを検索します。

1. 実行ファイル (**app.exe**) が位置する同じディレクトリ内。 **app.ini** ファイルがそこに
見つかり、ファイルが読み書き (RW) 権限を持っていた場合は、**app.exe** 実行ファイルはそれを使用します。
2. 「1」で見つからない場合、**app.exe** はローカル ユーザの「ドキュメント」¥app¥ ディレクトリ (**app** は実行ファイル名) を検索します。 **app.ini** ファイルがそこに
見つかり、ファイルが読み書き (RW) 権限を持っていた場合は、**app.exe** 実行ファイルは、それを使用します。
3. 「1」、「2」で見つからない場合、実行ファイルは読み書き (RW) 権限のある **app.ini** を探すために **PATH** ステートメントを使用します。
4. 読み書き (RW) 権限のある **app.ini** ファイルがそれでも見つからない場合、**app.exe** 実行ファイルは、ローカル ユーザの「ドキュメント」¥app ディレクトリ内に読み書き (RW) 権限を持つ新しい **app.ini** ファイルを作成します。この新しい **app.ini** は、最初に見つかった読み取り専用 **app.ini** ファイルのコピーになります。見つからなかった場合は、白紙の **app.ini** が作成されます。

Java、C#、RPG ファンクションのアップグレード要件

CA Plex r7.0 へアップグレードする際、既存の Java、C#、RPG ファンクションを再生成構築する必要はありません。

.NET サーバ ランタイムのアップグレード考慮事項

次のセクションでは、.NET サーバ ランタイムのアップグレード考慮事項を説明します。

Version プロパティ

固有の技術的理由がない限り、.NET リスナの Version プロパティは空白のままにしてください。「600」または「610」の値を指定している場合、削除する必要があります。

CA Plex r7.0 では、このプロパティが空の場合、Plex .NET ランタイムからデフォルトのバージョン番号が取得されます。

Version プロパティは下位互換性の理由から提供されています。例えば、バージョン r7.0 の .NET ランタイムに接続するために r6.0 の WinC クライアントを使用するには、このプロパティを「600」にセットしなければなりません。

CA Plex r5.5 からのアップグレード

このセクションでは、r5.5 (r5.5 SP1 を含む) からのアップグレードに関連するアップグレード要件について説明します。

Windows C++ ファンクションのアップグレード要件

r6.0 より前にリリースされた CA Plex で作成した Windows ファンクション (WinC と WinNTC) は、CA Plex r7.0 C++ ランタイム システムと互換性を持たせるために再コンパイルする必要があります。

1 つの Windows アプリケーションで、(6.0、6.1、7.0 の場合を除き) 異なるリリースの CA Plex で作成された複数の DLL を使用することはできません。CA Plex では、生成された DLL が別の DLL を呼び出す際には、必ずランタイムのバージョン チェックが行われます。互換性のないリリースで作成した DLL 間で呼び出しを行おうとすると、実行時エラーが発生します。CA Plex r5.5 以前のリリースから CA Plex r7.0 へ移行する際には、開発環境でアプリケーションのすべての DLL を構築し直す必要があります。

ランタイム アプリケーションの詳細については、オンライン ヘルプの「異なるバージョンの CA Plex で作成したアプリケーションの実行」を参照してください。

CA Plex r7.0 の新機能については、本書の「新機能」の章を参照してください。

COM インポートのアップグレード考慮事項

COM コンポーネント インポート機能を使用している場合は、そのリリースでの COM インポートおよびラップ生成プロセスに対する修正および改善が行なわれているため、アップグレード プロセスとして COM パッケージを再インポートおよび再生成することを推奨します。コンパイルを行う前にパッケージのラップ属性を見直す必要がある場合があります。次の点に注意してください。

- COM コンポーネント インポート ウィザードは、「既存パッケージを上書きしない」オプションを提供しますが、以前インポートした COM パッケージをアップグレードする場合は、このオプションを外すことは適切ではありません。
- 代替の方法として、COM コンポーネント インポートを実行する前に COM パッケージを削除する方法がありますが、これは COM パッケージを参照する既存のアクション ダイアグラム ステートメントを無効にします。

これらの制限を避けるために、次のアップグレード手順をお奨めします。

1. 新規のグループ モデルを作成し、COMPONENT ライブラリを追加します。
2. 新規のローカル モデルに抽出し、アップグレードする必要があるコンポーネントのための新規の COM パッケージをインポートするために COM コンポーネント ウィザードを使用します。
3. オブジェクト ブラウザで COM パッケージを右クリックし、[ツール] メニューから [XML エクスポート] を選択して、任意の名前の XML ファイルへパッケージをエクスポートします。
4. アップグレードする必要がある COM パッケージを含むローカル モデルを開きます。
5. [ツール] メニューから [インポート] - [XML インポート] を選択、[クリア] オプションを選択し、先に作成した XML ファイルをインポートします。

上記の手順は、既存のアクション ダイアグラム コードを保持しながら COM パッケージを更新します。

Java ファンクションのアップグレード要件

このセクションでは、Java ファンクションのアップグレード要件について説明します。

CA Plex r5.5 SP1 からのアップグレード

r5.5 SP1 (Build 5.5.93) からアップグレードする場合、既存の Java ファンクションを再生成する必要はありません。

CA Plex r7.0 Java ランタイム (obrun.jar) は、旧リリースとの下位互換性があります。r5.5 またはそれより前のリリースで作成されたファンクションでも、新しいランタイムを使用できます。CA Plex r7.0 製品にフル アップグレードしていなくても、修正や改善が加えられた新しいランタイムを活用できます。

CA Plex r5.5 からのアップグレード

CA Plex r7.0 で 1 つでも Java ファンクション再生成した場合は、そのファンクションと呼び出し関係にある他の 5.5 で生成されたファンクションもすべて再生成する必要があります。これは、CA Plex r5.5 SP1 以降でパラメータ形式が変更されたためです。CA Plex r5.5 (Build 5.5.63) または、それより前のリリースからアップグレードする場合は、すべての Java ファンクションを再生成および再コンパイルすることをお奨めします。それによって、生成されるクラスを減らすことができ、また、新しい形式で生成される Java コードとの互換性を保つためにはどのソース コード オブジェクトを変更する必要があるかを特定することができます。

注: Java の再生成および再構築に関するこの要件は、CA Plex r5.5 もしくは、それより前のリリースからアップグレードする場合のみ適用されます。Java ジェネレータの将来のリリースでは、この要件はなくなる見込みです。r5.5 SP1 からアップグレードする場合は、このような要件は存在しません。

CA Plex r7.0 Java ランタイム (obrun.jar) を CA Plex r5.5 Java アプリケーションで使用するには、obxxx.properties ファイルに次の設定が必要です。

Version=700

SPVersion=0

Java と C# Exec SQL アクション ダイアグラム ソース コード

CA Plex 6.0、6.1、7.0 において Java と C# ジェネレータは、デフォルトですべての Exec SQL ステートメントを Prepared ステートメントとして生成します。Prepared ステートメントで生成されてはいけなソース コード オブジェクトを示すために SRC SQL statement type SYS トリプルを使用することができます。

Prepared ステートメントの生成は、同じ SQL ステートメントが繰り返し実行される場合にパフォーマンスおよびスケーラビリティを改善かもしれません。しかし、その結果はアプリケーションや DBMS 固有要素に依存します。CA は、その都度最適な設定を決定するために SRC SQL statement type SYS の異なる値でのベンチマークを推奨します。

また、Prepared ステートメントの生成は Exec SQL ステートメントへのすべてのパラメータがテーブル内のカラムと一致することを前提としています。Exec SQL ステートメントは、様々な SQL コードを実装するために使用することができるため、この前提は常に真ではありません。Prepared ステートメントで生成すべきではないソース コード オブジェクトを示すために SRC SQL statement type SYS トリプルを使用することができます。r5.5 以前のリリースから Java アプリケーションをアップグレードする際には、エラーを避けるためにいくつかのソース コード オブジェクトに対してそのようなトリプルを追加する必要がある場合があります。

この変更は、C++ での Exec SQL 使用には影響しません。

Java と Oracle varchar 互換 サポート

Oracle データベースを対象とし、任意の **varchar** フィールドを使用する **Java** アプリケーションは、**r5.5** 以前のリリースからアップグレードした後に振舞いが異なる場合があります。**Plex** での **varchar** のモデル化の仕方により、アップグレード後に楽観的ロック エラーが起こる場合があります。例えば、

「インスタンスは他のユーザにより変更されました。」

標準的な解決は、対象の値が空の **varchar** フィールドに **FLD null VAL** トリプルを追加することです。詳細については、**CA Support Online Web** サイトで **Problem CPLEX 1296** を参照してください。

CA Plex r5.5 からアップグレードする場合の Java ソース コード

モデル内のソース コード オブジェクトにユーザが入力した **Java** ソース コードには、変更が必要な場合があります。ジェネレータによって作成される **Java** クラスの構造が、**CA Plex r5.5 SP1** で変更されたためです。**CA** のパターン ライブラリ (**JAVA API** ライブラリ など) に含まれているソース コード オブジェクトは、すでに変更済みです。**CA Plex r7.0** で再生成およびコンパイルを実行する前に、製品付属の **JAVA API** ライブラリから抽出し直してください。

Java ソース コードに問題があると、コンパイル時に「**unexpected type**」のエラーが発生します。例を次に示します。

C:\GENJAVA\MyFunction_ObFnc.java:行番号: unexpected type

required: variable

found : value

このような問題を解決するには、「**=**」演算子の代わりに **assign** メソッドを使用します。

r5.5 SP1 より前のリリースでは、メソッドの戻り値をソース コード **API** の出力パラメータに割り当てるために、演算子「**=**」を使用できました。例を次に示します。

```
&(4:) = anyMethod(&(1:), &(2:), &(3:));
```

r5.5 SP1 からは、「**=**」演算子の代わりに **assign** メソッドを使用するよう、上記のコード例を次のように変更する必要があります。

```
&(4:).assign(anyMethod(&(1:), &(2:), &(3:)));
```

Windows アプリケーション サーバの環境設定

製品名が AllFusion から CA に変更されたため、CA Plex Windows アプリケーション サーバの環境設定用のレジストリ キーが変更されました。

既存の環境設定は、CA Plex r7.0 のインストール プログラムによって自動的に、新しいキー値にコピーされます。しかし、コピーされた設定が CA Plex Windows アプリケーション環境マネージャで表示されない場合は、AppServer¥Bin フォルダ内にある migration2.exe プログラムを実行してください。

注: この移行プログラムの実行には、パラメータは不要です。

RPG ファンクションのアップグレード要件

CA Plex r7.0 へアップグレードする際、既存の RPG ファンクションの再生成構築は必要ありません。

ビルド ファイル (.BLD) の互換性

上記で述べられているように、新規のローカル モデルを作成することが推奨されます。そのため、アップグレード プロセスの一部としてビルド ファイルに対して作業を行う必要があります。CA Plex r7.0 で古い BLD ファイルを使用した場合、生成と構築ウィンドウを開く度に CA Plex は、エラー メッセージ (E-BLD-1777) を送信します。古い BLD ファイルを使用するには、CA Plex r7.0 と互換性を保つように、生成と構築オプションを見直す必要があります。

6.0 より前のビルド ファイルでは、生成と構築オプションの【構築ディレクトリ】セクションに System i の代わりに AS/400 製品名が使用され、表示されます。

表示されるエラー メッセージを避けるには、次のエントリを BLD ファイルへ追加します。

```
[Options]
Release=700
```

デフォルトで切り詰めない長いファイル名

【名前割当て】セクションの【生成時にロング ファイル名が桁落ち】オプションは、デフォルトでは選択されなくなりました。これは、このオプションを明示的に選択しない限り 8 文字より長いファイル名は切り詰められないことを意味します。前のリリースで生成時に切り詰められていた名前をモデルが持っている場合、アップグレードによる影響がある可能性があります。

CA Plex r5.1 からのアップグレード

次のセクションでは、**r5.1** からのアップグレードに関連するアップグレード要件について説明します。

CA Plex r5.5 での継承の解決方法の変更

CA Plex r5.0 では、継承エンジンが変更され、継承動作における制限とバグの問題が解決されました。また、**r5.0** で新しい従属関係ブラウザをサポートするための変更が加えられました。これらの変更により、新しいリリースへアップグレードする際に重大な問題が発生することがありました。**CA Plex r5.1** で、これらの問題が修正されましたが、いくつかの重大な問題は残ったままでした。

CA Plex r5.5 では、継承エンジンの既知のバグが修正されています。**r5.5** と **r5.5 SP1** の間では、継承エンジンはまったく変更されていません。**r5.5** より前のリリースでは継承エンジンの動作が異なるため、一部のユーザーに影響する可能性があります。次のセクションでは、3 通りのケースを紹介します。

ケース 1: 複数の継承呼び出しに関する解決方法

継承されたファンクション呼び出しが、アクション ダイアグラムで異なって解釈され、生成コードに別の呼び出しが作成されることがあります。典型的なモデルでは、このような影響を受けるファンクションは、あったとしてもわずかです。これらのケースは、生成されたアプリケーションのテスト、生成されたソースの比較、またはアクション ダイアグラム コードの比較によって特定できます。この問題の検出に関してサポートが必要な場合は、**CA テクニカル サポート**にご連絡ください。

この問題を解決するには、適切な置き換えトリプルを追加し、必要な動作に合ったファンクションが返るようにします。できれば、このようなトリプルを「標準レイヤ」のモデルに入力し、必要な変更を最小に抑えることをお勧めします。

背景

CA Plex r3.5 では、FNC is a FNC 動詞の動作が変更され、複数のファンクション継承がサポートされるようになりました。その結果、CA Plex の既存のファンクションも、トリプルを追加入力せずに、複数のファンクションから継承できるようになりました。また、[オブジェクト プロパティ] で、継承された複数の **Calls** トリプルを元の同じアクション ダイアグラム呼び出しに対応させて表示できるようになりました。各継承アクション ダイアグラムで実行できる呼び出しは **1** つだけなので、CA Plex では、ロード時に、使用可能なファンクションの中から実際に呼び出すファンクションが決定されます。

r5.0 より前のものでは、このようなアクション ダイアグラム呼び出しが明示的に処理されませんでした。そのため、この呼び出しに関する問題の解決方法は、通常の継承エンジンの規則に準拠していないことがありました。r5.0 あるいはそれ以降では、このような呼び出しを処理するための機構が追加されました。この機構は、何度も修正されています。最新の修正版は、r5.5 に組み込まれています。

例 1: フィルタ パターン

継承された複数のアクション ダイアグラム呼び出しは、Filter.FilteredGrid ファンクションなどで実行されます。次のトリプルを例として説明します。

```
EditDetailFilter is a ENT STORAGE/RelationalTable
EditDetailFilter is a ENT FOUNDATI/EditDetail
EditDetailFilter is a ENT FOUNDATI/Filter
EditDetailFilter.Edit replaces FNC UIBASIC/Grid
...by FNC EditDetailFilter.FilteredGrid
```

EditDetailFilter.Edit 呼び出しに注目すると、2 種類の BlockFetch ファンクションが呼び出されていることがわかります。



動詞	ターゲット	実	Vis	Via	From
calls	EditDetailFilter.Fetch.BlockFetch	☐	☒		EditDetail.Edit calls EditDetail.Fetch.BlockFetch
calls	EditDetailFilter.Filter.BlockFetch	☐	☒	EditDetailFilter.Edit is a EditDetailFilter.FilteredGrid	EditDetailFilter.FilteredGrid calls EditDetailFilter.Filter.BlockFetch

どちらの呼び出しも表示されることに注意してください（[Vis] カラムにチェックが設定されているため）。r5.0 と r5.1 でも同様の動作です。では、アクション ダイアグラムで実際に呼び出される **BlockFetch** はどちらなのか。その決定は、アクション ダイアグラムのロード時に行われます。結果は、CA Plex 最新リリースでの継承エンジンに加えられた変更に影響されます。意図としては **Filter.BlockFetch** が呼び出されるべきです。CA では、r5.5 でこれを実現するために、FOUNDATION パターン ライブラリに置き換えトリプルを追加しました。

FOUNDATI/Filter.FilteredGrid **replaces FNC** UIBASIC/UIBasic.Grid.BlockFetch
...by FNC FOUNDATI/Filter.Filter.BlockFetch

以前は、**BlockFetch** ファンクションを明示的置き換えることができず、代わりに、スコープ ビューだけで置き換えていました。

FOUNDATI/Filter.FilteredGrid **replaces VW** UIBASIC/UIBasic.Grid
...by VW FOUNDATI/Filter.Filter

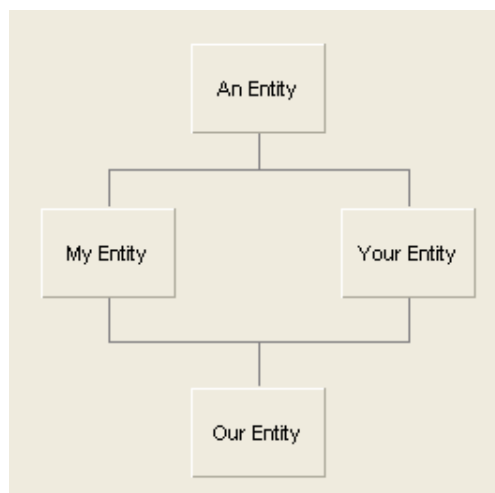
この方法は、修正された CA Plex r5.5 のアルゴリズムで要求されたファンクションの置き換えを起こさせるには不十分です。そこで、CA では **BlockFetch** の明示的な置き換えを追加することにより、r5.5 でその要求された動作を維持できるようにしました。このような振舞いを見せるユーザ独自のファンクションに対して同様の操作が必要になる可能性があります。

例 2: 「ダイヤモンド」継承

次の一連のトリプルを確認してください。

Our Entity.Our Function		An Entity.A Function
Function	<All>	Function
An Entity	function	A Function
A Function	calls	An Entity.A Validation Function
	function	A Validation Function
My Entity	is a	An Entity
	function	My Validation Function
My Function		My Function
	is a	My Entity.A Function
	replaces	My Entity.A Validation Function
	...by	My Entity.My Validation Function
Our Entity	is a	My Entity
	is a	Your Entity
Your Entity	is a	An Entity
	function	Your Validation Function
		Your Function
Your Function	is a	Your Entity.A Function
	replaces	Your Entity.A Validation Function
	...by	Your Entity>Your Validation Function
	function	Our Function
Our Function	is a	Our Entity.My Function
		Our Entity>Your Function
	replaces	Our Entity.My Validation Function
	...by	Our Entity.Our Validation Function
	function	Our Validation Function

このタイプのシナリオは、継承結果が次の図のようなダイヤモンド型の階層になっているため、「ダイヤモンド継承」と呼ばれています。



この継承階層は、**An Entity** で始まり、2つの分岐が **Our Entity** で1つに戻ります。この状況はかなり複雑ですが、実際のユーザモデルでは更なる分岐と関係する継承階層を持つ、より複雑なものがしばしば見受けられます。

Our Entity.Our Function の [オブジェクトプロパティ] を表示すると、2つの **Calls** トリプルが表示されているのを確認できます。注意すべき次の2つのポイントがあります。

- この一連のファンクション内で明示的に入力されたアクション ダイアグラム **Call** は1つだけですが、[オブジェクトプロパティ] では2つの呼び出し候補が表示されます。
- このシナリオは、3.5以降のすべての **CA Plex** リリースで再現させることができます。

オブジェクトプロパティ - Function: Our Entity.Our Function			
動詞	ターゲット	実	Vis
calls	Our Entity.Our Validation Function	☐	☒
replaces	Our Entity.A Validation Function	☐	☒
...by	Our Entity.Our Validation Function	☐	☒
is a	Our Entity.My Function	☒	☒
calls	Our Entity.Your Validation Function	☐	☒
is a	Our Entity.A Function	☐	☒
replaces	Our Entity.Our Validation Function	☐	☒
...by	Our Entity.Your Validation Function	☐	☒
is a	Our Entity.Your Function	☒	☒
replaces	Our Entity.My Validation Function	☒	☒
...by	Our Entity.Our Validation Function	☒	☒

実際に呼び出されるファンクションを判断する唯一の方法は、アクション ダイアグラムコードを検証することです。また、この例は最近のリリースにおける継承エンジンの変更を示します。r5.0 では、**Our Entity.Your Validation Function** が実際に呼び出されますが、r5.5 を含め他のリリースでは、**Our Entity.Our Validation Function** が呼び出されます。

ケース 2: パラメータの順序に関する変更

生成されたファンクション呼び出しでの継承パラメータの順序が、以前のリリースと異なることがあります。通常、関連するすべてのファンクションは、新しいリリースへのアップグレード過程で再生成されるので、この変更による悪影響はありません。ただし、この変更が重大な影響を与えるケースがあります。

- 既存の RPG および Java ファンクションで下位互換性を確保する場合は、追加のファンクションを再生成する必要があります。
- CA Plex で生成されたプログラムと統合する手組みのプログラムが存在する場合は、修正されたパラメータ インターフェースを考慮し、手組みで作成したプログラムを修正する必要がある可能性があります。

これらの問題は、生成されたアプリケーションのテスト、生成されたソースの比較、またはアクション ダイアグラム コードの比較によって特定できます。この問題の検出に関してサポートが必要な場合は、CA テクニカル サポートにご連絡ください。

ケース 3: イベント、サブルーチン、および収集ポイントに関する変更

継承されたイベント構造体とサブルーチンの順序が、以前のリリースと異なることがあります。サブルーチンの順序の変更が、生成アプリケーションに悪影響を与えることはありません。通常は、イベント構造体も同様です。ただし、複数のイベント構造体で同じ論理イベントを参照している場合、または、非修飾イベント構造体を使用している場合は、イベントの順序が問題となることがあります。例を次に示します。

Events Handler

Event

```
//全てのイベントからトリガされる条件を持たないイベント
```

```
Go Sub 一般的なイベント処理
```

Event Delete

```
Go Sub 削除処理
```

Event Delete

```
Go Sub 更なる削除処理
```

このイベント構造体の順序が異なると、生成されたアプリケーションの動作も異なります。実際には、この機能が利用されることは稀です。また、継承エンジンへの変更によって順序が大きく異なることは、より稀になります。他のケースと同様に、これらの問題は生成されたアプリケーションのテスト、生成されたソースの比較、またはアクション ダイアグラム コードの比較によって、特定できます。

基本的に、収集ポイント（前後ポイント）内の継承されたコードブロックの順序についても同様の注意事項が適用されます。CA のテストでは、このような結果を招く例は検出されませんでした。この問題が発生する可能性があることに注意してください。

CA Plex r5.0 からのアップグレード

r5.0 からのアップグレード際には、これまでの各セクションで説明したアップグレード要件に加えて、次のセクションを確認してください。

Windows クライアントの MFC ネイティブ コントロール

CA Plex r5.0 以降では、Windows クライアントで 2 つの GUI コントロール セットをサポートします。WinC アプリケーションのデフォルト設定では、グリッド以外のすべてのタイプのコントロールに MFC コントロールが使用されます。他方の GUI コントロール セットは「Winwidgets」と呼ばれ、r5.0 より前のリリースでは、このコントロール セットがデフォルトでした。

どちらのコントロール セットにも、長所と短所があります。Winwidgets コントロールは、CA Plex 1.0 以降のすべてのリリースで使用されています。つまり、それらは多くの CA Plex ユーザに長年にわたって正常に実装されている成熟した技術と言えます。CA Plex の以前のリリースからアップグレードする場合、下位互換性の問題を最小限に抑えられる可能性が高いことから Winwidgets コントロールを使用する方が、多くの場合、より簡単になります。

ただし、MFC コントロールにも次のような多くの優位性があります。

- サードパーティ製のテスト ツールとの互換性
- Windows 標準のルック アンド フィール
- 低水準言語制御のための MFC API へのアクセス

下位互換性オプション

いくつかの例外（次のセクションを参照）を除き、**Winwidgets** コントロールで利用可能な機能はすべて **MFC** コントロールでも利用できます。しかし、予期されないアップグレードの問題に対する防護措置として **Winwidgets** コントロールに戻ることができるランタイム オプションを利用することができます。

コントロールのタイプ毎、別々にオプションを利用できます。

```
[NativeControls]
;GUI controls either display as MFC controls (1) or Winwidgets (0).
;These settings have no effect on the WinC grid which is always Winwidgets
ListBox=1
SpinButton=1
SingleLineEdit=1
SingleLineEditNumeric=1
MultiLineEdit=1
ComboBox=1
RadioButton=1
CheckBox=1
PushButton=1
Statics=1
```

該当するコントロールを **Winwidgets** に戻したい場合には、必要なオプションを「0」に設定してください。**MultiLineEdit** の設定は例外です—**Winwidgets** は複数行編集コントロールを提供しないため、この値を「0」に設定することは、ネイティブ コントロールの初期の実装が有効になることを意味します。

注: 記載されていないアップグレード上の問題があれば、CA テクニカル サポートまでご連絡ください。

MFC コントロールでのアップグレード時の問題

アップグレードに関する既知の問題を以下に記載しています。記載されていないアップグレード上の問題があれば、CA テクニカル サポートまでご連絡ください。

- **Disabled のテキスト色** – MFC コントロールが使用不可 (Disabled) の時、テキストが Windows の標準色（多くの場合、淡いグレー）で表示されます。Text Color プロパティは、コントロールが使用不可 (Disabled) になると無視されます。編集コントロールについては、次のようなプロパティの組み合わせで、Text Color プロパティをサポートする読み取り専用の編集コントロールを定義できます。

Mode=Read-only

および

Disabled=No

- **コンボ ボックス サイズ** – 他のコントロール タイプとは異なり、コンボ ボックスの編集コントロール部分のサイズは、直接変更することはできません。このサイズは、コントロール内部のテキストのフォント サイズによって決まります。設計時に編集部分のサイズを変更することはできますが、実行時にその変更内容は無視されます。ただし、設計時に編集部分のサイズを変更した場合には、実行時にドロップダウン リストのサイズが変化します。
- **透過性** – MFC コントロールでは、透過性は完全にはサポートされていません。したがって、コントロールが重なる場合にはパネルの外観が変わることがあります。
- **Z オーダ** – MFC コントロールの Z オーダは、Winwidgets コントロールのデフォルトの Z オーダとは逆順になっています。このため、重なっているコントロールに依存するパネルの設計に影響する場合があります。

注: 設計時のコントロールの Z オーダ順序を実行時とできるだけ一致させるには、該当する各コントロールの Clip Control プロパティを Yes に設定してください。フレーム内のフレームなどの重なるコントロールを使用して作業する場合に役に立ちます。

Java ファンクション呼び出し

CA Plex r4.5 では、コマンド ラインまたは独自にコーディングした Java から Java のファンクションを呼び出す場合には、ファンクション名に接頭語としてパッケージを付ける必要はありませんでした (package.func)。

CA Plex r5.0 Service Pack 1 およびそれ以降では、呼び出し内にパッケージ指定されなかった場合に検索されるパッケージ リストを提供するために使用できる PackageList エントリが .properties ファイルにあります。通常、ファンクションのパッケージは各ファンクション呼び出しのコード内に直接生成されます。しかし、ダイナミック パーティショニング シナリオでは、これが当てはまらないことがあります。例えば、生成時のファンクション呼び出しのターゲットを Windows のファンクションにすることができます（この場合、生成される呼び出しにパッケージは組み込まれません）。実行時に目的のファンクションが Java に切り替わった場合に、そのファンクションを見つけるために PackageList が使用される場合があります。

CA Plex r4.5 からのアップグレード

このセクションは、**r4.5**に関連したアップグレード要件について説明します。

継承とプロパティの解決方法の変更

CA Plex r5.0 あるいはそれ以降には、特定の状況において継承オブジェクトのプロパティが変わる可能性のある継承エンジンの拡張機能が含まれています。

継承トリプルの目的語の置き換え

次の例を確認してください。

A is a B

B is a C

A replaces C by D

「**B is a C**」が **A** 到る時には、**A** は (**B** が継承する) **C** を継承せず、**D** から継承するようになります。

以前のリリースでは、**B** からの継承と競合する **D** から継承されたものは、**B** から継承したバージョンに従っていました。**r5.0** では、競合する **D** から継承されたものは、(**C** の継承により生じた) **B** から継承されるものよりも優先します。言い換えれば、「**A is a D**」が、「**A is a B**」と「**B is a C**」よりも優先するということです。この結果は、「後」のトリプルが「先」のトリプルよりも優先する継承の原則と矛盾がありません。

複数のレベルで入力された同一プロパティのトリプル

トリプルを入力、前のレベルに構成を変更、そして、同じプロパティを入力することにより **1** つのオブジェクトに複数の同じトリプル定義することが可能です。例えば、

A type Character (Level 3)

A type Numeric (Level 1)

r5.0 より前では、後のレベル（この例では **Level 3**）に設定した構成の場合、両方のトリプルがモデル エディタに表示され、継承エンジンに送られていました。モデル エディタに内に表示された順番で最後のトリプルが優先されます。つまり、この例では **A** は **Numeric** (数値) となります。

CA Plex **r5.0** では、「後のものが先のものを隠す」原則がレベルに適用されます。「**A type Character**」のみがモデル エディタに表示され、継承のために送られます。

RPG 内部ファンクションと「OBASE/現行日付と時刻を設定」のメタ変数

r5.0 では、RPG 内部ファンクションが独自のメタ変数空間を持つように修正され、呼び出しているファンクションのメタ変数空間は共有されなくなりました。この変更は、メタ変数のスコープに関連する文書化された機能性（メタ変数の状態は、タイプ **Meta** のファンクションへの呼び出しでのみ維持される）に一致し、他のジェネレータとの整合性を持ちます。

この修正によって、外部ファンクションから内部ファンクションへメタ変数の状態情報が渡されることを想定しているクラス ライブラリ内の箇所が浮き彫りになりました。

OBASE のバリエーション AS400 5250 および Windows/AS400 において、それは「OBASE/現行日付と時刻を設定」ファンクションを呼んでいました。

注：「OBASE/現行日付と時刻を設定」は、RPG400 で使用される新規のファンクションです。OBASE を Windows/AS400 バリエーションに設定し、OBDATE を「Windows クライアント」バリエーションに設定している構成では、「OBASE/現行日付と時刻を設定」ファンクションの代わりに、このファンクションの使用をお奨めします。このよい例としては、「OBASE/監査用エンティティ.監査フィールド」ファンクションでの呼び出しの変更があります。

この変更に伴うアップグレードの問題への対応を支援するために CA Plex r5.1 以降では、次のような Plex.ini ファイル オプションを使用することができます。

[RPG Generator]

Share Meta-Variables With Internal Functions=1

「1」を設定することで、r5.0 より前の動作に戻ります。エントリが存在しない場合には、デフォルト設定の「0」が使用され、r5.0 と同じ動作になります。

Java のメッセージとソース コードでの一重引用符の使用

CA Plex r4.5 およびそれ以前の Java ジェネレータでは、実行時に 1 つの一重引用符を使用する必要がある場合には、メッセージおよびソース コード内で必ず 2 つの一重引用符 (') を指定する必要がありました。これは、たとえば Exec SQL ステートメントで使用するためにパラメータをソース コード内に埋め込む場合に必要でした。この動作は、他のジェネレータと矛盾していました。

この問題は CA Plex r5.0 で修正されました。1 つの一重引用符 (') で指定できるようになりましたが、以前の回避策を使用している既存のメッセージやソース コードがあれば、r7.0 にアップグレードする際に編集が必要となります。

Java の値での円記号の使用

CA Plex r4.5 およびそれ以前のバージョンの Java ジェネレータでは、実行時に円記号 1 つの円記号を使用する必要がある場合は、必ず値の中で 2 つの円記号 (¥¥) を指定する必要があります。この動作は、他のジェネレータと矛盾していました。

この問題は CA Plex r5.0 で修正されました。1 つの円記号 (¥) で指定できるようになりましたが、以前の回避策を使用している既存の値があれば、r7.0 にアップグレードする際に編集が必要となります。

CA Plex r4.0 からのアップグレード

このセクションでは、r4.0 に関連したアップグレード要件について説明します。

置き換えとスコープされたオブジェクト

CA Plex r4.5 では、置き換えに対する制御の向上とパフォーマンスを改善するために、スコープしているオブジェクトに付けられた置き換えトリプルの解決方法が変更されました。この変更は、デフォルト設定では有効になっていません。この修正を有効にしたときに深刻な下位互換性の問題が発生する可能性があるため、有効にするにはそのための設定を Plex.ini ファイルに追加する必要があります。詳細については、オンライン ヘルプのトピック「置き換えおよびスコープされたオブジェクト」を確認してください。

CA Plex r3.5 からのアップグレード

「CA Plex r4.0 からのアップグレード」の説明に加え、次のセクションを確認してください。

For Update オプションの動作の変更

r4.0 では、悲観的同時実行をサポートするために、Position および Fetch アクション ダイアグラム ステートメントの For Update オプションの動作を変更しました。これらの変更により、ODBC ベースのアプリケーション (Windows サーバアプリケーションを含む) の動作が大きく変化する可能性があります。下位互換性オプションとその他の詳細については、オンライン ヘルプの「SQL インプリメンテーションで行をロックする方法」を参照してください。

C++ コードでの CONCAT 演算子の変更

CONCAT 演算子の動作説明では、後続ブランクはフィールド値から除去されることになっています。しかし、実際にはパネルには表示されませんが C++ ファンクションでブランクを保持することが可能でした。この矛盾は CA Plex r4.0 で修正され、後続ブランクはどのような場合にも除去されるようになりました。連結する文字列の中に空白を含める手法としては、ブランクが埋め込まれたメッセージとともに **Format Message** ステートメントを使用する方法をお奨めします。

System i フィールド値ファイルにアクセスする際のランタイムのレベル チェックの回避

System i 上の CA Plex ライブラリ内には YOBDDL プログラムがあります。System i フィールド値ファイルを CA Plex 内に構築すると、System i オブジェクト ライブラリに YOBDDL プログラムのコピーが作成されます。このコピーの名前は YOBVALSV です。YOBVALSV がすでに存在している場合、コピーは作成されません。

この YOBDDL プログラムは、CA Plex r4.0 で修正されました。そのため、古いバージョンの YOBVALSV を、各 System i オブジェクト ライブラリから削除する必要があります。System i で次のコマンドを入力します。

DLTPGM PGM(オブジェクト ライブラリ/YOBVALSV)

次に、新バージョンの YOBVALSV を System i の各オブジェクト ライブラリに配置します。これは次のいずれかの方法で実行します。

- System i フィールド値ファイルを再構築する
- System i 上で次のコマンドを入力して複製オブジェクトを作成する

```
CRTDUPOBJ OBJ(YOBDDL) FROMLIB(PLEX)
OBJTYPE(*PGM)
TOLIB(オブジェクト ライブラリ) NEWOBJ(YOBVALSV)
```

上記の手順に失敗すると、フィールド値の選択リストにアクセスする際に実行時のレベルチェックが行われることになります。

ダイナミック アプリケーション パーティショニング API

GetLocationInformation と SetLocationInformation の API を使用している場合は、CA Plex r4.0 で構造体 ObLocationInfo が変更されたことに注意してください。これらの API を実装したソース コードのオブジェクトを作成している場合、コードを編集してコンパイル エラーを回避する必要があります。

次の 3 つのフィールドが削除されています。

```
ObLongFld m_fDataConv  
ObLongFld m_fObTran  
ObCharFld m_ObTranDLL
```

これらは、次のフィールドに置き換えられています。

```
ObLongFld m_iDataConv
```

必要なソース コードの例については、Odap.mdl サンプル モデルを参照してください。

アップグレードしたローカル モデルでのリンカ オプション

CA Plex r4.0 では、カスタム C++ ビルド オプションという新しい機能が導入されました。リンカ オプション /OPT:NOWIN98 はデフォルトで使用されます。このオプションなしでコンパイルされた DLL は極めて大きくなります。ローカル モデルが、r4.0 より前の CA Plex で作成されている場合、このオプションはデフォルトで表示されないため、手動で追加する必要があります。詳細については、オンライン ヘルプのトピック「C++ 構築のカスタマイズ」を参照してください。

ランタイム プロパティ DLL のロード

ランタイム プロパティ DLL をロードするのに必要なソース コードが、CA Plex r4.0 で変更されました。アプリケーションが SetProperty API を使用している場合は、以下に示すようにランタイム プロパティ DLL をロードするソース コードを変更してください。

```
#ifdef _DEBUG  
ObPropertyAPI::SetValue(Ob600Prpd.dll,  
                        OB_ATOM_ATOM_INSTALL, 0, 0)  
#else  
ObPropertyAPI::SetValue(Ob600Prp.dll,  
                        OB_ATOM_ATOM_INSTALL, 0, 0)  
#endif
```

ユーザ データ (BSUPPORT) および取引先 (BCONTACT) のパターン

CA Plex r4.0 では、パネルを用いたファンクションは、**Services** から、**UI** に移動されました。CA Plex r3.5 で、これらのパターンのいずれかを以前に使用している場合は、このパターンから継承したファンクションのいくつかを再スコープする必要があります。新バージョンのライブラリから抽出すれば、オブジェクトブラウザで **Services** の下に既存のパネル ファンクションが実オブジェクトとして表示され（黄色）、同じ名前の新しいファンクションが **UI** の下に表示されます（まだ実オブジェクトではないので色はグレー）。既存のパネル ファンクションを再スコープするには、これらを **Services** ファンクションからドラッグして **UI** ファンクションにドロップします（**UI** ファンクションは、これを実行する前に実オブジェクトにしなければなりません）。

CA Plex r3.1 および r3.0 からのアップグレード

「CA Plex r4.0 からのアップグレード」および「CA Plex r3.5 からのアップグレード」の説明に加えて、CA Plex r3.1 が、16 ビットの Windows アプリケーションの作成をサポートする最後の CA Plex リリースであることに留意してください。

CA Plex r2.51 あるいはそれ以前のリリースからのアップグレード

r2.51 以前のリリースの CA Plex をアップグレードするには、最初に CA Plex r3.0 にアップグレードする必要があります。CA Plex r3.0 で文書化しているアップグレード手順に従ってください。

第 3 章: システム要件

以下は、CA Plex でアプリケーションを開発するために必要とされるハードウェアとソフトウェアの最小要件です。

[PC 開発環境](#) (39ページ参照)

[プラットフォーム固有のシステム要件](#) (40ページ参照)

[Windows C++ クライアント\(WinC ジェネレータ\)](#) (40ページ参照)

[System i \(System i 5250 と System i クライアント/サーバ ジェネレータ\)](#) (40ページ参照)

[C# サーバ ジェネレータ](#) (41ページ参照)

[C# クライアント ジェネレータ](#) (41ページ参照)

[Java ジェネレータ](#) (41ページ参照)

[EJB オプション](#) (41ページ参照)

[Windows C++ サーバ \(WinNTC ジェネレータ\)](#) (42ページ参照)

[配布先のシステム要件](#) (42ページ参照)

PC 開発環境

これは、推奨要件です。この要件を満たしていないと使用できないということではありません。

- 1 GHz 32-bit (x86) または 64-bit (x64) プロセッサ
- Windows 7、Windows XP (SP2 または SP3) または Windows Vista SP1 または Windows Server 2008
注: Windows の Server Core バージョンは、Plex の開発、運用、どちらの環境でもサポートされません。
- 1 GB 以上の RAM
- 40 GB 以上のハード ドライブ (4 GB 以上の空き領域を推奨)
- CD-ROM ドライブ
- 1024 x 768 以上の解像度を持つ 17 インチ カラー モニタ

Windows C++ アプリケーションを開発する場合は、Visual Studio 2005 に関する要件についても確認する必要があります。

<http://www.microsoft.com>

プラットフォーム固有のシステム要件

システム要件は、生成されたアプリケーションを展開させるプラットフォームに依存します。また、プラットフォームのための要件は、オペレーティング システム、コンパイラ、データベースの新しいバージョンがリリースされるたびに変化します。

Windows C++ クライアント (WinC ジェネレータ)

次に示すのは、Windows C++ クライアントの要件です。

- Microsoft Visual Studio 2005 Standard、Professional または Architect edition
注:
 - Visual Studio Express Edition は、MFC ライブラリを含んでいないため Plex C++ 構築と互換性がありません。
 - Visual Studio 2005 SP1 を適用する必要があります。(次のサイトからダウンロードしてください。 <http://www.microsoft.com>)
 - Visual Studio 2008 は CA Plex r6.1 C++ 構築とは互換性がありません。他の用途のために、同じマシン上に Visual Studio 2005 と 2008 を導入し、並行して実行しても問題ありません。
- CA Plex WinC ランタイム システム
- ODBC ドライバとデータベース (クライアント側でのデータ接続が必要な場合)
- 展開先プラットフォーム
 - Microsoft Windows XP SP2 または SP3
 - Microsoft Windows Vista SP1
 - Microsoft Windows 7

System i (System i 5250 と System i クライアント/サーバ ジェネレータ)

次に示すのは System i 5250 と System i クライアント/サーバ ジェネレータの要件です。

- TCP/IP
- CA Plex System i ライブラリ
- RPG III または RPG IV コンパイラ
- V5R4、6.1 または 7.1
- 5250 ジェネレータ用 CA ライセンス キー (任意)

C# サーバ ジェネレータ

次に示すのは C# サーバ ジェネレータの要件です。

- Microsoft .NET Framework Version 4.0 (CA Plex インストール CD から導入できます)。32 ビットと 64 ビットの両 .NET Framework がサポートされます。
- DBMS (Microsoft SQL Server 2008, Oracle 11g, IBM DB2 9.5) そして OLE DB プロバイダ
- データベース構築に使用する ODBC ドライバ (実行時には必要ありません)

注: Microsoft Visual Studio は、C# 開発には必要ありません。任意で、デバッグや他の開発作業用に Visual Studio をご利用ください。

C# クライアント ジェネレータ

次に示すのは C# クライアント ジェネレータの要件です。

- Microsoft .NET Framework Version 4.0 (CA Plex インストール CD から導入できます)。32 ビットと 64 ビットの両 .NET Framework がサポートされます。
- .NET クライアント ジェネレータ用 CA ライセンス キー

Java ジェネレータ

次に示すのは Java ジェネレータの要件です。

- Sun J2SE Development Kit 6.0 – 32 ビットと 64 ビット Java Runtime Environment の両方がサポートされます。
- Apache ANT 1.8.2 (CA Plex とともに自動的に導入されます)
- サードパーティ DBMS (Oracle、Microsoft SQL Server など) と JDBC ドライバ
- データベース構築に使用する ODBC ドライバ (実行時には必要ありません)

EJB オプション

次に示すのは EJB オプションの要件です。Java ジェネレータへの追加要件となります。

- Java EE アプリケーション サーバ (Oracle GlassFish、Oracle WebLogic、JBoss、IBM WebSphere など)
 - Java EE 6 SDK
- 注:** アプリケーションは、J2EE 1.3 によってパックされなければなりません。
- EJB コネクタ ジェネレータ用 CA ライセンス キー

Windows C++ サーバ (WinNTC ジェネレータ)

次に示すのは Windows C++ サーバの要件です。

- CA Plex Windows アプリケーション サーバ
- Microsoft Visual Studio 2005 Standard または Professional edition。Visual Studio についての詳細は、本書の Windows C++ クライアント (WinC ジェネレータ) セクションを参照してください。
- DBMS (Microsoft SQL Server 2008、Oracle 11g、IBM DB2 9.5) — 通常、最新のサービス パックの適用が推奨されます。
- Microsoft Windows Server 2003 または Microsoft Windows Server 2008

注: Windows C++ サーバ ジェネレータは既存の WinNTC アプリケーションの保守のためにのみ提供されます。CA は、Windows サーバ プラットフォームのための新規アプリケーション開発には、C# もしくは Java ジェネレータの使用を強く推奨します。

配布先のシステム要件

開発の場合と同じく、配布先のシステム要件は対象のプラットフォームに依存します。現在サポートされている配布先のプラットフォームの詳細については、CA テクニカル サポート — CA Plex 製品ホーム ページ上の互換性マトリックスを参照してください。

<http://www.ca.com/jp/support/>

第 4 章: インストールについての注意事項

CA Plex のインストールおよびアップグレード手順については、『CA Plex 導入ガイド』を参照してください。

CA Plex のサイレント (自動) インストールが可能です。サイレント インストールについての詳細は、『CA Plex 導入ガイド』を参照してください。

Microsoft Help Compiler のインストール

Microsoft Visual Studio 2005 には、HLP フォーマットのヘルプ ファイルを作成するためのヘルプ コンパイラが含まれていません。

Plex で HLP ファイルをコンパイルするには、別途ヘルプ コンパイラをダウンロードし、インストールする必要があります。Windows 95 Help Authoring Kit (hwcsetup.exe) は、次のサイトからダウンロードできます。<http://www.microsoft.com> MS ヘルプ コンパイラをデフォルトのロケーション C:\Program Files\Help Workshop にインストールしてください。

ヘルプ トピック コンパイルのための最小限の要件は、C:\Program Files\Help Workshop フォルダ内に 2 つのファイル HCRTF.EXE と HWDLL.DLL が存在していることです。

第 5 章: 公開修正モジュール

CA テクニカル サポート (<http://www.ca.com/jp/support/>) で公開されているソリューションを介して、本製品の公開修正モジュールの完全なリスト見つけることができます。

第 6 章: 既知の問題

このセクションでは、このリリースにおける CA Plex の既知の問題、回避策、および対処法を説明します。更なる情報については、CA テクニカル サポートを参照してください。
<http://www.ca.com/jp/support/>

内部テーブルの修正

CA Plex の内部テーブルを編集して独自のオブジェクト タイプや動詞を追加することもできますが、この機能は文書化されておらず、サポートの対象外になります。CA では、この方法で修正されたモデルのサポートは保証できません。どのような形であれ、CA Plex の内部テーブルには修正を加えないよう、強くお奨めします。

同様に、グループ モデル テーブル ファイルを手動で編集することもサポートされていません。これには lib.tab ファイルも含まれます。lib.tab ファイルには、グループ モデル ID の別名を制御する重要なデータが含まれています。

Visual Studio 2005 での C++ 構築パフォーマンス

以前のリリースと比較し、C++ 構築パフォーマンスが遅くなる可能性があります。Visual Studio 2005 を使用し、WinC および WinNTC 構築のパフォーマンスを改善するヒントについては、次の CA Support Online - SupportConnect - Knowledge Base 内の技術資料『TEC415593』を参照してください。 <http://ca.com/support>

Java クライアント パネルにおける IME Control の制限

Java クライアントでは、すべての IME Control プロパティ値または FLD IME control SYS トリプルで指定可能な値が有効なわけではありませんので注意してください。有効な値は次の通りです。

プロパティ値	有効?	説明
None	Yes	
On		注：指定された場合、Off と同等の振る舞いとなります。
Off	Yes	
Disabled		注：指定された場合、Off と同等の振る舞いとなります。

プロパティ値	有効?	説明
DBCS Roman	Yes	
DBCS Katakana		注：指定された場合、DBCS Hiragana と同等の振る舞いとなります。
DBCS Hiragana	Yes	
SBCS Roman Yes	Yes	
SBCS Katakana Yes	Yes	

第 7 章: インターナショナル サポート

インターナショナル版の製品とは、要求されるオペレーティング システムとベンダ製品のローカル言語バージョンで正しく実行され、ローカル言語の入出力データをサポートする英語の製品です。インターナショナル版の製品でも、日付、時刻、通貨、数値フォーマットのためにローカル言語の表現法で指定を行なえます。

翻訳版の製品 (しばしば、ローカライズ版製品と呼ばれます。) は、インターナショナル版の製品に、日付、時刻、通貨、数値フォーマットのためのローカル言語のデフォルト設定とともに、製品のユーザ インターフェース、オンライン ヘルプ、その他ドキュメントのローカル言語サポートを含めたものです。

この製品の英語のリリースに加えて、**CA** は、次の表内に記されている言語をサポートします。

言語	インターナショナル版	翻訳版
ブラジル系ポルトガル語	Yes	No
中国語 (簡体字)	Yes	No
中国語 (繁体字)	Yes	No
チェコ語	Yes	No
チェコ語	Yes	No
チェコ語	Yes	No
フィンランド語	Yes	No
フィンランド語	Yes	No
ドイツ語	Yes	No
ギリシャ語	Yes	No
ハンガリー語	Yes	No
イタリア語	Yes	No
日本語	Yes	Yes
韓国語	Yes	No
ノルウェー語	Yes	No
ポーランド語	Yes	No
ロシア語	Yes	No
スペイン語	Yes	No

言語	インターナショナル版	翻訳版
スウェーデン語	Yes	No
トルコ語	Yes	No

英語以外の言語では、限定されたテストが行なわれています。この限定されたテストは、**CA パートナ**、顧客、または **CA Plex** の以前のリリースからの継続したテストをベースに構成される場合があります。

これは、英語のリリースが、英語以外のオペレーティング システムのマシンで実行できるであろうことを意味し、それらの市場または言語のために製品がローカライズされているということを意図したものではありません。

第 8 章: マニュアル

次のガイドは、自動的に製品と一っしょに導入されます。また、次のサイトからも入手できます。<http://ca.com/support>

- CA Plex 導入ガイド (Plex_Getting_Started_JPN.pdf)
- CA Plex リリース ノート (Plex_Release_JPN.pdf)
- CA Plex 5250 チュートリアル (Plex_Tutorial_System_ENU.pdf)
- CA Plex Windows チュートリアル (Plex_Tutorial_Windows_ENU.pdf)

PDF ファイルを表示するには、(マシンに導入されていない場合) Adobe Reader をインストールする必要があります。

注: このリリースのために更新されたガイドは、次のサイトの「Documentation」ページ「CA」製品ドロップダウン リスト下にあります。 <http://ca.com/support>