

CA Plex

導入ガイド

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.

CA 製品リファレンス

本書は、次の CA 製品について記載しています。

- CA Plex

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。

CA 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://tinyurl.com/6j6ugb>

目次

第 1 章: 概要	11
はじめに	11
製品コンポーネント	11
サポートされるオペレーティング環境	12
追加の情報リソース	14
第 2 章: CA Plex のインストール.....	15
インストール チェックリスト	15
Unicenter Software Delivery を使用した CA Plex インストール	20
標準構成	21
ベース プロダクトのインストール.....	22
サイレントインストール	24
独自 iss ファイルの記録.....	25
システム パスの修正	25
CA Plex r7.0 と CA Plex r6.1、CA Plex r6.0 の並行導入.....	25
Adobe Reader のインストール	26
Java Development Kit のインストール	26
CA ライセンス キーの入力	27
CA ライセンス キー ファイルの作成	27
既存のライセンス キー ファイルの修正	27
CA ライセンス (ALP) についてのよくある質問と回答	28
ベース プロダクトの変更、修正、削除	31
その他のコンポーネントのインストール	31
Windows アプリケーション サーバのインストール.....	31

アプリケーション インテグレータ	32
Microsoft Visual Studio 2005.....	33
第 3 章: 初めてのアプリケーションを 20 分で開発	35
要件.....	35
64 ビット Windows のための追加要件.....	35
Windows Vista と Windows Server 2008 のための追加要件	36
概念.....	36
再利用可能なビジネス パターン.....	36
ワーク グループによる開発	36
コードの生成	37
用語	37
オブジェクト タイプ	38
ツールバー アイコン	39
プロジェクト管理アプリケーション.....	40
サンプル モデルのオープン	41
オブジェクト ブラウザ	42
「Project」エンティティとトリプル	43
「Project」エンティティの属性の指定.....	44
フィールド継承	47
継承を使用したフィールドプロパティの定義.....	47
継続トリプル	51
フィールドを任意にするために継続トリプルを使用	52
パターンと継承	53
継承を使用した「Project」エンティティの定義.....	54
「Project」エンティティへの機能の追加.....	54

スコープの詳細	58
Java のための生成構築オプションのセットアップ	58
生成と構築	59
ネイティブ プラットフォームの実装	60
メッセージ ログ	61
生成されたアプリケーション	62
「Project.Edit」 ファンクションの実行	62
デフォルトのパネル レイアウト	63
新規プロジェクトの追加	64
100 %のコード生成	65
データの保存	65
パネル デザイナ	66
ビジュアル開発	66
パネル デザイナのオープン	66
デザイン ウィンドウ	68
パネル パレット	69
プロパティ シート	70
複数行編集コントロールの定義	71
スピン コントロールの追加	75
ファンクションの再生成	76
復習	78
第 4 章: グループ モデル ライセンス	79
ローカル モデル ライセンス	79
[グループ モデル ライセンス] ダイアログへのアクセス	80
グループ モデル ライセンスの転送	80

直接転送	80
ライセンスのフロッピー ディスク転送	81
第 5 章: パターン ライブラリ	85
バージョン名とレベル名	85
パターン ライブラリのインストール	85
クラス ライブラリのインストール	85
ネットワーク上へのパターン ライブラリのインストール	86
第 6 章: System i コンポーネント	87
製品に含まれるもの	87
System i 上の開発に必要な最小要件	87
System i 上の展開に必要な最小要件	88
ライブラリ リストに関する注意事項	89
CD から System i への製品ライブラリの転送	89
保管ファイル (*SAVF) からの製品ライブラリのリストア	90
PC - System i 間の通信用ソフトウェア	91
CA Plex TCP/IP 環境の構成	91
YOBSYTCPDP TCP/IP ディスパッチャ プログラム	92
ディスパッチャ プログラムの代替バージョン	92
YOBSYTCPDP の V5R3 バージョン	92
V5R3 でのユーザ名制限	92
YOBSYTCP_R ディスパッチャ	93
ユーザ権限の要件	93
オブジェクト権限	93
System i TCP/IP ディスパッチャの開始	94
製品ライブラリの別名でのリストア	96

ジョブ記述の更新	96
PC ファイルの修正	96
第 7 章: Windows C++ サーバ コンポーネント	99
Windows C++ サーバの開発要件	99
Windows C++ サーバ コンポーネントのインストール	100
Windows クライアントの要件	100
アプリケーション サーバ	100
アプリケーション サーバのインストール	100
ジョブ ステータス データベース	101
構築サービスおよびディスパッチ サービスの構成	101
Oracle のサポート	102
Windows クライアント コンポーネント (Microsoft RPC) のインストール	102
Windows C++ アプリケーション サーバ	102
第 8 章: Java コンポーネント	103
Java の開発要件	103
Java コンポーネントのインストール方法	104
JAVA_HOME の環境変数を定義	105
ランタイム用のサーバの構成	105
System i での Java の設定	106
PLEXJVA700 ライブラリ	106
System i への Objava ディレクトリのコピー	107
System i 上の Java ディスパッチャを構成	107
Java クラスの最適化	107
System i 上の Java ディスパッチャの起動	108
第 9 章: CA 2E データ移行	109

製品に含まれるもの	109
CA 2E データ移行の要件.....	109
メモリ	109
System i に必要な最小要件	110
製品ライブラリのリストア	110
CA 2E 並行インストール.....	111
第 10 章: アプリケーション インテグレータ	113
製品に含まれるもの	113
システム要件	114
アプリケーション インテグレータのインストール	114
ODBC データ ソースの設定	115
System i からのインポートの準備.....	115
CA Plex の複数バージョンの実行	115
第 11 章: よくある質問と回答 (FAQ)	117
質問と答え	117
Appendix A: Acknowledgments.....	125
Sun Microsystems, Inc. Binary Code License Agreement	125
RSA Security	128
decNumber 3.32	128
Apache ANT	129

第 1 章: 概要

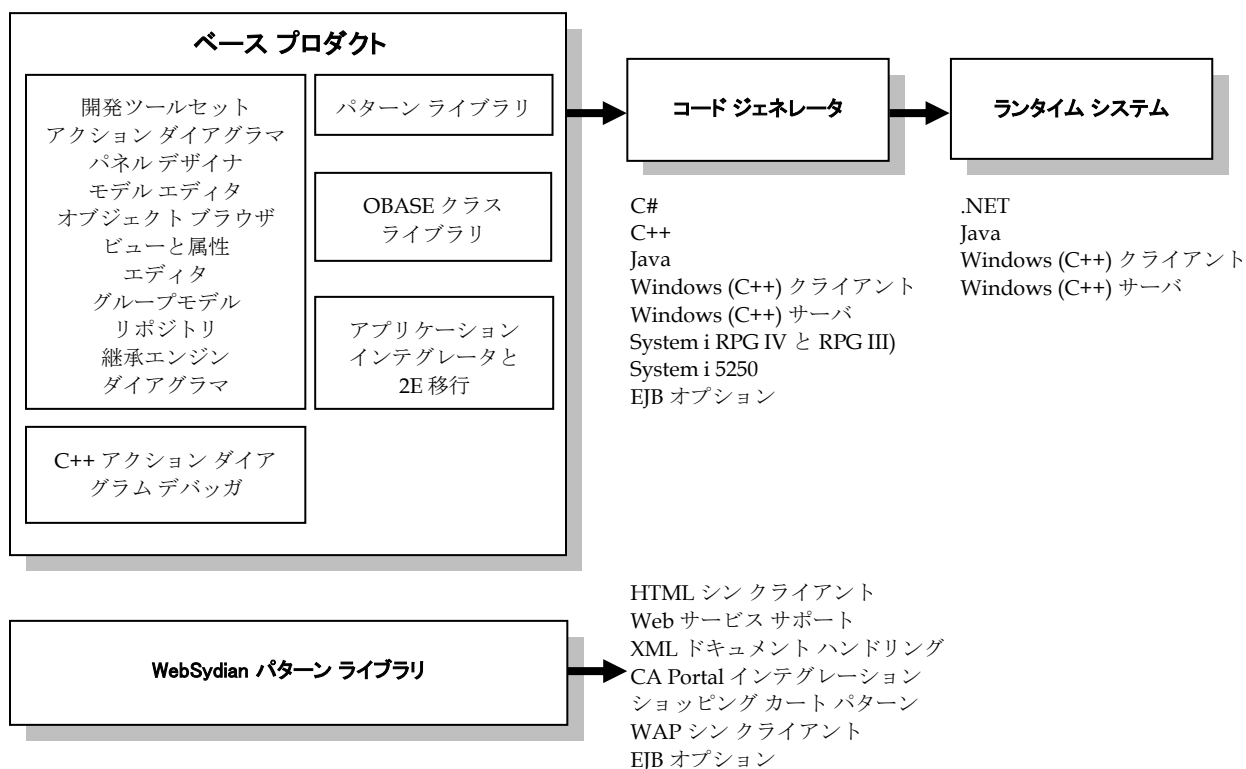
CA Plex は、アーキテクチャ型の RAD ツールとして、ソフトウェア パターン、モデル、およびジェネレータを使用して、Java、J2EE、Windows (.NET マネージドと MFC アンマネージドの両方)、IBM i の各オペレーティング環境に対応したビジネス アプリケーションの迅速な設計および保守を可能にします。

はじめに

はじめに、「CA Plex のインストール」の章で説明している基本的なインストール手順に従って、インストールを行ってください。続いて、アプリケーションの作成方法について概要を説明した「初めての Java アプリケーションを 20 分で開発」の章に目を通すことをお勧めします。”

製品コンポーネント

以下の図は、CA Plex の主要なコンポーネントを表しています。



CA Plex には以下が含まれます。

- GUI スクリーン デザイナ、特定の言語を指定しないビジネス ロジック エディタ、ダイアグラマ、および影響分析の各ツールを備えた **Windows** ベースの視覚的な IDE。
- 複数のバージョン、言語、およびプラットフォームにまたがって設計モデルを格納することができるマルチ開発リポジトリと組み込み構成管理。
- **Web**、ワイヤレス、**Java**、**System i** および **.NET** のオペレーティング環境用にアプリケーションを実装するために必要な **C#**、**C++**、**HTML**、**Java**、**RPG**、または **SQL** のコードを **100%** 自動的に作成するコード ジェネレータ。CA Plex は、**HTML**、**GUI** クライアント、**5250** ホスト スクリーン、サーバ プログラム、データベース オブジェクトを含む完全なアプリケーションを生成します。

CA Plex では、オペレーティング環境に固有の専門知識はほとんど必要ありません。開発者は、テクノロジーの変化を気にしなくてもよいので、技術的な問題ではなくビジネス上の問題の解決に力を注ぐことができます。

CA Plex によって生成されるアプリケーションは、一連の豊富な業界標準と **.Net**、**XML**、**J2EE**、**J2SE**、**WAP**、**Web** サービス、**Microsoft SQL Server**、**Oracle**、**DB2**、**Linux** を含むテクノロジーをサポートします。

サポートされるオペレーティング環境

CA Plex は、以下のオペレーティング環境用にネイティブな **N** 階層のコードを作成します。

- **Java** サーバ — **JDBC** データ アクセスを行なう **Java** サーバは、**Linux**、**Solaris**、および **i5/OS** を含む複数のオペレーティング環境にまたがりサポートされます。**J2EE** サポートのためには **EJB** オプションが用意されています。
- **.Net** サーバ — **.NET** ランタイム システムを備えた **C#** コード生成は、業界標準データベースへのデータ アクセスに **OLE DB** を使用します。
- **System i** サーバ — **RPG IV** または **RPG III** のコード生成および **DB2/400** への **DDS** アクセスをサポートします。
- **XML** ベースの **Web** サービス — **XML** ドキュメント ベースの **Web** サービスは、**Websyidian** パターン ライブラリを介してサポートされます。
- **HTML** **Web** クライアント。HTML Web クライアントは **Websyidian** パターン ライブラリを通します。
- **GUI** リッチ クライアント — リッチ GUI クライアントは、**Win32**、**Java** (**Swing**) と **.NET** (**WPF**) のオペレーティング環境、**ActiveX**、**JavaBean** GUI コンポーネントをサポートします。加えて、すべてのオペレーティング環境および業界標準データベースのための **ODBC** または **JDBC** を介したスタンドアロン クライアント データ アクセスをサポートします。

- **System i 5250** — System i 5250 は、RPG IV または RPG III のコード生成および DB2/400 への DDS アクセスとともにドラッグ アンド ドロップ対応の 5250 スクリーン デザイナを含みます。
- **Windows C++ サーバ** — Windows C++ サーバは、（OCI を介して）Oracle を（ODBC を介して）Microsoft SQL Server をサポートします。また、広範囲にわたる COM の統合をサポートします。

追加の情報リソース

本書を確認後、提供されている多くの情報リソースを参照することができます。例えば、下記は簡単なアプリケーションの作成を通してステップ バイ ステップの学習が行なえるチュートリアルです。

- **Windows** 版チュートリアル
- **System i 5250** 版チュートリアル
- **初めての Java** アプリケーションを 20 分で開発

CA Plex では、広範囲にわたる情報と **CA Plex** の使用方法に関する詳細手順を提供するオンライン ヘルプを用意しています。

第 2 章: CA Plex のインストール

CA Plex でアプリケーションを開発する際の、ハードウェアとソフトウェアの最小要件については『CA Plex リリース ノート』を参照してください。

インストール チェックリスト

CA Plex のインストール要件は、対象開発プラットフォームによって異なり、また要件には製品に同梱されていない他社製品も含まれます。

注: 詳細なバージョン情報とその他のシステム要件については、『CA Plex リリース ノート』を参照してください。

互換性は変更される場合があります。CA Plex と他社製品との互換性に関する最新情報については、サポート Web サイト (<http://www.ca.com/jp/support/>) を確認し、CA テクニカル サポート- 製品別サポートから CA Plex を選択してください。

次の表を使用し、どのコンポーネントをインストールする必要があるのか、また関連するインストール手順の参照先を確認してください。

コンポーネント	説明	参照先
ベース プロダクト		
CA Plex ベース プロダクト	CA Plex 開発環境、ヘルプ ファイル、サンプル モデル、およびチュートリアル モデルを含みます。	本章の「ベース プロダクトのインストール」
CA ライセンス キー	CA Plex およびそのオプションのコンポーネントを使用するには、ライセンス キーが必要です。オプションのコンポーネントは次のとおりです。 ■ エンタープライズ オプション (EJB と .NET クライアント ジェネレータが必要です) ■ IBM i 5250 ジェネレータ ■ Websydian パターン ライブラリ	本章の「CA ライセンス キーの入力」

コンポーネント	説明	参照先
パターン ライブラリとクラス ライブラリ		
パターン ライブラリ	最新のパターン ライブラリです。 CA Plex を用いたほとんどのアプリケーション開発で使用をお勧めします。	本書の「パターン ライブラリ」の章
クラス ライブラリ (OBASE)	CA Plex 用に開発された最初のパターン ライブラリのセットです。 System i 5250 プラットフォーム以外の新規開発プロジェクトのためにはお勧めできません。	classlib.chm ヘルプ ファイル
ツールとユーティリティ		
アプリケーション インテグレータ	さまざまなデータ ソースからの情報を CA Plex にリパース エンジンアリングするために使用されます。	本書の「アプリケーション インテグレータ」の章の「アプリケーション インテグレータのインストール」
CA 2E データ移行	CA 2E データ モデルのスキーマを CA Plex にインポートするために使用されます。	本書の「 CA 2E データ移行」の章の「 CA 2E データ移行の要件」
Microsoft Word Viewer	生成された Windows のレポートを表示および印刷するために使用されます。	オンライン ヘルプ『ユーザガイド』、「レポートの設計と印刷」の章の「印刷プログラムの指定」を参照)
Adobe Acrobat	オンライン版のチュートリアル ガイドを表示するために使用されます。	本書の Adobe Reader のインストール

コンポーネント	説明	参照先
プラットフォーム固有の要件		
C# ジェネレータ	■ アプリケーションを開発するには、Microsoft .NET Framework SDK Version 4.0 以上をインストールする必要があります。そのための Web インストーラが、Plex 製品 CD で提供されています。 ■ アプリケーションの実行には、Microsoft .NET Framework Version 4.0 以上をインストールする必要があります。そのためのフル インストーラが、Plex 製品 CD で提供されています。 ■ DBMS (Microsoft SQL Server, Oracle, IBM DB2) と OLE DB プロバイダ。 ■ データベース構築のための ODBC ドライバ。実行時には必要ありません。 注: Microsoft Visual Studio 2005、2008 および 2010 は、C# 開発のためには必要ありませんが、デバッグなどの開発作業を行うのに便利です。これらの開発のために推奨されるバージョンは、Microsoft Visual Studio 2010 です。	ObNET.chm ヘルプ ファイル.

コンポーネント	説明	参照先
Java ジェネレータ	■ Sun J2SE Development Kit 7.0 ■ Apache ANT (CA Plex とともに自動的にインストールされます。) ■ Microsoft SQL Server 、 Oracle、 IBM DB2 など他社の DBMS および JDBC ドライバ ■ データベース構築のための ODBC ドライバ。実行時には必要ありません。 ■ Java 構築を行うには、環境変数 JAVA_HOME が定義されている必要があります。	本書の「Java コンポーネント」の章。
EJB のオプション	■ WebLogic、JBoss、WebSphere などの J2EE アプリケーション サーバ。J2EE SDK には、テストに便利なアプリケーション サーバ (Reference Implementation 版) が含まれます。 ■ Java 2 Enterprise Edition SDK 1.4. ■ J2SE SDK. これは、Java ジェネレータの要件と同じです。 ■ JDBC ドライバ および DBMS。これは、Java ジェネレータの要件と同じです。	オンライン ヘルプ 『Java プラットフォーム ガイド』の「EJB オプション」

コンポーネント	説明	参照先
Windows C++ クライアント	■ Microsoft Visual Studio 2005 Standard edition もしくはそれより上のエディション。 ■ 注: Visual Studio Express edition は、CA Plex と互換性がありません。 ■ Windows XP、Windows Vista または Windows 7 のデプロイ プラットフォーム ■ CA Plex WinC ランタイム システム ■ ODBC ドライバとデータベース (Optional)	Microsoft Visual C++ コンパイラ
Windows C++ サーバ (WinNTC ジェネレータ)	■ CA Plex Windows アプリケーション サーバ ■ Microsoft Visual Studio 2005 ■ DBMS (Microsoft SQL Server, Oracle, IBM DB2) ■ Windows Server 2003 または Windows Server 2008	本書の「Windows C++ サーバ コンポーネント」の章の「Windows C++ サーバ コンポーネントのインストール」
System i (System i 5250 および System i クライアント/サーバ ジェネレータ)	■ CA Plex System i ライブラリ。 ■ TCP/IP ■ サポートされている i5/OS バージョン情報については、「1: 概要 - サポート オペレーティング環境」(ページ 1-2) を参照してください。 ■ RPG IV または RPG III コンパイラ	本書の「System i コンポーネント」の章
オープン データベース (ODBC)	■ ODBC ドライバおよび DBMS が必要です。	

注: 以前のバージョンからアップグレードする場合の手順の詳細については、リリースノートを参照してください。

CA Plex (または、コンポーネントのいずれか) をインストール、または、前バージョンからの統合を行う前に、次の章を確認し、本リリースのために更なる導入または構成設定が必要かどうかを判断してください。

Unicenter Software Delivery を使用した CA Plex インストール

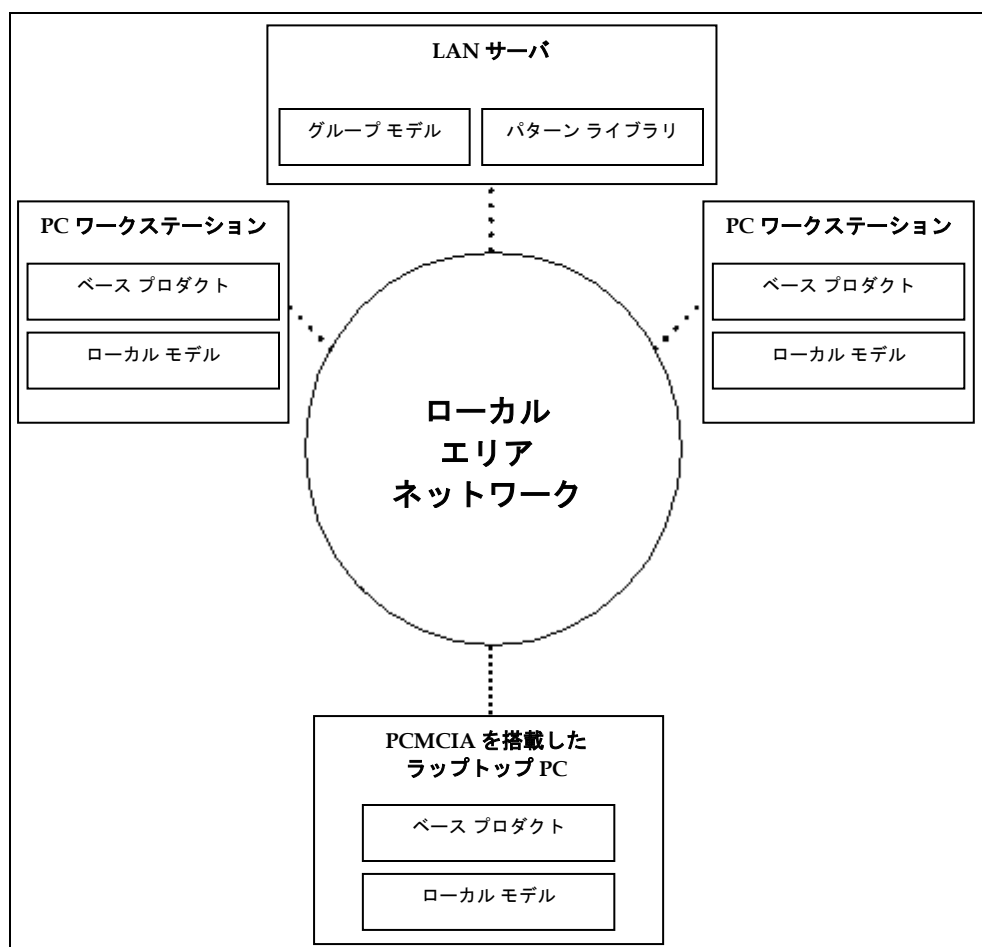
Unicenter Software Delivery (USD) を使用して CA Plex をインストールすることができます。そのための USD パッケージを CA サポート Web サイト (<http://ca.com/support>) で提供しています。

USD を使用した CA Plex インストール

1. CA サポート Web サイトから USD パッケージをダウンロードします。
2. ご使用の SDO Server に USD パッケージを解凍します。
3. SDO Server とともに USD パッケージを登録します。
4. 目的のシステムへ USD パッケージをデプロイします。

標準構成

CA Plex の構成は、さまざまな方法で行うことができます。以下の図は、LAN 環境における CA Plex コンポーネントの一般的な配置を示しています。



グループ モデル ライブラリとパターン ライブラリは、LAN サーバ上に常駐します。ベース プロダクトとローカル モデルは、開発者のワークステーションに常駐します。これらは、ネットワークへの接続が可能なラップトップにも常駐できます。この例では、グループ モデルを使用する場合以外はネットワークに接続する必要はありません。

ベース プロダクトのインストール

作業を開始し、最初のアプリケーション開発を行うためには、ベース プロダクトをインストールしなければなりません。

注: CA Plex をインストールするには、管理者権限が必要です。管理者としてログインせずに、CA Plex をインストールしようとした場合、エラー 1628: Failed to complete installation from InstallShield. が発生します。

CA Plex ベース プロダクトを開発ワークステーションにインストール

1. CA Plex 製品 CD を CD-ROM ドライブに挿入します。
[CA Plex インストール CD] ウィンドウが表示されます。
2. CA Plex ドキュメントを読み、インストールとアップグレード情報を確認するには、[Read] ボタンをクリックします。
3. [インストール] をクリックしてインストールを開始します。
4. 表示されるメニューから、以下のいずれかのインストールを開始します。

CA Plex

CA Plex ベース プロダクトをインストールします。

CA Plex Windows C++ アプリケーション サーバ

Windows サーバに WinNTC アプリケーションをデプロイするために必要な CA Plex Windows C++ アプリケーション サーバをインストールします。このオプションは、開発に必要な構築ツールを含みません。構築ツールをインストールするには、ベース プロダクトのカスタム インストールを実行してください。

Websyidian Web ライブラリ

CA Plex アプリケーションを Web にすることが可能な他社製のライブラリをインストールします。

Microsoft .NET Framework 4.0

Microsoft .NET Framework 4.0 をインストールします。インストール後に再起動が必要な場合があります。Microsoft .NET Framework 4.0 のインストールは、.Net ベースの開発のためには必ず必要です。

Microsoft Windows SDK for Windows 7 と .NET Framework 4

Web インストーラで、Microsoft .NET Framework SDK version 4.0 をインストールします。

Java JDK 7.0

CA Plex で Java 開発を行うのに必須のコンポーネント Java Development Kit (JDK) 7.0 をダウンロードするための Sun Web サイトへのリンクです。
C#、Windows C++ または System i RPG の開発には必要ありません。

[CA Plex] をクリックすると [InstallShield Wizard] ウィンドウが表示されます。

5. [次へ] ボタンをクリックします。

[使用承諾契約] ウィンドウが表示されます。

6. ライセンス契約の内容を確認し、契約の条項を受け入れる場合は [同意する] を選択し、[次へ] ボタンをクリックします。

個人情報を入力するダイアログが表示されます。

7. 名前と会社名を入力し、[次へ] ボタンをクリックします。

[セットアップ タイプ] ダイアログが表示されます。

8. インストールするセットアップのタイプを指定します。

[標準]構成

ヘルプ ファイル、チュートリアル ファイル、およびサンプル ファイルを含むすべてのプログラム ファイルとパターン ライブラリがインストールされます。

[標準] インストールの場合、デフォルトでは **OBASE** クラス ライブラリおよび **Windows C++** サーバ コンポーネントはインストールされません。**OBASE** クラス ライブラリをインストールする場合は、[カスタム] インストールを選択します。

[コンパクト]構成

ディスク スペースを節約するために、プログラム ファイルとパターン ライブラリのみがインストールされ、ヘルプファイル、チュートリアル ファイル、およびサンプル ファイルはインストールされません。

[カスタム]構成

インストールする製品コンポーネントを指定することができます。

9. セットアップタイプを指定し、[次へ] をクリックします。デフォルトのインストール先 **C:\Program Files\CA\Plex\7.0** または **C:\Program Files (x86)\CA\Plex\7.0** をそのまま使用することも、別のディレクトリを指定してインストールすることもできます。

指定したインストール先ディレクトリに加えて、**Program Files\CA\SharedComponents\Jre** ディレクトリにも **CA Plex** によって **Sun** の **JRE** のコピーがインストールされます。

注: インストール プログラムで、前バージョンの **CA Plex** と同じディレクトリにこのリリースをインストールすることはできません。代わりに、別ディレクトリにインストールするか、前バージョンをアンインストールする必要があります。

10. [カスタム] を選択した場合、インストールするコンポーネントを選択します。[カスタム] インストールのデフォルト設定は、[標準] インストールと同じですが、クラス ライブラリ、アプリケーション インテグレータ、および **CA Plex Windows C++** アプリケーション サーバ **SDK** を選択することができます。
11. 生成される **Windows** クライアントでデフォルトとして使用する言語を選択します。この設定は、後で変更できます。

注: 詳細については、オンライン ヘルプ トピック 「国別言語ライブラリの指定」を参照してください。

12. 設定を変更したい任意のダイアログに戻るために [戻る] ボタンをクリックするか、インストールを開始するために [次へ] ボタンをクリックします。

13. ファイルをコピーする前に設定を確認し、インストールが完了したら [完了] ボタンをクリックします。

CA Plex がコンピュータにインストールされます。

14. プロンプトが表示されたら、コンピュータをすぐに再起動する、あるいは後で再起動するのいずれかを選択し、[OK] をクリックします。

CA Plex セットアップの実行中のインストール エラーを確認するには、次のコマンドを Windows コマンド プロンプトから入力します。

```
F:¥PlexSetup¥setup.exe /v"/L*v C:¥PlexInstall.log"
```

Windows Vista でこのコマンドを実行するには、管理者として Windows コマンド プロンプトを開始する必要があります。このコマンドは、C ドライブのルートに問題の診断を支援する情報を含んだ PlexInstall.log ファイルを作成します。

上記の例では、F: ドライブが CA Plex インストール CD が挿入されている CD/DVD ドライブです。

CA Plex Windows C++ アプリケーション サーバの場合は、コマンドは次のようになります。

```
F:¥Plex Application Server¥setup.exe /v"/L*v C:¥PlexInstall.log"
```

サイレント インストール

ユーザが介在せず製品を自動的にインストールする必要がある場合、サイレント インストールが役に立ちます。製品をサイレント モードでインストールするには、**setup.exe** コマンドとともに **/s** コマンド ライン スイッチを使用します。たとえば、コマンド ラインから次のように入力します。

```
d:¥Plexsetup¥setup.exe /s
```

あるいは、

```
d:¥Plexsetup¥setup /s /f1".¥setup64.iss"
```

d:¥ は、CD-ROM ドライブです。

最初のコマンドは、32 ビット Windows、デフォルト インストール ディレクトリへの標準的な CA Plex 製品インストールを実行します。2 番目のコマンドは、64 ビット Windows へのインストールを実行します。

Windows Vista または Windows 2008 のコマンド ライン ウィンドウを使用してサイレント インストールを実行する場合は、管理者として既にログオンしている場合でも、管理者としてコマンド ライン ウィンドウ プロンプトを開く必要があります。

サイレント モードで CA Plex をアンインストールするには、次のコマンドを使用します。

32 ビット Windows:

```
d:\Plexsetup\setup.exe /s /f1".\setupUnInstall.iss"
```

64 ビット Windows:

```
d:\Plexsetup\setup /s /f1".\setup64UnInstall.iss"
```

独自 iss ファイルの記録

同梱の **setup.iss** ファイルは、**CA Plex** が標準的なインストール オプションで、デフォルトの場所にインストールされるように記録されています。しかし、これは開発者の要件とは一致しない場合があります。

setup.exe コマンドとともに **/r** コマンド ライン スイッチを使用して独自の **setup.iss** ファイルを記録することができます。標準では **setup.exe** は **C:\Windows** フォルダに **setup.iss** ファイルを作成します。

setup.iss ファイルをリネームし、**setup.exe** を実行する際に **/f1** コマンド ライン スイッチとともにそのファイルを指定することができます。

システム パスの修正

CA Plex ベース プロダクトをインストールすると、自動的にシステム パスが変更されます。具体的には、**CA Plex** の **Bin** ディレクトリ（通常は、**C:\Program Files\CA\Plex\7.0\Bin**）がシステム パスに追加されます。

システム パスを確認または変更するには、[コントロール パネル] の [システム] アイコンをダブルクリックし、詳細設定タブから環境変数を開きます。

CA Plex r7.0 と CA Plex r6.1、CA Plex r6.0 の並行導入

CA Plex r7.0 C++ ランタイムは、**CA Plex r6.1** および **CA Plex r6.0** と下位互換があります。これは、再コンパイルをすることなく **CA Plex r7.0** バージョンのランタイムを使用して **CA Plex r6.1** および **CA Plex r6.0** で生成された **C++** ファンクションを使用できることを意味します。

注： 上位互換はありません。ランタイム エラーが発生するため **CA Plex r7.0** ファンクションで **CA Plex r6.1** および **CA Plex r6.0** ランタイムは使用しないでください。

7.0、6.1、6.0 の **C++** ランタイム DLL は、同じ名前を持っています。例えば、**ob600lc.dll**、**ob600nwi.dll** などです。**CA Plex** インストールがシステム パスを変更する際、**Plex 7.0 Bin** フォルダが **Path** 記述の始まりに置かれます。これは、デフォルトでその PC 上の **6.0** および **6.1** 両方の **WinC** アプリケーションが **7.0 WinC** ランタイム DLL を使用して実行されることを意味します。

Adobe Reader のインストール

オンライン版のチュートリアルを表示するには、Adobe Reader のインストールが必要です。

Adobe Reader のインストール

1. [CA Plex インストール CD] のウィンドウで、[ドキュメント] をクリックし、[Adobe Acrobat] のインストールをクリックします。
2. ダイアログに表示される指示に従います。

Java Development Kit のインストール

CA Plex で Java を使用してアプリケーション開発を行うには、J2SE Development Kit (version 7.0) をインストールする必要があります。

Java Development Kit のインストール

1. [CA Plex インストール CD] のウィンドウで、[インストール] をクリックし、他のインストール オプションを表示します。
2. [JDK 7.0 のダウンロード] ボタンをクリックします。
3. Web サイトの指示に従って JDK 7.0 をダウンロードおよびインストールします。

注: CA Plex をインストールした後で JDK コンポーネントをインストールする場合は、Java 開発に関連する CA Plex 構成変更を手作業で行なう必要があります。

CA ライセンス キーの入力

CA Plex ライセンシングを適用するには、CA Plex を実行しているマシン上の **ca.olf** ファイルに ALP キー認証からの情報を配置する必要があります。CA Plex のライセンス キー ファイル (**.olf**) は、製品とは別に電子メールで送信されます。受け取った **olf** ファイルを次の手順で適用してください。

CA ライセンス キー ファイルの作成

ca_lic フォルダ（通常は **Program Files¥CA¥SharedComponents** 内）で、**ca.olf** という名前のファイルを探します。存在しない場合は、ファイル作成を行います。

ca.olf ファイルの作成

1. 入手したライセンス ファイル **olf** を **ca_lic** ディレクトリ配下にコピーします。
2. ファイル名を **ca.olf** にします。

ファイル名は、すべて半角小文字を使用します。

既存の CA ライセンス キー ファイルの修正を行う場合は、次の手順に従ってください。

既存のライセンス キー ファイルの修正

ca.olf ファイルが存在する場合は、次の手順に従ってください。

ca.olf の修正

1. **ca.olf** のバックアップを取ります。（名前の変更または該当ディレクトリ以外の場所へ移動）
2. 入手したライセンス ファイル **olf** を最新の **ca.olf** とし、**ca_lic** ディレクトリ配下にコピーします。
3. メモ帳のようなテキスト エディタを使用して **ca.olf** ファイルを開きます。
4. 旧 **ca.olf** の「FEATURE」から始まる行を最新の **ca.olf** の最終行に追加します。
注: 既存の「FEATURE」行は削除しないでください。
5. 編集した **ca.olf** ファイルを **ca_lic** フォルダに上書き保存します。

注: 「FEATURE」行は、証明書の 2 行目に折り返されている場合がありますが、ca.olf ファイルに入力するときは改行を入れずに 1 行で入力する必要があります。

CA ライセンス (ALP) についてのよくある質問と回答

問: ALP とは何ですか？

答: ALP とは Automated License Program (自動ライセンス プログラム) の略で最新の CA ライセンシング、オーダー フルフィルメントです。CA ソフトウェアを実行、適切なライセンス ファイルをリクエスト/作成するために、お客様にハードウェア適合性を識別するためのクライアント サイド アプリケーションの使用を要求することなく、ALP 製品は CA ライセンス データベースに記録されているその製品とハードウェア情報をもとにした認証ファイルと共に出荷されます。

問: xxxxxx.olf というライセンスファイルが送られてきましたが、どのように使用するのですか？

答: 製品をインストールした環境の ca_lic ディレクトリに.olf ファイルをコピーします。ca_lic ディレクトリの場所や詳しい適用方法は[こちら](#)をご覧ください。

問: CA ライセンスが、C:\CA_LIC ではなく、Program Files サブ フォルダへインストールされることがあるのはなぜですか？

答: 最新の CA ライセンス インストールは CA インストール標準に準拠し、マシン上に既存の C:\CA_LIC ライセンス フォルダが見つからない場合は C:\Program Files\CA\SharedComponents\CA_LIC フォルダへインストールされます。

これに対して C:\CA_LIC フォルダがマシン上に既に存在する場合は、このフォルダに新たなライセンスが引き続きインストールされます。

インストール標準に準拠するライセンス パッケージは、バージョン 1.52 以上です。

ライセンスのダウングレード時、以下の問題が起こる場合があります。

インストール済みの既存のライセンスがないコンピュータ上にバージョン 1.52 以上のライセンスをインストールし、その後に旧バージョンのライセンスをインストール (ライセンスのダウングレード) すると、旧バージョンは自動的に C:\CA_LIC フォルダにインストールされ、2 つの CA_LIC フォルダ (C:\CA_LIC および C:\Program Files\CA\SharedComponents\CA_LIC) が作成されることになります。

この問題を修正するには、C:\Program Files\CA\SharedComponents\CA_LIC フォルダにある mergealic.exe を実行するか、あるいはライセンスの最新バージョンを supportconnect.ca.com サイトからインストールしてください。

これにより、2 つのフォルダが C:\Program Files\CA\SharedComponents\CA_LIC へマージされます。今後は、C:\Program Files\CA\SharedComponents\CA_LIC がライセンスの管理フォルダになっていきます。

問: mergeolf ユーティリティはどのように使用するのですか？

答: 構文: MERGEOLF -n <new_olf> -c <current_olf> -o <output_olf> -b <backup_olf> -d

-n は、マージするための新しい OLF ファイルの名前です。

-c は、マージするための現在の OLF ファイルの名前です。デフォルトは **ca.olf**。

-o は、作成するための新しい OLF ファイルの名前です。デフォルトは **ca.olf**。

-b は、現在の OLF ファイルのバックアップの名前です。デフォルトは **ca.olf**。

-d は、デバッグを有効にします。(mergeolf.log)

例:

MERGEOLF -n ca.no1 -c c:¥ca_lic¥ca.olf -o c:¥ca_lic¥ca.olf -b c:¥ca_lic¥ca.old

問: ALP キーの正しいフォーマットは？

答: 以下は、ALP Execution Key の例です。

ID_1 3991FFEF "Customer Name" Technical Contact Name (As recorded in our database)

ID_2 E4A19DB8 "Company Name" Company Name

ID_3 C2F2E617 "00000000" CA Site ID Number

ID_4 9550AD9 "email@domain.com" Technical Contact Email (As recorded in our database)

SERVER CAI 7152

DAEMON CAI_lic_d/usr/bin

FEATURE 0 CAI_lic_d 1.000 20-APR-2000 0 0CA4D081978DD1BD5C76 "(MODEL) (COMPONENT)" ANY # 0000000-000:

「FEATURE」行としてみなされる 7 行目が、実際の暗号化された **Execution Key** です。ID、Server、Daemon 行は、ファイル内で冒頭に一度だけ記載されているべきヘッダ行です。上記以外の行が存在する場合、ライセンスが無効になる可能性があります。

もし、ALP キーで問題が発生した場合は、次の点を確認してください。

- **ca.olf** ファイル内の「FEATURE」行を分断している改行がないかどうか。
- **ca.olf** ファイル内の「FEATURE」行間にブランク行がないかどうか。
- 新規に作成した **ALP Execution Key** の ID 行が、**ca.olf** ファイル内の既存の ID 行を置き換えていないかどうか。
- **OLF** の名前が **ca.olf** になっており、名前に余分な拡張子を持っていないかどうか。
例えば、**ca.olf.txt** または **ca.olf.olf**（これは、エクスプローラで「登録されている拡張子は表示しない」を有効にしている場合に **Windows** システムで最もよく起こりえる原因です。）

問: ALP ライセンス キーを導入しましたが、次のエラーが発生します。

- CA Licensing -- The license found is inadequate for the Hardware Unit rating of this machine. You might have upgraded the machine and if so, please rerun the appropriate license program to properly license your product

答: 通常、このエラーは **lic98.cap** ファイルが不正に破損しているために発生します。テキスト エディタで **lic.98.cap** ファイルを開き、ファイルが次のフォーマットの検出モデル タイプのリストで構成されていることを確認してください。

SYS (検出モデル) (階層) (チェックサム)

- ファイルが不正にフォーマットされている場合、インストール元から **lic98.cap** をコピーするか、サポート Web サイトから最新データ ファイルをダウンロードしてください。
- **lic98.cap** ファイルが存在しない場合、現行の **ALP** ライセンス ランタイムをインストールしていない可能性があります。**CA** ライセンス デスクにご連絡ください。
- **lic98.cap** ファイルが存在しており、有効に思われる場合、**CA** ライセンス デスクにご連絡ください。

ベース プロダクトの変更、修正、削除

CA Plex ベース プロダクトをインストールした後も、インストールを再度実行し、ベース プロダクトの変更、修正、削除を行えます。

CA Plex ベース プロダクトの変更、修正、削除

1. [スタート] - [設定] - [コントロール パネル] の順にクリックします。
コントロール パネル ウィンドウが表示されます。
2. [アプリケーションの追加と削除] アイコンをダブルクリックします。
プログラムの追加と削除ウィンドウが表示されます。
3. 製品のリストから [CA Plex] を選択し、[変更と削除] をクリックします。
プログラムの変更、修正、または削除を選択する画面が表示されます。
4. 必要に応じて、[変更]、[修正] または [削除] を選択します。
選択した項目のダイアログが表示されます。
5. [次へ] をクリックして、ダイアログ ボックスに表示される指示に従います。

その他のコンポーネントのインストール

ここでは、その他のコンポーネントをインストールする方法について説明します。

Windows アプリケーション サーバのインストール

CA Plex ベース プロダクトのインストール時、あるいは CA Plex ベース プロダクトをインストール後に、アプリケーション サーバ ツールをインストールすることができます。このプロダクト オプションは Windows C++ Server の開発に必要ですが、C# での開発には必要ありません。

CA Plex ベース プロダクトのインストール時にアプリケーション サーバ ツールをインストール

1. セットアップのタイプを指定する際、[カスタム] をクリックします。
2. すでにチェックされているものに加え、CA Plex Windows アプリケーション サーバをチェックします。

CA Plex ベース プロダクトのインストール後にアプリケーション サーバをインストール

1. [スタート] - [設定] - [コントロール パネル] の順にクリックします。
2. コントロール パネルで、[アプリケーションの追加と削除] アイコンをダブルクリックします。
3. 製品のリストから [CA Plex] を選択します。
4. [変更と削除] をクリックします。

5. InstallShield Wizard で、[変更] を選択してから [次へ] をクリックします。
6. [CA Plex Windows アプリケーション サーバ] をクリックしてチェックをいれます。
7. [次へ] ボタンをクリックすると、インストールが始まります。

アプリケーション インテグレータ

CA Plex ベース プロダクトのインストール時、あるいは CA Plex ベース プロダクトをインストール後に、アプリケーション インテグレータをインストールすることができます。

CA Plex ベース プロダクトのインストール時にアプリケーション インテグレータ ツールをインストール

1. セットアップのタイプを指定する際、[カスタム] をクリックします。
2. すでにチェックされているものに加えて、アプリケーション インテグレータをチェックします。

CA Plex ベース プロダクトをインストールした後にアプリケーション インテグレータ ツールをインストール

1. [スタート] - [設定] - [コントロール パネル] の順にクリックします。
2. コントロール パネルで、[アプリケーションの追加と削除] アイコンをダブルクリックします。
3. 製品のリストから [CA Plex] を選択します。
4. [変更と削除] をクリックします。
5. InstallShield Wizard で、[変更] を選択してから [次へ] をクリックします。
6. [アプリケーション インテグレータ] をクリックしてチェックをいれます。
7. [次へ] ボタンをクリックすると、インストールが始まります。

Microsoft Visual Studio 2005

CA Plex で C++ コード(WinC と WinNTC オブジェクト)の生成と構築を行うには、Microsoft Visual Studio 2005 C++ コンパイラをインストールする必要があります。Visual Studio の Standard edition、Professional edition、およびそれ以上のものは、CA Plex と互換性をもちます。

CA Plex に必要なのは、Microsoft Visual Studio の標準インストールのみです。構成が異なる場合の手順については、Visual C++ のマニュアルを参照してください。

次の点に注意してください。

- C# ジェネレータ、System i 5250 ジェネレータおよび Java ジェネレータには、Microsoft Visual C++ コンパイラは必要ありません。
- Windows サーバ (WinNTC) ファンクションを構築する場合は、そのサーバ ファンクションを構築するマシンに Visual Studio をインストールする必要があります。
- Microsoft Visual Studio 2005 Express edition は MFC をサポートしていないため、CA Plex C++ 構築と互換性がありません。

第 3 章: 初めてのアプリケーションを 20 分で開発

CA Plex は、パターンを使用してソフトウェア アプリケーションの設計、作成、および保守をスピーディに行うアーキテクチャ型 RAD ツールです。本章では、CA Plex の革新的な機能のいくつかを紹介します。

本章では、シンプルなプロジェクト管理アプリケーションを定義します。CA Plex のパターン ライブラリを作業の効率化のために使用し、パネル デザイナでダイアログに変更を行ないます。本章の終了までには、アプリケーションの起動および実行の第 1 段階が終了します。

本章の例では Java アプリケーションを作成します。例題を修了するために Java プログラミングについて知る必要は何もありません。開発プロセスは、開発しているプラットフォームに関係なくほぼ同じです。

要件

Java アプリケーションを作成する前に、次の作業を行なう必要があります。

- CA Plex のインストール
- 本書の「Java コンポーネント」章に記述されている Sun JDK 7.0 のインストール
- 本書の「Java コンポーネント」章に記述されている JAVA_HOME 環境変数の定義

64 ビット Windows のための追加要件

本チュートリアルを 64 ビット Windows で実行したい場合、次の追加要件を考慮する必要があります。

- チュートリアルは、32 ビット Microsoft access ODBC ドライバと JDBC-ODBC ブリッジを使用します。そのためチュートリアル アプリケーションを実行するには、JDK の 32 ビット バージョンをインストールし、使用する必要があります。
- これらの考慮点は、運用環境の 64 ビット システム上で稼動する Oracle、IBM DB2 または Microsoft SQL Server を使用する標準的なアプリケーションには適用されません。その場合は、64 ビット JDK そして 64 ビット JDBC ドライバが推奨されます。

Windows Vista と Windows Server 2008 のための追加要件

本チュートリアル アプリケーション Windows Vista と Windows Server 2008 で実行する場合、レコードの更新に失敗する可能性があります。これは、デフォルトの **Microsoft Access** データベースの使用に関連する制限のため発生します。チュートリアルを実行するためには、この制限を回避するために **Microsoft SQL Server Express** エディションなど他のデータベースを使用してください。

概念

以下のセクションでは、アプリケーション環境で使用される概念について説明します。

再利用可能なビジネス パターン

CA Plex 開発者は、パターンを再利用してアプリケーションをモデル化します。パターンは、ソフトウェア システムの再帰的な問題への解決と言えます。パターンは、いろいろな状況で再利用可能な抽象記述です。パターンの一例として **BStructure** ビジネス構造があります。

ビジネス構造パターンには、データベース スキーマ、ユーザ インターフェース設計、およびプログラムがあり、これらを利用して部品表、組織図、勘定科目一覧表などの階層的なデータ構造を実装できます。**CA Plex** には、このようなパターンのライブラリが収録されていますが、他社が提供する更なるライブラリを利用することもできます。

CA Plex パターン ライブラリからいくつかのパターンを使用して、サンプル アプリケーションを構築します。

ワーク グループによる開発

CA Plex には、ソフトウェア開発チームがアプリケーションの設計と構築を共同で作業できるようにするための、モデル ベースのワーク グループ環境が設けられています。この環境の心臓部ともいえるリポジトリには、以下の機能があります。

- 複数開発者のサポート—開発者はオフラインで作業し、その変更内容を中央リポジトリに対して更新できます。
- 組み込みの構成管理—1 つのモデルで単一のアプリケーション設計情報を複数のバージョン、プラットフォーム、および言語で格納できます。
- 複数のモデルにまたがって格納される設計情報の統合

便宜上、本章のサンプルでは単一ユーザ環境で使います。ローカル モデルと呼ばれるユーザ モデル作成され、用意されています。

コードの生成

リポジトリに保管された設計情報に基づき、CA Plex は、複数のプラットフォームへアプリケーションとデータベース オブジェクトを実装するためのコードをすべて自動で生成します。現在サポートされている実装オプションを次に示します。

- Java または HTML クライアントと、Java、.NET、System i サーバからなる多階層のウェブ アプリケーション
- C# WPF、Win32 C++ または Java Swing クライアントと、Java、.NET、System i サーバからなる多階層のクライアント/サーバ アプリケーション
- System i 5250 のキャラクタ ベースの端末アプリケーション

CA は、お客様の要望と技術の進歩に追従できるよう、利用可能なジェネレータを定期的に改良および拡張しています。ジェネレータは、低位の技術やその実装細部から開発者を分離し、CA Plex ユーザは、単に既存設計を再生成するだけで新たなプラットフォームを利用できます。

用語

本章では、「Project」という 'エンティティ' を作成します。「Project」エンティティは、開始日、終了日といった情報を格納するために 'フィールド' を使用します。エンドユーザがプロジェクトを作成、変更、削除できるように 'ファンクション' を継承します。エンティティ、フィールド、ファンクションはすべて CA Plex のオブジェクト タイプです。

'ローカル モデル' は、構築するアプリケーションの設計情報を格納するファイルです。ローカル モデルで、オブジェクトを作成し定義します。

'オブジェクト ブラウザ' は、モデル内のすべてのオブジェクトを表示します。このブラウザで、利用できるパターン ライブラリのオブジェクトや、モデルに自身で定義したオブジェクトを確認することができます。

オブジェクト タイプ

CA Plex は、多数のオブジェクト タイプを使用しています。次のリストは、本章でアプリケーションを構築する際に目にするアイコン、オブジェクト タイプ、略語、オブジェクト記述など CA Plex が各オブジェクト タイプを識別するために使用されるものを示しています。

オブジェクト タイプ	略語	アイコン	記述
Entity (エンティティ)	ENT		エンティティとは、情報を記録したい対象であり、実体を持つもの（顧客や注文など）と持たないもの（プロジェクトや役職名など）があります。
Field (フィールド)	FLD		フィールドとは、従業員名、価格、注文状況などエンティティについて記録したい情報の断片です。
Function (ファンクション)	FNC		ファンクションとは、アプリケーションの機能を定義した処理群です。1つのファンクションは、プログラムまたはメソッドとほぼ同等です。
Panel (パネル)	PNL		パネルとは、アプリケーションのユーザ インターフェースを構成するウィンドウとダイアログです。
Table (テーブル)	TBL		テーブルとは、リレーショナル データベースを使用するアプリケーションでデータを格納する物理的な構造です。
View (ビュー)	VW		ビューは、データベース テーブル内のデータの全部または一部を表現します。ビュー自体はデータを一切含みません。ビューは、そこからテーブル内のデータを見ることができる一種の窓と考えることができます。
Diagram (ダイアグラム)	DGM		ダイアグラムは、モデル内のオブジェクトのサブセットを視覚的に表します。これらは、オブジェクト、およびオブジェクト間の関係を示します。
Message (メッセージ)	MSG		メッセージはテキストを保持します。テキストは、エラー メッセージに使用されたり、ウィンドウやダイアログのキャプションなどとして使用されます。

ツールバー アイコン

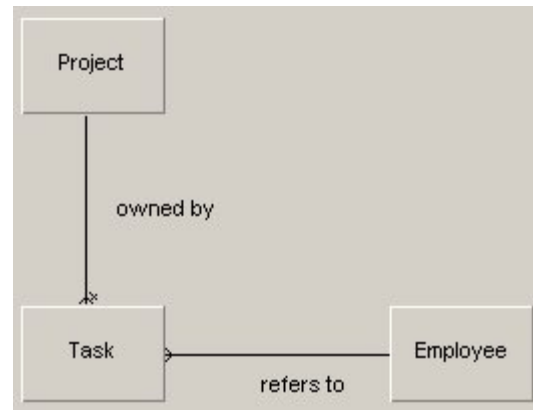
次のリストは、ツールバー アイコンのリストと各アイコンの記述を示しています。

アイコン	名前	記述
	モデル エディタ	モデル エディタを開きます。
	ブラウザの表示 / 非表示	オブジェクト ブラウザを表示または非表示にします。
	フィールド	フィールド オブジェクトを表示します。
	2 レベル	2 段深いレベルで、モデル エディタに詳細を表示します。
	1 レベル	1 段深いレベルで、モデル エディタに詳細を表示します。
	フォーカス解除	エディタの表示をリセットします。
	ライブラリ オブジェクトの表示 / 非表示	ライブラリ オブジェクトを表示または非表示にします。
	エンティティ	エンティティを表示します。
	検査	その対象を定義するトリプルのみを表示します。
	新しい生成と構築	新規に生成と構築ウィンドウを開きます。
	生成と構築	選択されたオブジェクトの、生成構築プロセスを開始します。
	WinC 構築結果	WinC の構築結果を表示します。
	実行	アプリケーションを実行します。
	コピー	現在のプロジェクト データを元に、新しいプロジェクトを作成します。
	エディタ	選択対象に基づいた適切なエディタを開きます。
	パネル パレット	パネル パレットを表示します。
	プロパティ	プロパティを表示します。
	チェック マーク	パネル エディタに入力した値を確定します。

プロジェクト管理アプリケーション

以下のセクションで、シンプルなプロジェクト管理アプリケーションを作成します。このサンプルでは、プロジェクトはタスク グループを構成し、各タスクには従業員が 1 人割り当てられます。モデルは「Project」、「Task」、「Employee」の 3 つのエンティティを持ちます。時間を節約するため、「Employee」エンティティはすでに定義されています。

次の図は、本章で作成するアプリケーションのエンティティ リレーションシップ ダイアグラムです。後で、このようなダイアグラムを作成します。



エンティティ リレーション ダイアグラムを扱ったことがある場合、図から以下を読み取ることができます。

- プロジェクトは複数のタスクを持てますが、タスクは 1 つのプロジェクトのみに従属します。
- タスクはプロジェクトに所有されています。これは、プロジェクトを削除すると、そのプロジェクトのタスクもすべて削除されることが期待されます。
- 従業員はタスクに割り当てられます。1 人の従業員を複数のタスクに割り当てることができますが、各タスクは割り当てられた 1 人の従業員しか持てません。
- 従業員はいかなる特定のタスクにも依存していません。すなわち、タスクを削除しても、その従業員レコードも削除することが必ずしも期待されるわけではないことを意味します。

いかなるコードを確認することなく上記のすべてを理解することができます。ダイアグラムは、CA Plex がそのアプリケーションを生成するために使用する有益な情報を示します。

このサンプル アプリケーションでは、モデルを非常に単純化しています。過去に実際のプロジェクト管理システムを開発したことがある方の場合、このサンプル モデルとは異なる設計をするだろうと思う部分があるかもしれません。

サンプル モデルのオープン

まず、CA Plex を起動して、用意されているサンプル モデルを開きます。

サンプル モデルのオープン

1. [スタート] - [プログラム] - [CA] - [CA Plex r7.0] - [CA Plex] をクリックします。

CA Plex アプリケーションが開きます。

2. [ファイル] メニューから [開く] を選択します。

[ファイルを開く] ダイアログが表示されます。

3. 「マイ ドキュメント」(Windows Vista または Windows 7 では「ドキュメント」) をクリックし、CA Plex 7.0 サブ フォルダに移動します。「Tutorial」ショートカット フォルダをクリックします。

4. 「TUTORIALJAVA.mdl」を選択し、開きます。

[オブジェクト ブラウザ] が表示され、アプリケーション メイン ウィンドウのタイトル バーにモデル名が表示されます。

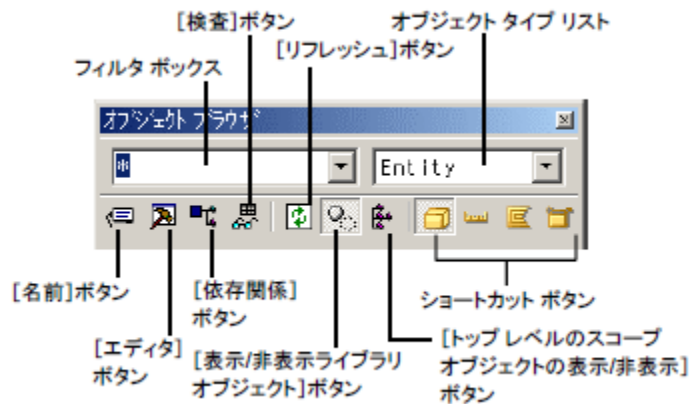
注: 作業を保存します。本章の途中でチュートリアルを中断する必要がある場合、[保存] ツール ボタンをクリックして進捗状況を保存できます。CA Plex では、エディタによっては終了時に、変更を加えていればその内容を保存するかどうか尋ねるプロンプトも表示されます。

オブジェクト ブラウザ

ツールバーの右端にある、[表示/非表示オブジェクト ブラウザ] ツールバー ボタンをクリックしてオブジェクト ブラウザの表示/非表示を切り替えます。

作業中、オブジェクト ブラウザを開いたままにしておき、パレットとして使用し、そこから必要なオブジェクトをドラッグできます。

次の図は、アプリケーションで表示した際のオブジェクト ブラウザを示しています。



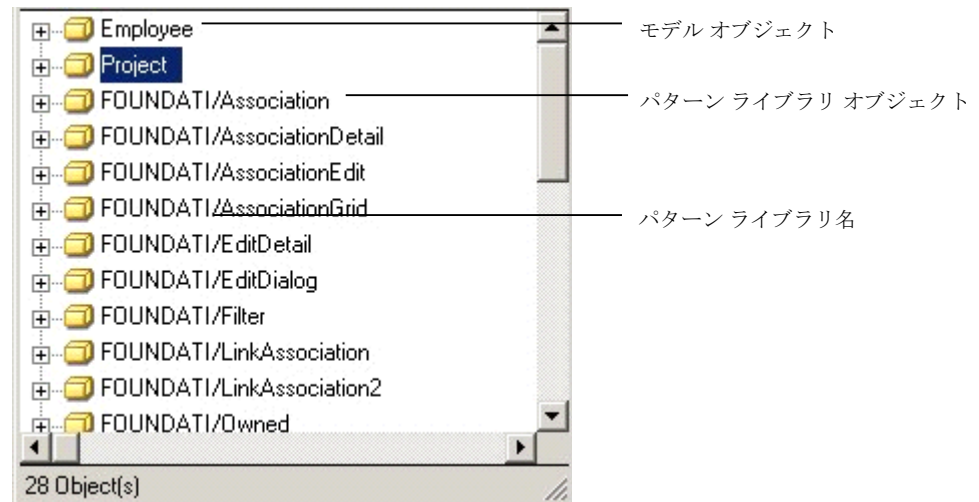
デフォルトでは、一度にオブジェクト ブラウザで表示できるオブジェクトは、**1** タイプです。メイン オブジェクト タイプによってスコープされている他のタイプのオブジェクトがある場合は、それらも表示されます。詳細については、本章の「スコープの詳細」を参照してください。

オブジェクト ブラウザには、エンティティ、フィールド、およびファンクションを表示するショートカット ボタンが付いています。ファンクションを表示するようにオブジェクト ブラウザを設定すると、スコープされないファンクションだけが表示されます。他のオブジェクトにスコープされたファンクション（エンティティやビューなど）は、オブジェクト ブラウザがそのオブジェクト タイプにフォーカスしている場合に表示されます。

表示するオブジェクト タイプをオブジェクト リストから選択することもできます。

[表示/非表示ライブラリ オブジェクト] ボタンをクリックしてライブラリ オブジェクトを表示できます。

オブジェクトがパターン ライブラリ内にあるとわかりますが、これはデフォルトでライブラリ名がオブジェクト名の前に付くからです。たとえば、以下の図の 3 番目のエンティティは「FOUNDATI/Association」ですが、これは「Association」というエンティティが「FOUNDATION」というパターン ライブラリの中にあるという意味です。



「Project」エンティティとトリプル

CA Plex では、トリプルを使ってオブジェクトを定義し、オブジェクト間の関係を指定します。その名が示すとおり、トリプルは、主語オブジェクト、動詞、目的語オブジェクトの 3 つの部分で構成されています。

たとえば、本章では以下のトリプルを使って「Project」エンティティの一意の識別子を定義します。

Project **known by FLD** Project ID

この例では、「Project」（エンティティ）がトリプルの主語オブジェクト、「**known by**」（または「**known by FLD**」）が動詞、「Project ID」（フィールド）が目的語オブジェクトになります。

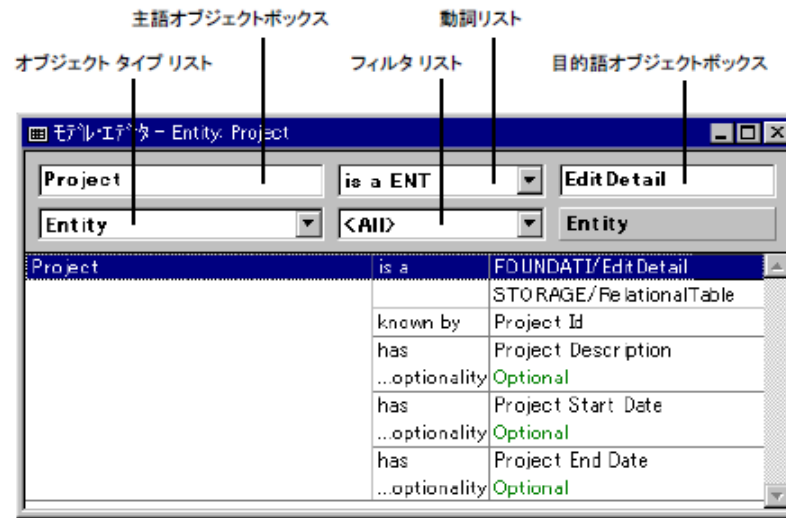
CA Plex のマニュアルでは、動詞は常に太字で印字され、ほとんどの場合、主語と目的語のオブジェクト タイプを伴います（たとえば、ENT **known by FLD** となります）。

トリプルは、CA Plex に情報を入力するためのシンプルで容易に理解できる言語です。トリプルの例をさらに以下に示します。これらは、本章で入力するものです。

- Project **has FLD** Project Description
- Project Description **is a FLD** LongDescription
- Project **described by DGM** Project Diagram

このサンプル アプリケーションでは、トリプルを入力し、「Project」エンティティのフィールド、すなわちプロジェクトの識別子、説明、開始日、終了日を定義します。

モデル エディタを使って、トリプルを表示、追加、編集、削除します。次の図は、本章を終了した時点でのモデル エディタの状態を表しています。



「Project」エンティティの属性の指定

次の一連の手順では、モデルに以下のトリプルを追加します。

Project *known by* Project ID
 Project *has* Project Description
 Project *has* Project Start Date
 Project *has* Project End Date

「Project」エンティティの属性の指定

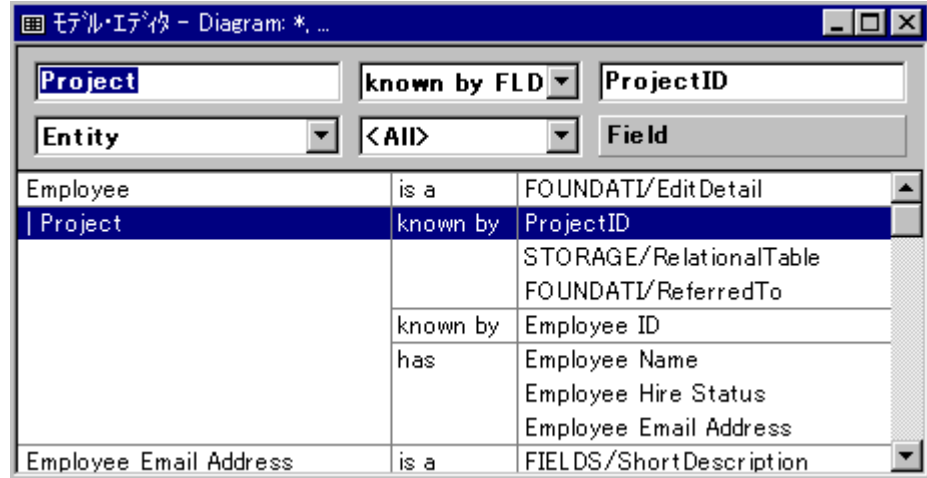
1. [ツール] メニューから [モデル エディタ] をクリックします。

モデル エディタが表示されます。すでにいくつかのトリプルがモデル エディタに入力されているのがわかります。

2. オブジェクト タイプ リストから「Entity」を選択します。
3. 主語オブジェクト ボックスに「**Project**」と入力します。プロジェクトについてまだ何も指定していませんが、名前を入力すると、CA Plex ではそのエンティティをモデルに追加します。
4. 動詞リストから「**known by FLD**」を選択します。

注: リストをスクロールする代わりに、動詞リストをクリックし、使用したい動詞が現れるまで最初の文字のキーを押して表示させることもできます。known by FLD 動詞を選択するには、K キーを 1 回押します。

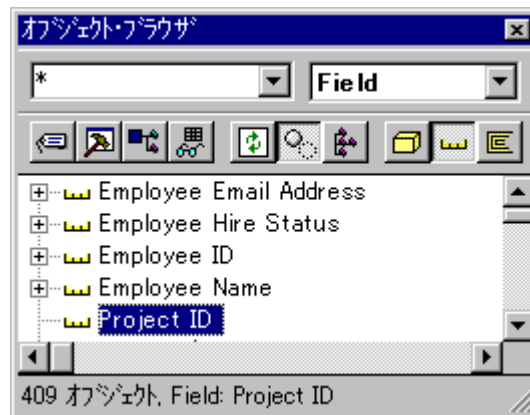
5. 目的語オブジェクト ボックスに「Project ID」と入力して、Enter キーを押します。
モデル エディタは、下の図のように表示されます。



これで「Project known by Project ID」トリプルがされました。このトリプルは「Project」エンティティの主キーを定義します。

6. オブジェクト ブラウザが開いていなければ、[表示/非表示オブジェクト ブラウザ] ツールバー ボタンをクリックします。
7. オブジェクト ブラウザの [フィールド] ボタンをクリックして、フィールド オブジェクトを表示します。

先程追加した、「Project ID」フィールドに注目してください。ローカル モデルで定義するオブジェクトはすべてオブジェクト ブラウザの先頭に表示されます。パターン ライブラリ オブジェクトは、ローカル オブジェクトの下に表示されます。



すでに定義されている他のフィールドがあることがわかります。たとえば、「Employee Email Address」や「Employee Hire Status」などがあります。時間を節約するため、「Employee」エンティティとそのフィールドは、作業中のモデルですでに定義されています。

「Project」はまだモデル エディタの主語オブジェクト フィールドに表示されているはずです。

8. 動詞リストから「**has FLD**」を選択します。
9. 目的語オブジェクト ボックスに「**Project Description**」と入力します。
10. Enter キーを押します。

「Project has Project Description」というトリプルを作成しましたが、これは「Project」エンティティの「Project Description」フィールドを定義します。このフィールドを使ってプロジェクトの記述を格納します。このトリプル「**ENT has FLD**」は、非キー属性を作成します。非キー属性の値は、各エンティティに対して必ずしも一意である必要はありません。ですから、たとえば「Project Description」フィールドに同じテキストがあるプロジェクトが複数存在する場合があります。

11. 8 から 10 までの手順を繰り返して、以下のトリプルを作成します。

Project **has** Project Start Date
Project **has** Project End Date

12. オブジェクト ブラウザの [リフレッシュ] ツールバー ボタンをクリックします。
オブジェクト ブラウザに新規フィールドが表示されます。



フィールド継承

継承を使ってフィールドのプロパティを定義します。CA Plex では、継承を「is a」動詞を使用して簡単に定義できます。

次の一連の手順では、以下のトリプルを入力します。

- Project ID *is a* FIELDS/Identifier
- Project Description *is a* FIELDS/LongDescription
- Project Start Date *is a* DATE/CheckedDateISO
- Project End Date *is a* DATE/CheckedDateISO

FIELDS と DATE という接頭辞は、対象のオブジェクトがそれぞれ、FIELDS と DATE というパターン ライブラリに属していることを示します。

前のセクションで定義したフィールドには、文字と日付という、2 種類の異なるデータタイプが存在します。それはフィールドの種類が異なることを意味します。「Project ID」フィールドはプロジェクトを一意に識別するコードを、「Project Description」フィールドはテキストを、「Project Start Date」と「Project EndDate」は日付を保持します。今の時点では、モデルに「Project」エンティティに属するフィールドが存在するだけで、格納するデータ タイプに関する情報はありません。

継承は、あるオブジェクトが別のより一般的（抽象的）なオブジェクトのプロパティを取得できるようにするメカニズムです。パターン ライブラリ フィールドを継承することで、アプリケーションは以下を行うことができるようになります。

- フィールドに入力したデータの妥当性検査（たとえば、エンド ユーザーが間違って 2 月 31 日と入力してしまうのを防ぎます。）
- データの画面への適切な表示（Windows の設定に合わせた日付の表示など）
- データのデータベースへの適切な格納（「Project Description」フィールド用にデータベースにテキスト フィールドを作成し、「Project Start Date」フィールドおよび「Project End Date」フィールド用に日付フィールドを作成します。）

継承を使用したフィールド プロパティの定義

エンティティにフィールド プロパティを定義することができます。プロパティは、フィールドがそれぞれ格納するデータのタイプに関する情報を提供します。

「Project」フィールドのプロパティの定義

1. [表示/非表示ライブラリ オブジェクト] ボタンをクリックし、ライブラリ オブジェクト を表示します。
2. オブジェクト ブラウザでフィールドがフォーカスされ、フィールド オブジェクトが表示されるように [フィールド] ボタンをクリックします。

- オブジェクト ブラウザで名前（名前の左のアイコンではありません）をクリックして、「Project ID」フィールドを選択し、このフィールドをオブジェクト ブラウザからモデル エディタの主語オブジェクト ボックスにドラッグします。

これでモデル エディタの主語オブジェクト タイプは「Field」に変わり、動詞リストはフィールド用に定義できる動詞のみを表示するように変わります。

注: オブジェクトをドラッグする際に、カーソルは閉じた小包の形のアイコンに変化します。オブジェクトをドロップする位置の上まで来ると、アイコンは開いた小包の形に変わります。

次の表は、閉じた小包と開いた小包の形のアイコンを示したものです。

閉じた小包の形のアイコン	開いた小包の形のアイコン
	

- 動詞リストから、「is a FLD」を選択します。
- オブジェクト ブラウザから、ライブラリ オブジェクト「FIELDS/Identifier」を目的語オブジェクト ボックスにドラッグし、Enter キーを押します。

注: 「FIELDS/Identifier」フィールドを見つけるには、オブジェクト ブラウザを下へスクロールする必要があります。オブジェクト ブラウザの先頭のフィルタ ボックスを使用すると、ライブラリ項目の一部だけを表示することができます。この場合、「FIELDS/Identifier」だけを表示するので、「*Identifier*」と入力します。作業終了後、忘れずにフィルタの設定を「*」に戻してください。

これで、トリプル「Project ID is a FIELDS/Identifier」を作成しました。

- ［オブジェクト ブラウザ］上の［リフレッシュ］ボタンをクリックします。
「Project ID」フィールドの左にプラス（+）記号が付いていて、現在スコープされたオブジェクトがあることを示しています。
- プラス記号をクリックして、フィールドを展開します。



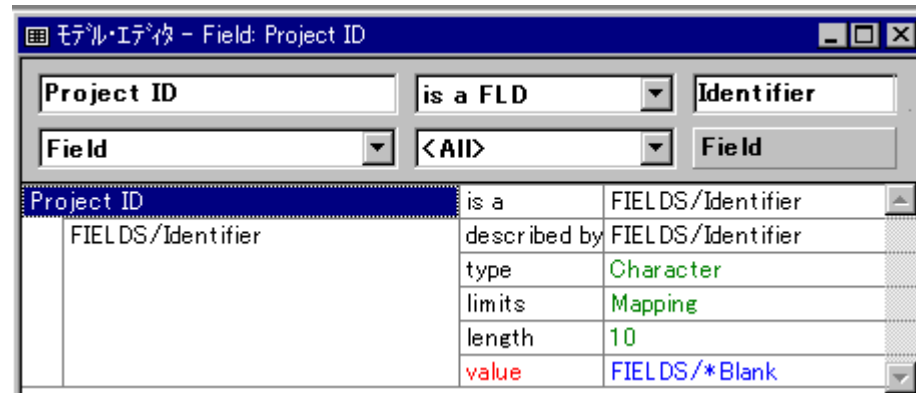
「Project ID」が「*Blank」という値 (Value) を持ったことが確認できますが、「FIELDS/Identifier」から他に何が継承されたのかはこれだけではわかりません。Value は CA Plex のオブジェクト タイプの 1 つです。

8. 「Project ID」フィールドをオブジェクト ブラウザからモデル エディタのボディにドラッグします。ボディはエディタの最下部で、ここにはトリプルがすべて表示されます。

1 つ以上のオブジェクトをモデル エディタのボディにドラッグすると、それらのオブジェクトに定義されたトリプルだけが表示されます。これを、モデル エディタ「フォーカス」といいます。「Project ID」フィールドをモデル エディタにドラッグすると、モデル エディタがこのフィールドにフォーカスし、トリプル「Project ID is a FIELDS/Identifier」を表示します。しかし、これだけではまだ多くの情報は得られません。

9. オブジェクトが継承元オブジェクトから何を継承したかを詳しく調べるには、[2 レベル] ツールバー ボタンをクリックします。

モデル エディタは、もう一段深いレベルの詳細を表示します。



これで「Project ID」は（手順 7 で確認している）値「*Blank」に加えて、文字データ タイプとフィールド長 10 を継承していることが確認できます。

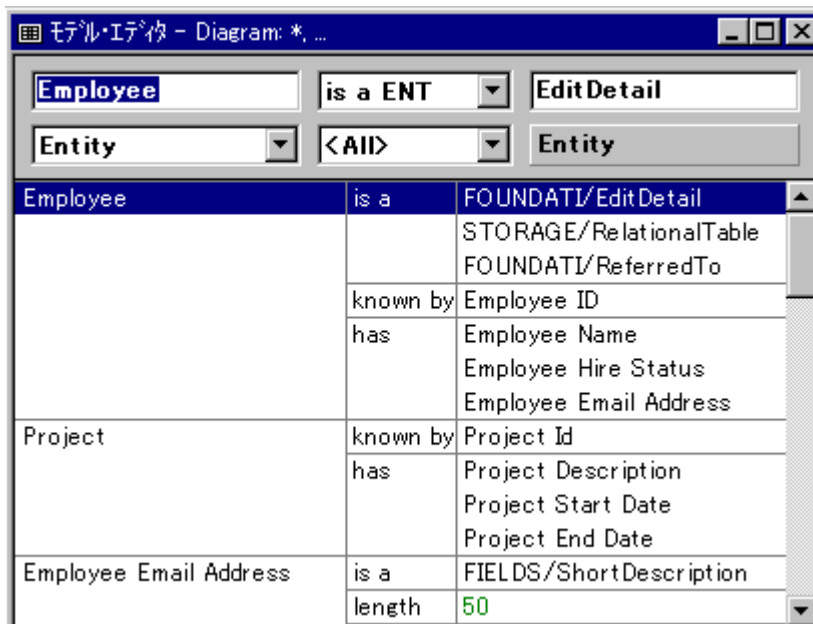
注: 手順 5 で、ライブラリ オブジェクト「FIELDS/Identifier」を、オブジェクト ブラウザからモデル エディタの目的語オブジェクト ボックスにドラッグしましたが、オブジェクト名を直接目的語オブジェクト ボックスに（ライブラリ名を付けず）入力しても結果は同じです。手順 5 では、「Identifier」と入力することも可能です。

重要: 誤ったオブジェクト名を入力すると、その誤った名前前で新しいオブジェクトが作成されてしまうおそれがあります。このような場合は、オブジェクト ブラウザで誤りのあるオブジェクトを見つけて削除し、[削除] ダイアログ ボックスの[連鎖的に削除] チェック ボックスを有効にしてください。

10. [1 レベル] ツールバー ボタンをクリックし、モデル エディタが 1 レベルの情報を表示するように設定します。

11. [フォーカス解除] ボタンをクリックして、モデル エディタの表示をリセットします。

表示をリセットすると、モデル内のトリプルが再びすべて表示されます。



注： 上図よりも多くのトリプルがモデル エディタに表示された場合、モデルはライブラリ オブジェクトを表示するように設定されています。その場合、[ライブラリ オブジェクトの表示/非表示] ツールバー ボタンをクリックします。

12. 「Project Description」フィールドをモデル エディタのボディから主語オブジェクト ボックスにドラッグします。このフィールドは、「Project has Project Description」トリプルの 3 番目のカラムにあります。
13. 目的語オブジェクト ボックスに「LongDescription」と入力し、Enter キーを押します。

モデル エディタがトリプルが次のように表示します。

Project Description is a FIELDS/LongDescription

これはパターン ライブラリ フィールド名の綴りが正しかったことを示しています。

注： モデルの新規オブジェクトを作成し、それがたまたまライブラリ オブジェクトと名前が同じ場合は、そのオブジェクトの名前を変更するか、（作成するつもりがなかった場合）削除しなければなりません。

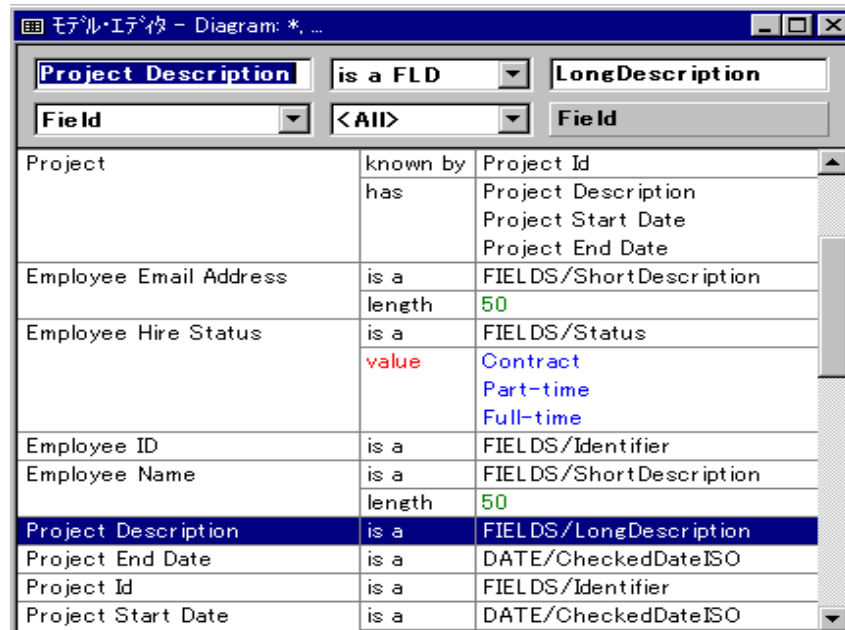
14. 手順 3 から 5 または手順 12 から 13 を繰り返して、以下のトリプルを作成します。

Project Start Date is a DATE/CheckedDateISO

Project End Date is a DATE/CheckedDateISO

15. [リフレッシュ] ツールバー ボタンをクリックします（オブジェクト ブラウザではなく、メイン ツールバーのボタン）。

モデル エディタは下の図のように表示されるはずです。



「DATE/CheckedDateISO」を継承すると、フィールドにエンド ユーザが有効な日付を入力することを確実にする機能が追加されます。

16. 手順 8 と手順 9 で説明されているプロセスを使って、これらのフィールドがパターン ライブラリ フィールドから継承している特性を調べます。

継続トリプル

ここでは、トリプル構文の拡張機能である、継続トリプルを紹介します。継続トリプルは他のトリプルとほとんど同じですが、主語オブジェクトそのものがトリプルであるという点が他と異なります。

次の一連の手順では、以下の継続トリプルを入力して特定のフィールドが任意入力であることを指定します。

Project **has** Project Description
...**optionality** Optional

Project **has** Project Start Date
...**optionality** Optional

Project **has** Project End Date
...**optionality** Optional

本章の後半で他の継続トリプルも入力します。

フィールドを任意にするために継続トリプルを使用

エンド ユーザがデータを入力する場合、通常すべてのフィールドにデータを入力することはありません。開発者は、いくつかのフィールドを必須入力に、一方で他のフィールドを任意入力に指定することができます。任意を指定しない場合、デフォルトの設定ではフィールドは必須になります。

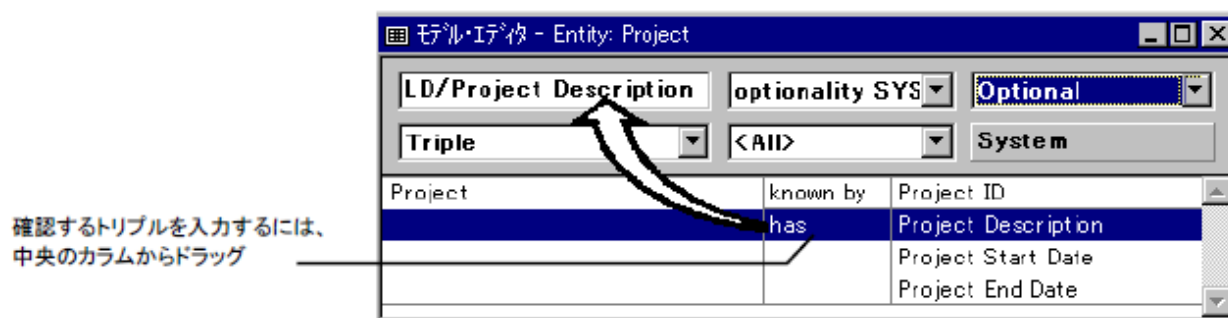
エンド ユーザが必須フィールドを空白にしたまま、ダイアログを閉じようとする、メッセージ ダイアログ表示され、空白の必須フィールドへのデータの入力を促します。入力が済むまでは、ダイアログを閉じることはできません。この処理はパターン ライブラリの機能として定義されているもので、CA Plex にハードコードされているものではありません。したがって、必要に応じて変更することができます。

フィールドを任意にするために継続トリプルを使用

1. オブジェクト ブラウザにエンティティを表示させるために、[エンティティ] ツールバー ボタンをクリックします。
2. オブジェクト ブラウザで「Project」エンティティを選択し、[検査] ツールバー ボタンをクリックします。

これにより、モデル エディタは「Project」エンティティにフォーカスし、そのエンティティを定義するトリプルだけを表示します。

3. トリプル「Project has Project Description」の中央をクリックして、選択します。
4. このトリプルをモデル エディタの主語オブジェクト（左上）フィールドにドラッグします。



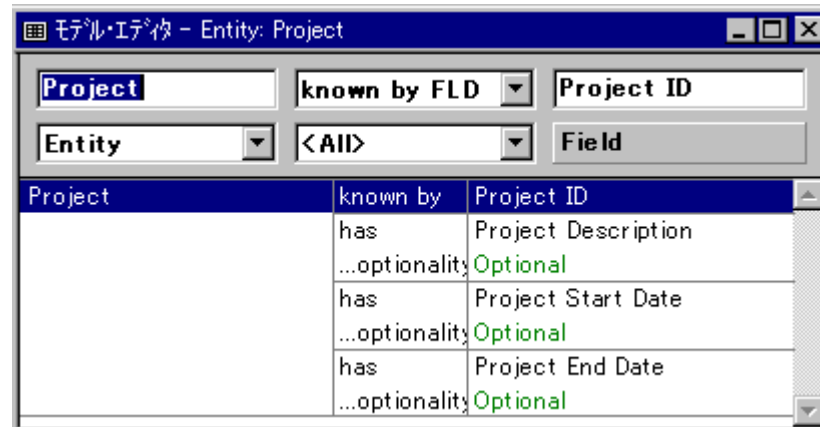
5. 動詞（中央上部）フィールドから「optionality SYS」を選択し、目的語オブジェクト（右上）フィールドから「Optional」を選択します。
6. Enter キーを押します。

継続トリプルが入力されました。

Project **has** Project Description
...optionality triples Optional

注: 上記トリプルが表示されない場合、[1 レベル] ツールバー ボタンをクリックし、モデル エディタが 1 レベルの情報を表示するように設定します。

7. 同じ手順で、「Project Start Date」フィールドと「Project End Date」フィールドを任意にします。
8. モデル エディタをリフレッシュします。
モデル エディタは下図のようになります。



パターンと継承

継承は、CA Plex でアプリケーションを設計するときの主要な再利用手段です。あらかじめ構築されテストされたライブラリ パターンを継承することにより、生産性と品質を大幅に向上できるという利点があります。

パターンとは、再利用およびカスタマイズできるように設計された CA Plex オブジェクトのグループです。本章では、付属のパターン ライブラリから継承します。更なるパターン ライブラリを他社から入手することもできます。

パターンとテンプレート 他製品のツールに付属しているテンプレートやフレーム ワークと比較するとパターンの利点が見えてきます。一般的にテンプレートは、定義済みのソリューションをコピーする手段です。継承は、コピーと同じではありません。オブジェクトが CA Plex のパターンを継承する場合、その関係は動的なものになります。つまり、パターンを変更すると、そのパターンを継承しているオブジェクトもすべて変更されるということです。

また、パターンはツールにハードコードされているものではなく、CA Plex を用いて作成されるという点にも留意してください。付属のパターンから継承した自分専用のパターンを作成することで、パターン ライブラリをカスタマイズすることができます。あるいは、専用のパターンを新規に作成することもできます。

パターンとコンポーネント パターンとコンポーネント（COM や EJB コンポーネントなど）は、互いに補足しあう関係にある技術です。CA Plex のパターンを使用すれば、コンポーネントのグループと組み合わせて実装することができます。コンポーネントはよく家の建築ブロックにたとえられます。このたとえをパターンに当てはめると、パターンは家の青写真すなわち設計図と考えることができます。

パターンと同様、コンポーネントもソフトウェアをできるだけ再利用できるようにするものです。コンポーネントは変更できませんが、パターンはその内部構造を特定の場所を変更することが可能です。もう 1 つの違いは、パターンが純粋な設計上の構造体であるのに対し、コンポーネントは実行アプリケーションの一部を構成するという点です。

継承を使用した「Project」エンティティの定義

「Project」エンティティにデータを格納するフィールドが定義され、フィールドの継承元 パターン ライブラリ フィールドが指定されました。次の手順では、「Project」エンティティにユーザ インターフェース、およびデータベースとやりとりを行う機能を追加します。

再び継承を使って、プロジェクト エンティティのための機能を定義します。2 つのトリプルを追加するだけで定義できます。

Project is a FOUNDATI/EditDetail
Project is a STORAGE/RelationalTable

最初の継承トリプルで、そのエンティティのためのユーザ インターフェースを表示、処理する機能を得ることができます。2 番目の継承トリプルで、リレーショナル データベースへの「Project」レコードの読み込み、書き出し機能が提供されます。

以下の手順を、順を追って実行すれば、これらの 2 つのトリプルの入力結果を容易に理解できるようになります。

「Project」エンティティへの機能の追加

継承を使用して、データベースとやりとりを行うために「Project」エンティティにユーザ インターフェースと機能を追加することができます。

「Project」エンティティへの機能の追加

1. オブジェクト ブラウザにエンティティが表示されていない場合は、[エンティティ] ツールバー ボタンをクリックします。

注:「Project」エンティティの左にプラス記号がないことに注意してください。これは、このエンティティにスコープするオブジェクトがないという意味です。

上記のトリプルを追加するには、モデル エディタの主語オブジェクト タイプを「Entity」に設定する必要があります。オブジェクト タイプは直接変更できますが、オブジェクト ブラウザからオブジェクトをドラッグすることで、モデル エディタはオブジェクト タイプをセットし、そのオブジェクトのすべてのトリプルを表示します。

2. 「Project」をオブジェクト ブラウザからモデル エディタの主語オブジェクトボックスにドラッグします。

これは、[検査] ツールバー ボタンを使用するのと同様です。つまり、モデル エディタを「Project」エンティティに関連するトリプルだけを表示するようにし、オブジェクト タイプを変更（まだ「Entity」にセットされていないものと仮定）、主語オブジェクト ボックスに「Project」をセットします。

3. 動詞リストから「is a ENT」を選択します。

注: この手順で使用する「is a」動詞は、「Project」のフィールド用に継承を指定するのに使用した動詞とは異なります。同名の動詞がいくつかありますが、主語オブジェクトと目的語オブジェクトが異なります。CA Plex は、目的語オブジェクトに一致した動詞だけ選択できるようにします（この手順では「is a ENT」動詞）。

「is a」動詞のタイプの詳細については、オンライン ヘルプのキーワードで「is a」を検索してください。

4. 目的語オブジェクト ボックスに「EditDetail」と入力し、Enter キーを押します。

トリプル「Project is a FOUNDATI/EditDetail」が作成されました。これは、「Project」が「FOUNDATION」パターン ライブラリの「EditDetail」パターンの構造と機能を継承していることを示します。ライブラリ オブジェクトが表示されていることを確認し（[表示/非表示ライブラリ オブジェクト] ボタンをクリック）、下にスクロールして、オブジェクト ブラウザ内の「FOUNDATION/EditDetail」パターンを見つけることができます。

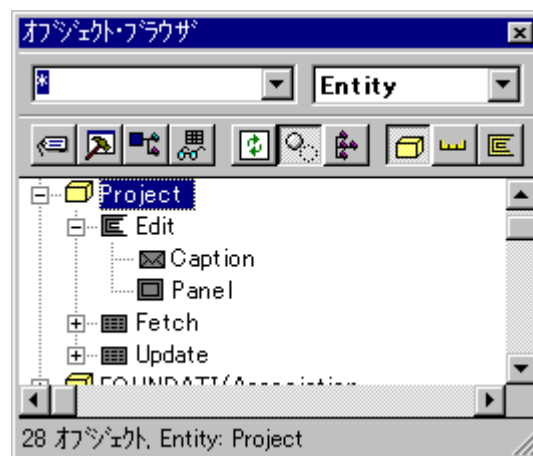
「EditDetail」パターンの詳細については、オブジェクト ブラウザ内でこのパターンを選択し、Shift+F1 キーを押します。

5. オブジェクト ブラウザ内で [リフレッシュ] ツールバー ボタンをクリックします。

「EditDetail」を継承することで「Project」エンティティにスコープしたオブジェクトがいくつか作成されています。

スコープされたオブジェクトの詳細については、「スコープの詳細」を参照してください。

6. エンティティ アイコンの左のプラス記号をクリックして「Project」を展開し、次に「Edit」ファンクションの隣のプラス (+) 記号をクリックして展開します。



「Project」はスコープされたパネルとキャプションを持つ 1 つのファンクション「Edit」と 2 つのビュー「Fetch」と「Update」を継承しました。これらのオブジェクトは「Project」にユーザ インターフェースを追加し、データベースへのデータの格納や取得を可能にします。

特に以下の作業が可能です。

「Edit」ファンクションはそれにスコープされたパネルを表示します。

「Caption」オブジェクトと「Panel」オブジェクトは、「Edit」ファンクションによってスコープされており、「Edit Projects」パネルのレイアウトが格納されています。このパネルが実際どのようなものかについては、本章で後述する

「「Project.Edit」ファンクションの実行」にある生成されたアプリケーションの図を参照してください。

「Fetch」ビューと「Update」ビューはデータベース レコードを読み書きするファンクションをスコープします。

次に、「Project」エンティティが情報を格納する方法を指定します。アプリケーションはリレーショナル データベースを使用しますので、データベース テーブルを作成、保守するプロセスが必要です。この機能を実行するために、「STORAGE/RelationalTable」から継承するように「Project」を設定します。

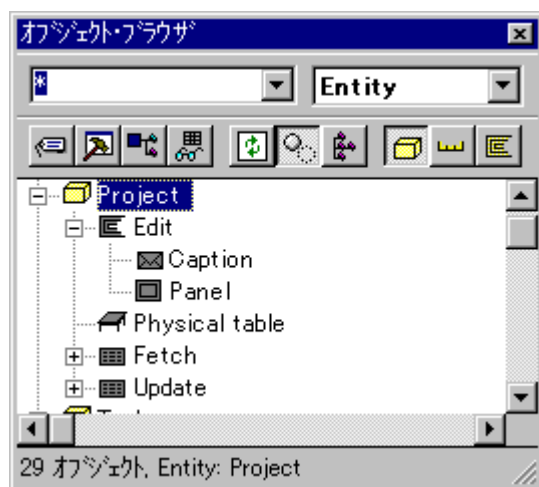
モデル エディタの主語オブジェクト ボックスにまだ「Project」があり、動詞リストに「is a ENT」が表示されているはずです。

7. 「STORAGE/RelationalTable」をオブジェクト ブラウザから目的語オブジェクト ボックスまでドラッグして、以下のトリプルを作成します。

Project is a STORAGE/RelationalTable

8. [リフレッシュ] ボタンをクリックしてして、モデル エディタで「Project」エンティティ用に定義したトリプルをすべて表示します。
9. オブジェクト ブラウザ上の [リフレッシュ] ボタンをクリックします。

オブジェクト ブラウザはこのように見えているでしょう。

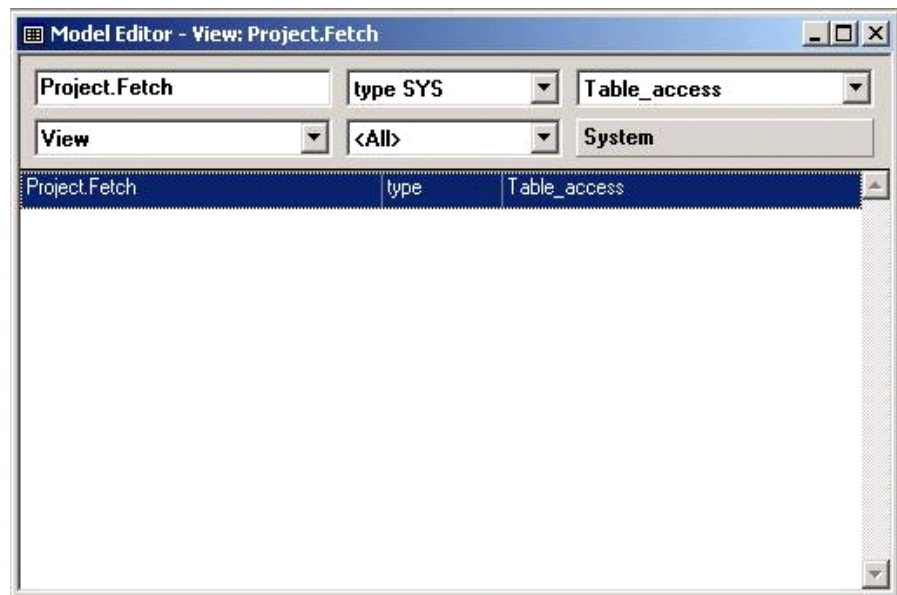


「Project」エンティティは、「STORAGE/RelationalTable」から「Physical Table」というオブジェクトを継承しています。これで、アプリケーション生成時にデータベース内に作成されるテーブルを定義します。「Project」エンティティ用に定義したフィールドは、このテーブルに格納されます。

Microsoft Access は、ビューをサポートしません。しかし ODBC ドライバは、SQL CREATE VIEW ステートメントをサポートし、Microsoft Access にクエリを作成します。

つまり、CA Plex でビューを生成、構築すると、Microsoft Access にクエリが作成されるということです。Microsoft テクニカル サポートは未公開機能としてこの状況を説明しています。CA Plex でのその機能の使用は避けてください。そのかわり、すべてのビューに VW type SYS = Table Access のトリプルを追加し、ビューが実装されないようにしてください。

10. ビューの **Project.Fetch** を、オブジェクト ブラウザからモデル エディタの主語オブジェクト ボックスまで、ドラッグします。
11. 動詞リストから「**type SYS**」を選択します。
12. 目的語オブジェクト ボックスの Table-access を選択して、Enter を押します。



13. ビューの Project.Update に同じトリプルを作成します。

スコープの詳細

スコープされたオブジェクトとは、他のオブジェクトに属するオブジェクトです。スコープされたオブジェクトは、スコープするオブジェクトとは別々に存在することはできません。たとえば、ファンクションが削除されると、ファンクションによってスコープされるパネルは削除されます。

スコープされないタイプのオブジェクトがあります。このタイプは、モデル内の他のすべてのオブジェクトに依存しないで存在するオブジェクトです。たとえば、通常、エンティティはスコープされません。逆に、テーブルは常にエンティティにスコープされます。テーブル オブジェクトを作成するには、所属するエンティティを指定しなければなりません。

スコープされたオブジェクトを参照するときには、ほとんどの場合、あいまいさをなくするため、そのフル ネームを使用しなければなりません。たとえば、「Edit」というファンクションをスコープする「Project」というエンティティがあるとします。このファンクションのフル ネームは「Project.Edit」で、これにより、「Employee.Edit」などのこのモデルに既に存在する他の「Edit」ファンクションと区別されます。

前出の手順は、パターンから継承するほとんどの場合で、スコープされたオブジェクトが作成されることを示したものです。スコープされたオブジェクトは、手動で作成することもできます。

Java のための生成構築オプションのセットアップ

本書の「Java コンポーネント」章で説明されている Oracle JDK 7.0 をインストールしていると、[生成と構築] オプションが正しく初期化されており、次の「生成と構築」セクションに進むことができます。

Java のための生成構築オプションのセットアップ

1. [ツール] メニューの [生成と構築] を選択します。
2. [構築] メニューの [生成と構築オプション] を選択します。
生成構築オプションのダイアログが表示されます。
3. [システム定義] を選択して、右上の **Local System** をダブルクリックします。
システム プロパティ ダイアログが表示されます。
4. [Java 構築] オプションを選択します。
[Java 構築] オプションのダイアログが表示されます。
5. Java ランチャー (jdk\bin 内の java.exe あるいは javaw.exe) が存在するディレクトリを [表示] ボタンをクリックし、選択するか、パスを直接入力してください。

6. OK をクリックします。

生成構築オプションのダイアグラムが表示されます。

7. OK をクリックします。

生成と構築のダイアログが閉じます。

生成と構築

これで「Project」エンティティのフィールドを定義し、これらのフィールドのプロパティを指定しました。すでに「Project」エンティティの機能も定義し、基本的なユーザ インターフェースとデータベースの読み書き機能も備えています。これで、「Project」エンティティを生成構築する準備が整いました。生成構築とは CA Plex がモデル情報をソース コードに展開し(生成)、次に生成されたソース コードをコンパイルされたファイルにする(構築)プロセスです。

モデル内のオブジェクトを生成構築した後、プログラムを実行して、作成した内容を確認できます。

「Project」エンティティの生成と構築

1. [ツール] メニューから [生成と構築] を選択するか、[新しい生成と構築] ツールバー ボタンをクリックします。

[生成と構築] ウィンドウが表示されます。ウィンドウを開くと、メッセージ ログが現れます。

注: Java アプリケーションを生成と構築するために、生成構築オプションで Java のための設定をしなければなりません。

2. [オプション] メニューから、[メッセージ ログを開かない] をクリックして、ウィンドウを最小化します。

これは、新しいメッセージを受け取る度に、メッセージ ログがポップアップしないようにします。

3. [生成と構築] ウィンドウにライブラリ オブジェクトが表示されている場合は、[ライブラリ オブジェクトの表示/非表示] ツールバー ボタンをクリックして、それらを非表示にします。

これで、[生成と構築] ウィンドウに「Project」エンティティと「Employee」エンティティが表示されます（「Employee」は、事前に モデルに追加されています）。



4. 「Project」エンティティを選択します。

5. [生成と構築] ツールバー ボタンをクリックします。

「Project」が展開され、その下の生成可能なオブジェクトがすべて強調表示されます。スコープされたオブジェクトがすべて選択されるわけではありません。

[生成の確認] ダイアログが表示され、生成されるオブジェクトの数が表示されます。

6. [はい] をクリックします。

CA Plex は、これらのオブジェクトを生成し、次に生成状況を表示します。

注: 2つの警告メッセージが生成と構築中に出力されますが、これは予期されたものであり、問題ではありません。

詳細については、本章の「メッセージ ログ」を参照してください。

7. 生成プロセスが終了すると、[キャンセル] ボタンが [OK] ボタンに変わります。
[OK] ボタンをクリックして、[生成状況] ダイアログを閉じます。

CA Plex は、オブジェクトをコンパイルと構築するためのプロンプトを表示します。

8. 両方のプロンプトに対して、[はい] をクリックします。

最初の Java コンパイル ダイアログは、データベース テーブルおよびビューを構築するためのものです。次の Java コンパイル ダイアログは、Java ファンクション構築のためのものです。

CA Plex は新しい Shell Build ウィンドウを開き、Apache ANT と Java コンパイラを使用して生成されたコードを構築します。

同時に、データベース テーブルが、このサンプル アプリケーションで使用される ODBC データ ソースに送られます。セットアップを容易にするため、CA Plex のインストール時に、Microsoft Jet データベースのためのデータソースを自動的に作成しています。

メッセージ「BUILD SUCCESSFUL」が Shell Build ウィンドウに表示されると、Java 構築が完了です。Shell Build ウィンドウを閉じることができます。

ネイティブ プラットフォームの実装

CA Plex には、サポートされるすべてのプラットフォームで生成とコンパイルを行うための完全な環境が用意されています。Java 実装については、いかに Apache ANT とシームレスに統合されているかをご確認いただけます。

CA Plex は、Java SE プラットフォームに基づく Java コードを生成します。モデル中の各ファンクションは、Java クラスのセットとして実装され、1つまたは複数の JAR ファイルにパッケージされます。

C# でアプリケーションを実装することを選択した場合、CA Plex は Microsoft の .NET Framework に基づいた C# コードを生成します。同様に、ネイティブの RPG および DDS コードの生成し、IBM System i 上のアプリケーションを実装することもできます。

メッセージ ログ

生成と構築中に、2つの警告が「メッセージ ログ」に出力されますが、これは予期されたものであり、問題ではありません。その他の警告やエラー メッセージが表示された場合は、問題が発生していることを示します。最も可能性の高い原因は、必要なトリプルのすべてがモデルに入力されていないということです。この場合は、前に戻って指示どおりであるかどうかをチェックしてください。

CA Plex が正しくインストールされていない場合にもエラーは発生します。

- 2つの警告メッセージ (BLD9081) は、各ビューのソースが ODBC データ ソースに投入されなかったことを示しています。これは、予期されたもので、このサンプルアプリケーションで使用する Microsoft Office Access データベースにビューを実装しないように、CA Plex を設定したために出力されています。

Java 構築エラーのチェック

Java 構築エラーのチェック

1. 「Java 構築結果の表示」 ツールバー ボタンをクリックして構築結果を表示し、エラーのないことを確認します。

このサンプル アプリケーションでは、構築結果を確認するように促されることはありませんが、アプリケーションが正常に動作しない場合は、エラーがないかどうか確認してください。

2. 「保存」 ツールバー ボタンをクリックして、モデルを保存します。

生成されたアプリケーション

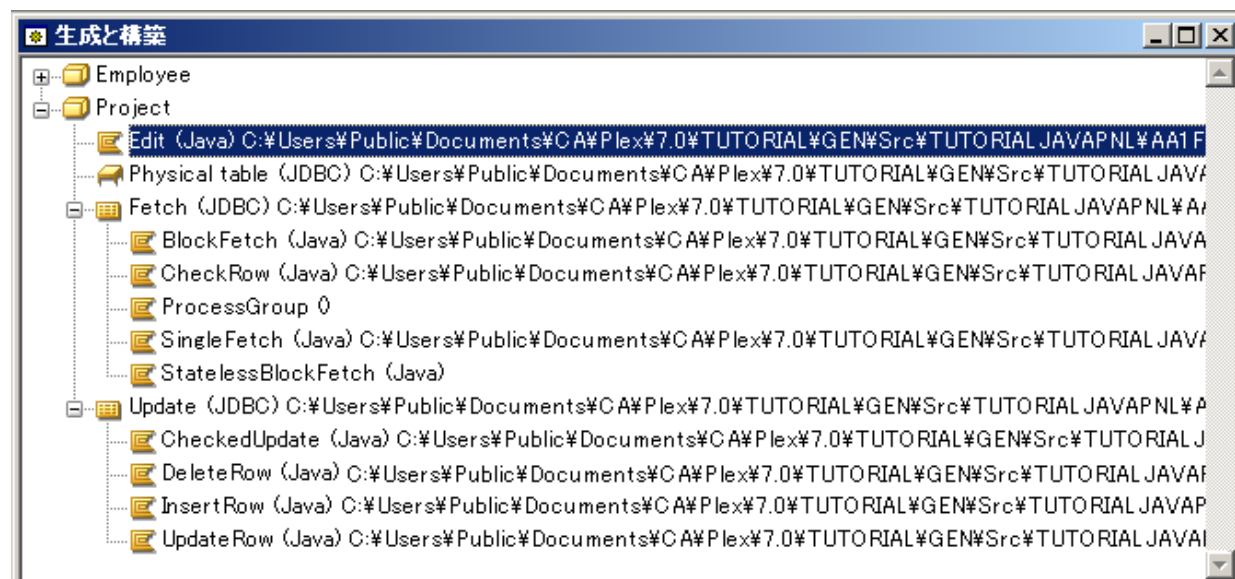
いくつかのトリプルを入力するだけで、フル機能を備えたアプリケーション（「EditDetail」パターンのインスタンス）が作成されました。生成されたアプリケーションを使って、プロジェクトの作成、変更、削除が可能です。各プロジェクトの記述を作成し、開始日と終了日を示すことができます。

「Project.Edit」ファクションの実行

フル機能を備えたアプリケーションが作成されています。プロジェクトを作成、変更、削除するためのアプリケーションを実行できます。

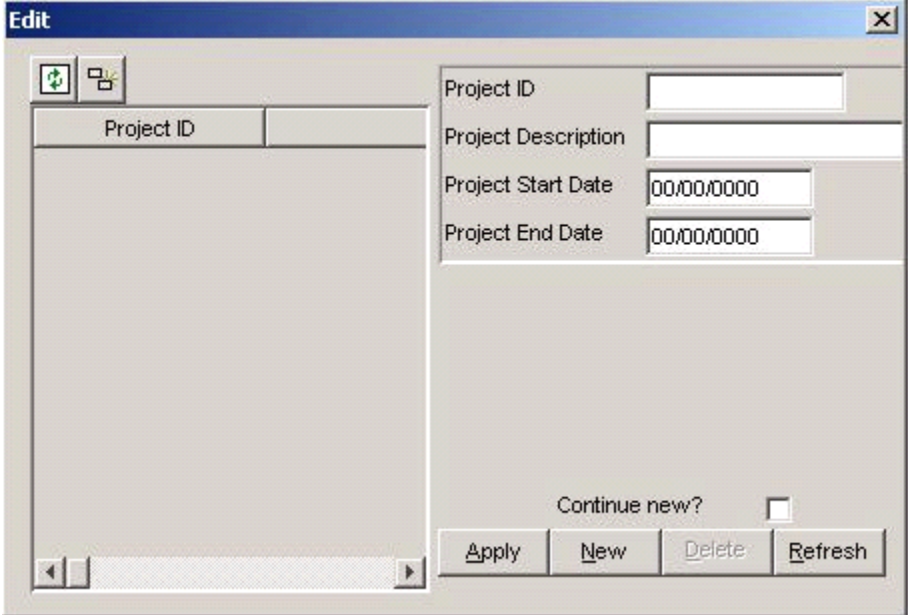
「Project.Edit」ファクションの実行

1. [生成と構築] ウィンドウで、「Project」エンティティの下の「Edit」ファンクションを選択します。



2. 「実行」 ツールバー ボタンをクリックします。

生成されたアプリケーションが起動します。次のように表示されます。



このダイアログを開いたままにしておきます。次の手順で、プロジェクトをデータベースに追加する方法について説明します。

デフォルトのパネル レイアウト

CA Plex は、ユーザがこれまでに入力した設計の詳細に基づいて、自動的に正しいフィールドをパネルに追加し、デフォルトのレイアウトを提供します。このデフォルトのレイアウトは後で調整します。このパネルの大まかな外観（左側にグリッドがあり右側にフィールドがある）と機能は、継承するパターン ライブラリによって決まります。この外観と動作のほとんどは上書きできます。

新規プロジェクトの追加

作成したアプリケーションを使用して、新規プロジェクトを追加します。

新規プロジェクトの追加

1. 先の手順で既に表示されている「Edit」ダイアログに次の値を入力します。

注: 日付のフォーマットは、Windows コントロール パネルの地域の設定によって異なります。

フィールド	入力
Project ID	Proj01
Project Description	Temporary description.
Project Start Date	12/10/2006
Project End Date	01/01/2007

現時点では、フィールド枠がパネル枠の右端を超えてしまっているため「Project Description」フィールドに「Temporary description」とだけ入力します。それより長く入力したとしても、入力内容を確認できません。「Project Description」は任意の属性として定義されているので、空白のままにすることもできます。

データ入力終了後、「Project Description」がパネル枠の右端からはみ出さないようにパネル レイアウトを変更します。

2. 「Apply」ボタンをクリックします。

値はデータベースに記録され、ウィンドウの左側のグリッド領域に新規プロジェクトが表示されます。

3. 「Copy the data」ツールバー ボタンをクリックすると、新しいプロジェクトが作成され、その中に Proj01 のデータが挿入されます。次の 2 つのプロジェクトをデータベースに追加します。

フィールド	入力
Project ID	Proj02
Project Description	Temporary description.
Project Start Date	03/01/2006
Project End Date	04/04/2006

フィールド	入力
Project ID	Proj03
Project Description	Temporary description.
Project Start Date	05/05/2006
Project End Date	06/05/2006

- ダイアログの右上の「閉じる」ウィンドウ ボタンをクリックして、アプリケーションを終了します。

注: パターン ライブラリから継承したパネルには、どれにも「キャンセル」ボタンがありません。「閉じる」ウィンドウ ボタンを押すと、未確定の操作はすべて破棄されます。「Apply」をクリックした場合は、ウィンドウが閉じる前に変更内容が更新されます。

100 %のコード生成

以上で、13 のトリプルをモデルに入力しました。このトリプルから、CA Plex は、8,000 行を超える Java および SQL コードを生成し、完全に機能するアプリケーションを作成しています。つまり、すべてのコードを生成しているということです。

どうぞご自由に試してみてください。パターン ライブラリから継承した参照整合性チェック機能も確認してください。たとえば、複数のレコードに対して同じ「Project ID」を入力できません。

データの保存

デフォルトでは、アプリケーションを構築するたびに、CA Plex はデータベース内のテーブルなど、「生成と構築」ウィンドウで選択されたすべてオブジェクトを再構築します。データベース テーブルを再構築すると、テーブル内のデータはすべて消去されるので、ローカル モデルをデフォルトの設定のままにしておくと、次の再構築時に、入力したばかりのデータがすべて失われます。

このデータの消失は、テーブルに対して「TBL **implement** SYS No」トリプルを入力することで防げます。これは、テーブルがスコープされているエンティティ（この場合は「Project」）を次回再構築される際に、テーブルが再構築されるのを防ぎます。このトリプルは、少なくとも 1 度テーブルを構築した後に設定します。

データの保存と、「Project」エンティティ テーブルの再生成の防止

- オブジェクト ブラウザで、「Project.Physical Table」を選択し、それをモデル エディタの主語オブジェクト ボックスまでドラッグします。
- 動詞リストから、「**implement** SYS」を選択します。
- 目的語オブジェクト リストから、値「No」を選択します。
- Enter キーを押します。

5. モデルを保存します。

注： フィールドの追加などテーブルに影響する変更をエンティティに加える場合は、**implement SYS** 値を「Yes」に戻し、テーブルを再生成する必要があります。テーブル内のデータはすべて失われます。再構築後に入力したデータを保存する場合は、**implement SYS** 値を必ず「No」にリセットしてください。

パネル デザイナ

「Project.Edit」ファンクションを実行したときにおわかりのように、ダイアログの外観はすべてが適切な状態とはいえません。「Project Description」フィールドがパネルの右端からはみ出しており、2、3 の単語を入力するだけのスペースしかありませんでした。

パネル デザイナにより、「Project.Edit」が表示するパネルを変更できます。「Project Description」フィールドのサイズを変更し、それに複数行編集コントロールを追加することができます（つまり複数行のテキストを入力できるようになります）。スピン コントロールを、日付フィールドに追加することもできます。

ビジュアル開発

すでにおわかりのように、CA Plex アプリケーション設計の出発点は、通常のビジュアル開発ツールとは異なり、設計モデルであり、画面レイアウトではありません。これは、大規模なアプリケーション開発を成功させるためには、設計とモデリングが最も重要な要素となるからです。ただし、ビジュアル プログラミング ツールと同様、CA Plex にも、グラフィカル ユーザ インターフェース (GUI) を設計するための使いやすいエディタが用意されています。これには、ネイティブの GUI コントロールの豊富なセットのほかに、他社のコンポーネント (Windows C++ では ActiveX コントロール、Java では JavaBeans) を使用できる機能も含まれています。エディタにはもう 1 つのモードがあり、このモードでは、System i 5250 アプリケーションのキャラクタベースの画面を同様に設計することができます。

パネル デザイナのオープン

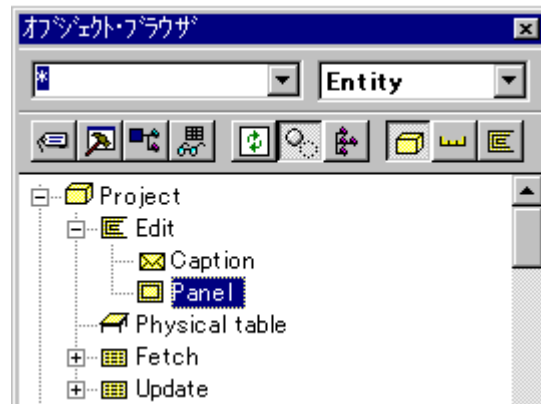
パネル デザイナでパネルの外観を変更できます。それは 3 つの個別のウィンドウ (デザイン ウィンドウ、パネル パレットおよびプロパティ シート) で構成されています。

パネル デザイナのオープン

1. オブジェクト ブラウザにエンティティが表示されていない場合は、[エンティティ] ツールバー ボタンをクリックします。

注： このボタンがツールバー上にはないことに注意してください。このボタンは、オブジェクト ブラウザ上にあります。オブジェクト ブラウザとツールバーの両方にあるボタンもいくつかあります。

2. 「Project」エンティティが表示されていない場合は、[リフレッシュ] ツールバー ボタンをクリックします。
3. 「Project」エンティティ、「Edit」ファンクションを展開し、次の図のように「Panel」を選択します。



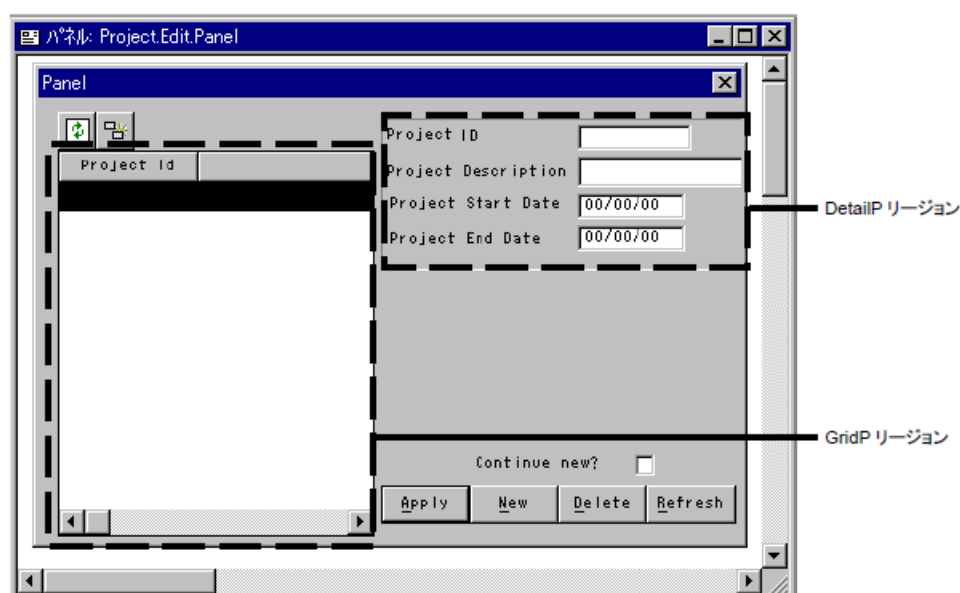
4. [エディタ] ツールバー ボタンをクリックします。

パネル デザイナが表示されます。任意のオブジェクトを選択して、[エディタ] ボタンをクリックすることができます。CA Plex は、常に適切なエディタを開きます。

デザイン ウィンドウ

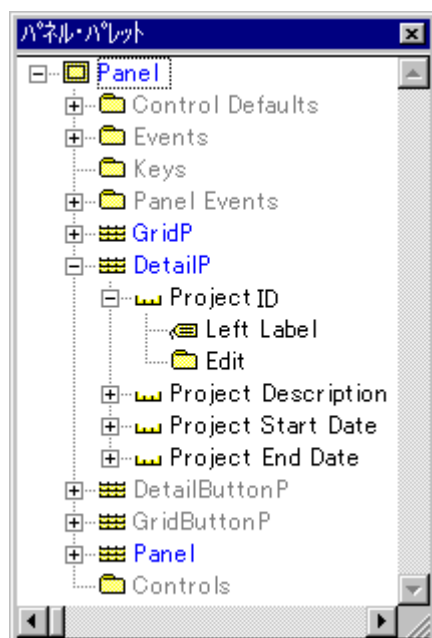
デザイン ウィンドウはパネル デザイナのメイン ウィンドウで、パネルの外観を表示します。パネルのプロパティを変更する際は、このウィンドウでその変更によりパネルの外観がどのようになるのかを視覚的に確認できます。このウィンドウを使用して、ボタン、フィールド、その他のユーザ インターフェース 要素を選択、移動、サイズ変更することができます。他のウィンドウを使って、パネルとその要素の外観に変更を加えると、その変更はこのウィンドウにも表示されます。

パネルの要素は、リージョンにグループ分けされます（各リージョンには名前が付いています）。次の図は、「Project.Edit」によって表示された GridP リージョンと DetailP リージョンを示しています。



パネル パレット

「ウィンドウ」メニューから「パネル パレット」を選ぶことにより、パネル パレットを開くことができます。パネル パレットは、フィールド、ラベル、ボタンなどパネルのすべての要素を表示し、それらの要素はフォルダ（）とリージョン（）にグループ分けされています。フォルダやリージョンを展開し、含まれている要素を確認することができます。目に見える要素のほとんどはリージョンに含まれています。次の図のパネル パレットでは、このパネルの 5 つのリージョン（GridP、DetailP、GridButtonP、DetailButtonP、Panel）が確認できます。



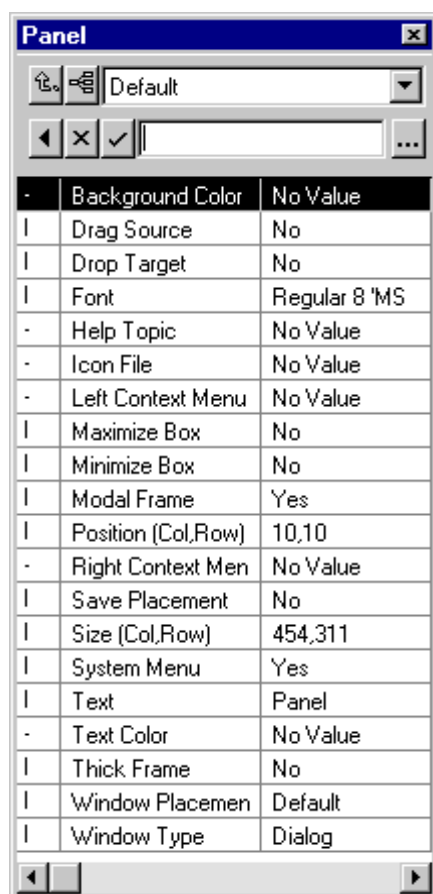
リージョンが展開されると、そのリージョンに含まれる要素が表示されます。たとえば「DetailP」リージョンを展開すると、「Project ID」、「Project Description」、「Project Start Date」、「Project End Date」というフィールドが表示されます。

パネルに表示される場合、通常フィールドは 2 つ以上のパーツを持っており、ほとんどの場合 1 つのラベルと 1 つのコントロールを含んでいます。上図では「Project ID」フィールドが「Left Label」ラベルと「Edit」コントロールを表示していることが確認できます。ラベルは、左ラベル（Left Label）、上ラベル（Top Label）、右ラベル（Right Label）かが示されます。ラベル タイプは、それがコントロールの左右（左/右 ラベル）、あるいはカラム見出し（上ラベル）として表示されることを示しています。コントロールはエンド ユーザが情報をやりとりするものです。コントロールの設定により、それを編集ボックス、リスト ボックス、コンボ ボックスなどとして表示することができます。

プロパティ シート

パネル上の各要素は、サイズ、色、位置など要素に関連した一連のプロパティを持っています。プロパティ シートは、現在選択されている要素のプロパティを表示します。要素が何も選択されていない場合は、プロパティ シートはパネル全体のプロパティを表示します。

その要素について変更できる内容を確認するには、デザイン ウィンドウまたはパネル パレットで要素を選択し、プロパティ シート上でそのプロパティを確認してください。表示されるプロパティは、選択される要素の種類によって異なります。



たとえば、ボタンにタブ ストップ (Tab Stop) プロパティを設定して、タブ順に含ませるかどうかを指定できますが、フィールド ラベルはタブ順に含まれることはないので、ラベルを選択してもプロパティ シートにこのプロパティは表示されません。

複数行編集コントロールの定義

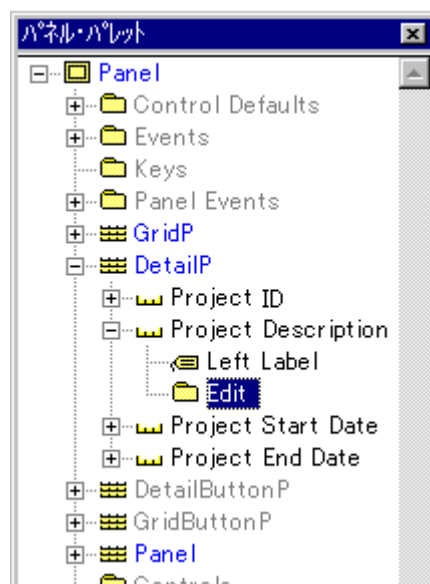
テキストを複数行で表示させ、ダイアログの端からはみ出さないように「Project.Edit」によって表示されるパネルを変更します。

Project.Edit パネルの変更と複数行編集コントロールの指定

1. パネル デザイナのウィンドウ同士が重ならないように位置を調整し、各ウィンドウの内容に対して作業を行えるようにします。

要素に変更を加えるには、まずパネル上で選択する必要があります。要素だけを選択し、要素の入っているリージョンを選択しないようにするには、パネル パレットを使用します。

2. まだ展開していない場合は、パネルを展開してこの「Edit」パネルのリージョンを表示します。
3. 「DetailP」リージョンを展開し、次に「Project Description」フィールドを展開します。



4. 「Edit」コントロールを選択します。

プロパティ シートの値は、編集ボックスで変更可能なプロパティを示します。また、デザイン ウィンドウで、ラベル「Project Description」の隣の編集ボックスだけが選択されていることを確認してください。

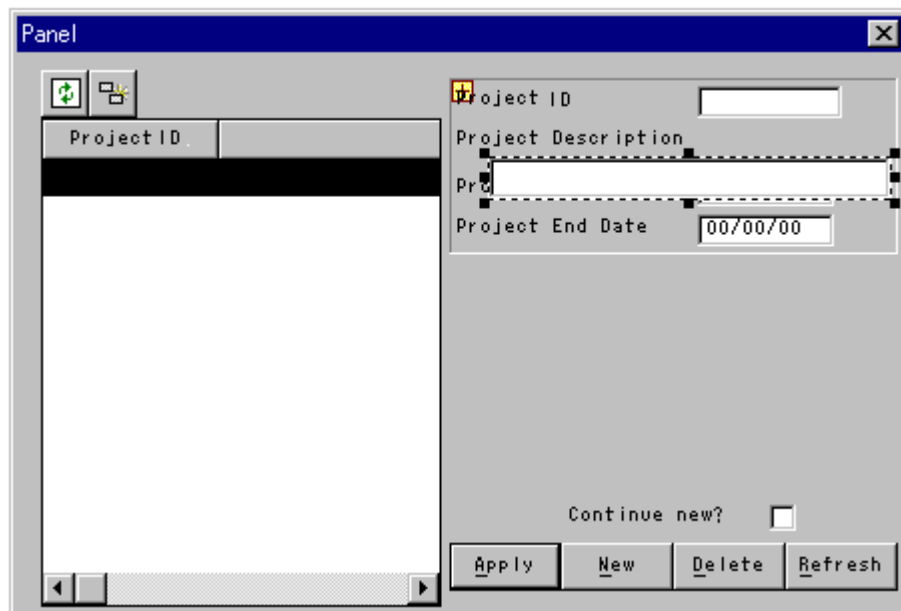
5. デザイン ウィンドウで、ラベルの下に編集コントロールをドラッグし、その左端が「Project」の「j」にほぼ揃うようにします。「Project Start Date」フィールドに重なることは気にしないでください。これは後で修正します。

プロパティ シートの位置 (Position (Col, Row)) プロパティは、要素を動かすと変更されます。変更を微調整する場合は、プロパティ シートで正確なピクセル位置を指定することができます。

フィールドは、まだ、パネルの右端を超えています。

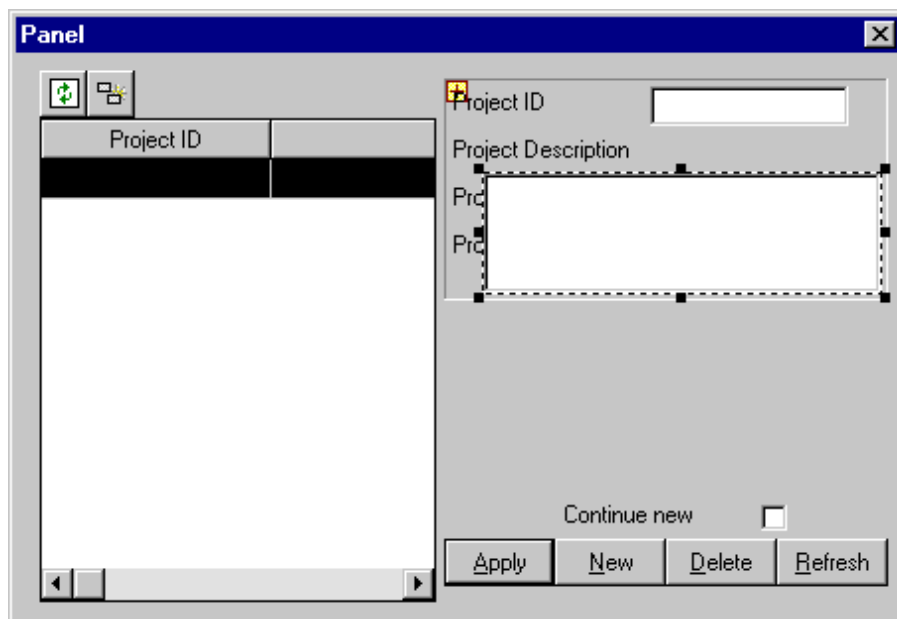
6. プロパティ シートで、サイズ (Size (Col, Row)) プロパティをクリックします。
7. プロパティ シート上部の編集域で、値を 200,20 に変更し、次にチェック マーク ボタンをクリックします。

編集コントロールとそのフィールドの若干外側を取り囲む縁取り枠が、パネル内に収まる大きさにサイズが変わります。



8. 編集ボックスの底辺中央のハンドルをドラッグして、ボックスを縦に伸ばします。高さをこれまでの約 3 倍にします。

パネルは、次の図のように見えるはずです。



これで、「Project Description」編集ボックスの位置とサイズを変更しました。次に 2 つのプロパティを設定しますが、これによりエンド ユーザは編集ボックスに 2 行以上のテキストを入力できるようになります。

9. プロパティ シートで、「Multiline」プロパティをダブル クリックし、その値を「No」から「Yes」に変更します。
10. プロパティ シートの「Scrolling」プロパティを選択します。

注: 「Scrolling」プロパティが表示されない場合は、「Multiline」プロパティが「No」から「Yes」に変更になっていることを確認してください。

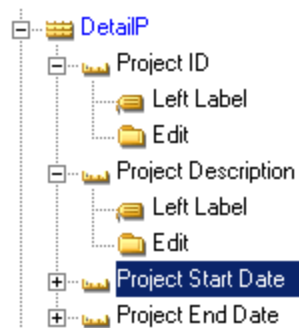
プロパティ シートの上部にある入力域が、リストから選択できるように変わることにご注意ください。値をタイプすることはできません。

11. リストから「Vertical」を選択します。

「Scrolling」プロパティを「Vertical」に設定すると、カーソルが右端まで来ると、複数行編集ボックスのテキストが次の行に折り返されます。このオプションを設定しないと、テキストは次の行に折り返されず、最初の行の後ろに続いてフィールドの右端を超えてしまいます。

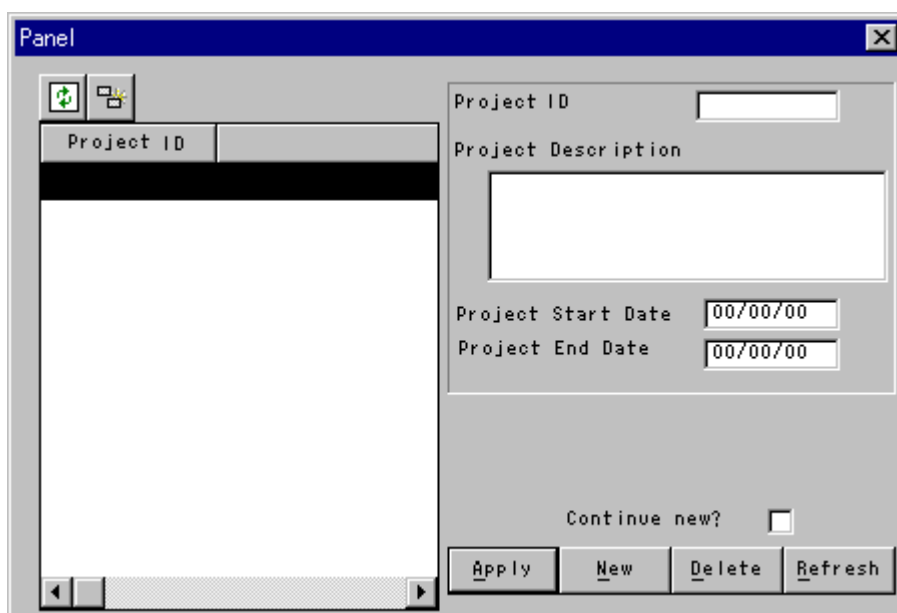
「Project Description」編集ボックスのサイズと形状を変更すると、「Project Start Date」と「Project End Date」日付フィールドを覆い被します。次に、これら 2 つのフィールドを下に移動します。

12. パネル パレットで、「DetailP」リージョンの「Project Start Date」フィールドを、展開せずに選択します。



13. Ctrl キーを押したまま、「Project End Date」フィールドを選択します。これで、両方のフィールドが選択され、デザイン ウィンドウで確認することができます。
14. デザイン ウィンドウで、フィールドをドラッグして、「Project Description」複数行編集ボックスの下まで持ってきます。

デザイン ウィンドウで、フィールドのまわりの縁取り枠のサイズが変更になり、すべてのフィールドのまわりにフィットするようになります。

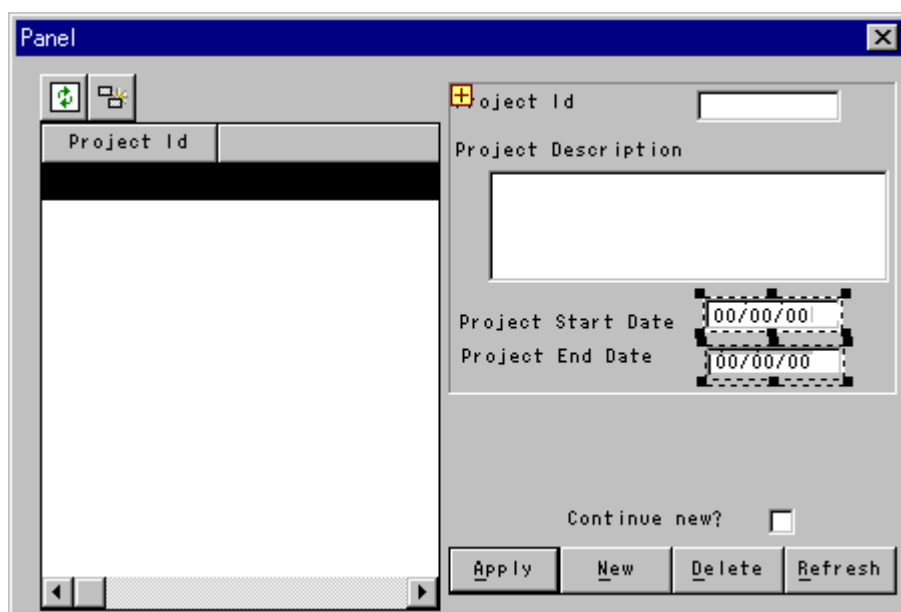


スピン コントロールの追加

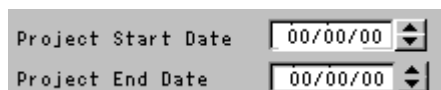
スピン コントロールは、エンドユーザに、エディット ボックスの中で値をタイプする代わりに矢印ボタンのクリックにより、フィールドの値を増加あるいは減少させます。

次の手順では、スピン コントロールが含まれるよう 2 つの日付フィールドを修正します。

1. デザイン ウィンドウで、Ctrl キーを押したまま、日付フィールドの編集コントロールの 1 つをクリックします。Ctrl キーを押したままにすると、リージョンではなくて、リージョンに含まれるコントロールが選択されます。
2. Ctrl キーと Shift キーを同時に押したまま、他の日付コントロールをクリックします。これで、両方の日付コントロールが選択され、デザイン ウィンドウは以下の図のように表示されます。



3. プロパティ シートで「Spin Control」プロパティを探し、これを「Yes」に設定します。



パネル上で複数の要素を選択し、そのすべてについてプロパティを変更することもできます。

4. [ファイル] メニューから [保存] を選択します。
5. デザイン ウィンドウを閉じます。

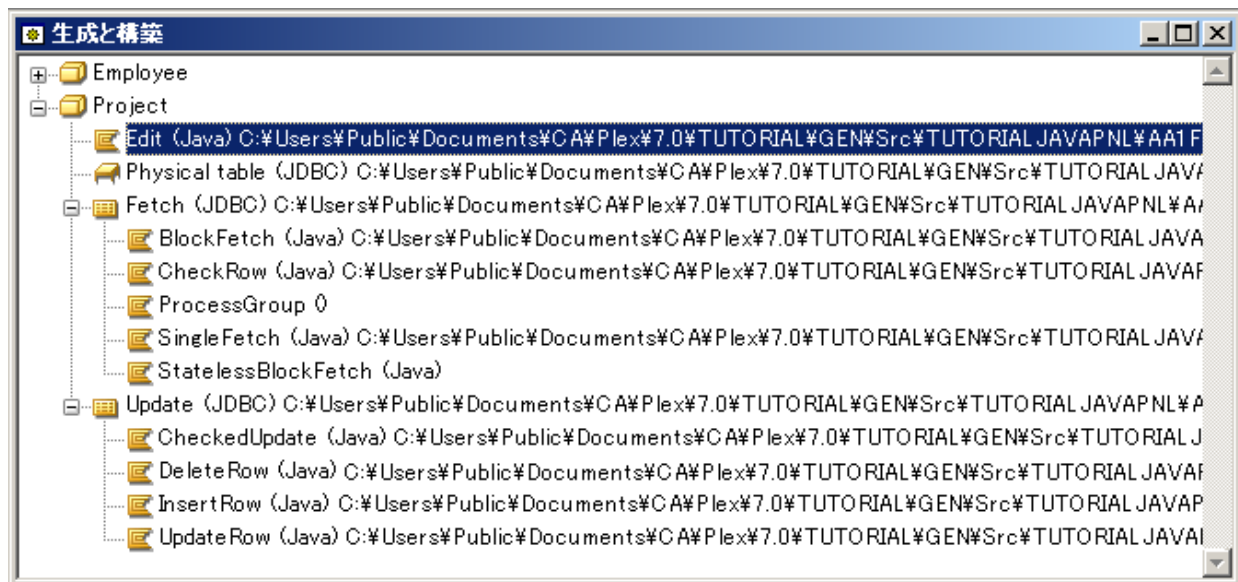
これで、パネル デザイナ全体が閉じます。

ファンクションの再生成

これでパネルの変更が終わりましたが、パネルを表示するファンクションの再生成と再構築を行わないと、変更内容を確認できません。

「Project.Edit」の再生成と再構築

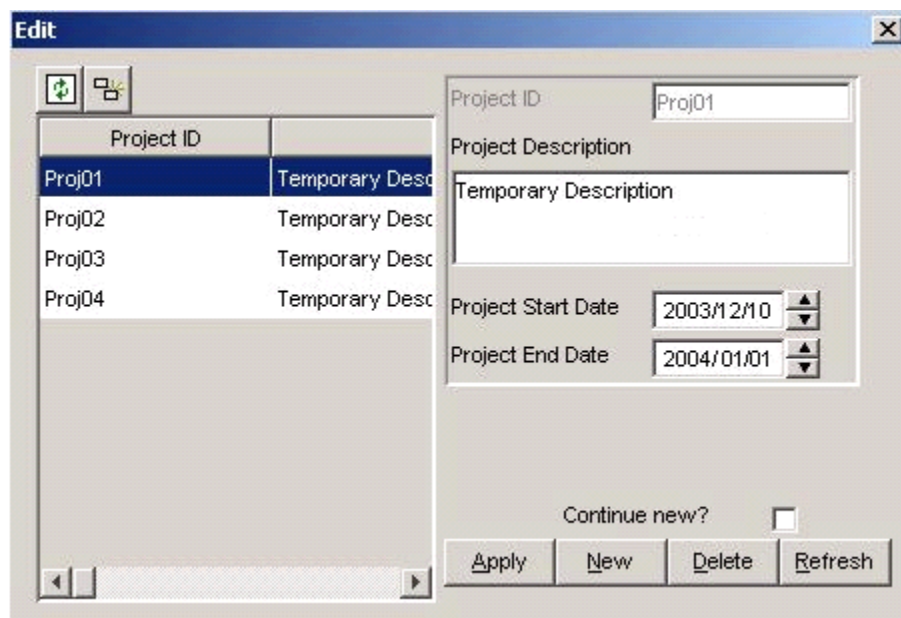
1. 「生成と構築」ウィンドウが開いたままであれば、そのウィンドウに切り替えます。または、「新しい生成と構築」ウィンドウ ツールバー ボタンをクリックします。
2. 「Project.Edit」を選択します。



パネルを再構築するには、これをスコープするファンクションを再生成および再構築します。変更したのは「Project.Edit.Panel」オブジェクトだけなので、「Project.Edit」だけを再生成および再構築するだけです。加えた変更がスコープされたテーブルまたはその他のオブジェクトに影響する場合は、それらのオブジェクトも同様に再生成および再構築する必要があります。

3. 「生成と構築」 ツールバー ボタンをクリックします。
4. 生成を促すプロンプトが表示されたら、「はい」をクリックし、構築を促すプロンプトが表示されたら、再度「はい」をクリックします。
5. 生成および構築プロセスが完了したら、「生成と構築」ウィンドウ内で「Project.Edit」を選択し、「実行」 ツールバー ボタンをクリックします。

変更されたダイアログが現れ、そこには入力済みのデータが表示されています。



6. Proj01 の「Project Description」フィールドのテキストを選択し、それを長いテキストで置き換えます。例を次に示します。

Create a database application for the McCready account. Contact is Jim Hauser, 415-555-3146.

7. [Apply] をクリックします。

グリッド リージョンと詳細リージョンの双方で記述が変更されている点に注意してください。

8. Proj02 と Proj03 を選択し、上記のプロセスを繰り返してその記述を以下に変更します。

Proj02: Create an email client for internal support staff.

Proj03: Convert the ODBC version of the McCready application to a System i version.

9. アプリケーションを閉じます。

復習

本章で学習した内容は、以下のとおりです。

- モデル エディタを使用してトリプルを追加する方法について学習。
- フィールドを定義し、データ アクセスおよびユーザ インターフェース ファンクションを作成するための、パターン ライブラリ オブジェクトの継承。
- 「Project」 エンティティのデータベース テーブルと継承ファンクションの生成と構築、およびいくつかのサンプル データの追加。
- パネル デザイナを使用した継承パネル レイアウトの変更。
- 変更内容を確認するための、影響を受けるオブジェクトの再生成と再構築。

本章では、次のパターン（およびそのスコープされたオブジェクト）をご紹介します。

パターン	説明
 FIELDS/Identifier	10 文字のデータ フィールド。
 FIELDS/ShortDescription	20 文字の文字フィールド。
 DATE/CheckedDateISO	ISO の日付標準形式で日付データが格納される日付フィールド。入力された日付が有効かどうか検査します。
 FOUNDATION/EditDetail	単一ダイアログ内で、データベース レコードをリスト表示し、レコードの編集、追加、変更、削除ができます。

本章では、次のトリプルをご紹介します。

トリプル	説明
ENT known by FLD	エンティティの主キーを定義します。
ENT has FLD ... optionality SYS	エンティティのデータ属性の定義、およびフィールドへの値の入力が必須か任意かを定義します。
ENT is a ENT	主語エンティティは、目的語エンティティの特性を継承します。
FLD is a FLD	主語フィールドは、目的語フィールドの特性を継承します。
TBL implement SYS	データベース テーブルを生成、構築するかどうかを指定します。

第 4 章: グループ モデル ライセンス

グループ モデル ライセンスは、独立系ソフトウェア ベンダ (ISV) がグループ モデルの無許可使用を防ぐためのオプション機能です。CA とビジネス パートナは、販売の際にどのグループ モデルのライセンスが必要かをお知らせします。

ライセンスが必要なグループ モデルを初めて参照したり開く際に、ライセンスが必要なグループ モデルを示すライセンス用ダイアログ ボックスが表示されます。ライセンスが必要なグループ モデルにはライセンス コードが必要です。

ライセンスが必要なグループ モデルのライセンスでトライアル ライセンスが有効になっていると、一定の期間だけ有効なトライアル ライセンスを使用できます。この期間が終了する前に、ライセンス コードを請求して入力しなければなりません。トライアル ライセンスが有効な場合ライセンス コードを待っている間でもすぐにグループ モデルを使用できます。この機能が有効でない場合は、ライセンス コードを受け取るまでグループ モデルは使用できません。ライセンス コードを受け取ったら、[グループ モデル ライセンス] ダイアログに入力します。

[グループ モデル ライセンス] ダイアログから[ライセンス]ボタンで表示されるダイアログの電子メールや FAX 機能を使ってライセンス コードを請求できます。

ライセンス コードの取得元に関する情報は、ライセンスされたグループ モデルに格納されています。電子メールと FAX 機能を使うと、この情報が自動的に表示されます。[ヘルプ] メニューの [グループ モデル バージョン情報] を選択し、[ライセンス] ボタンを押しても、同じ情報が表示されます。

ライセンスされたグループ モデルのライセンス用ダイアログ ボックスへのアクセス方法については、「[グループ モデル ライセンス] ダイアログ ボックスへのアクセス」を参照してください。必要に応じて、この手順にしたがって、グループ モデルのライセンスを保守します。

ローカル モデル ライセンス

ベンダは、グループ モデル ライセンスのほかに、ライセンスされたグループ モデルを使用するローカル モデルも保護することができます。これは、ライセンスされたグループ モデルのオブジェクトを使用しているローカル モデルを開くたびに、ライセンス チェックが実行されることを意味します。このようなローカル モデルを使用するためには、ワークステーション上に有効なライセンス コードが必要です。

[グループ モデル ライセンス]ダイアログへのアクセス

ライセンスされたグループ モデルのライセンス コードを入力するには、[グループ モデル ライセンス] ダイアログを使用します。

[グループ・モデル・ライセンス]ダイアログへのアクセス

1. CA Plex を開始します。
2. ホスト グループ モデルにログインします。
3. [グループ モデル] ウィンドウで、ライセンスされたグループ モデルを選択し、[ヘルプ] メニューから [グループ モデル バージョン情報] を選択します。
4. [ライセンス] をクリックします。

ライセンスされたグループ モデルがまだ有効でない場合は、このモデルが初めて開かれたときに、このダイアログが自動的に表示されます。

グループ モデル ライセンスの転送

グループ モデル ライセンスは、安全性が高く、特定の PC に固有のものです。複数の PC で同じライセンス コードを使用することはできません。その代わり、グループ モデル ライセンスを別のワークステーションに転送することができます。ネットワークを通じて直接転送することも、フロッピー ディスクを使って間接的に転送することも可能です。

直接転送

ネットワーク上の別の PC に存在する、ライセンスのないグループ モデルにライセンスを転送する場合、直接転送を行います。

直接転送を行います。

1. 転送先の PC で、転送先のグループ モデル（ライセンスのないバージョン）に最低 1 回アクセスし、ライセンス ディレクトリを作成します。ライセンスを転送する前に、CA Plex が閉じていることを確認してください。

重要: 転送先の CA Plex が開いたままの状態では絶対にライセンスを転送しないでください。その状態で転送すると、ライセンスは使用できなくなります。

2. ライセンス ファイルがコピーされるのを可能にするために、転送先の PC で、転送先 CA Plex のライセンス ディレクトリ (Windows¥Oblicense) を共有し許可をフルに設定にしてください。

3. 転送先 CA Plex のライセンス ディレクトリにディレクトリ パスを割り当てます。

注: ¥¥MyServer¥¥Windows¥¥Oblicense などの UNC パス名はサポートされていません。K:¥¥Windows¥¥Oblicense などのドライブ名を使って転送先 PC にパスを割り当ててする必要があります。

4. 転送元のワーク ステーションで、ホスト グループ モデルにログインします。
5. [グループ モデル] ウィンドウで、ライセンスされたグループ モデルを選択し、[ヘルプ] メニューから [グループ モデル バージョン情報] を選択します。
6. [ライセンス] をクリックし、次に[転送]ボタンをクリックします。
7. [直接ライセンス転送] ダイアログに、転送先 PC の Windows¥¥Oblicense ディレクトリのパスを入力します（パスの先頭にはドライブ名が必要です）。

注: これは、CrypKey 4.2 のためのルート ディレクトリ OBLICENSE、あるいは CrypKey 6.1 のための OBLICENSE¥¥61 ディレクトリのいずれかです。

8. [OK] をクリックすると、ライセンスが転送されます。

メッセージ ボックスに転送結果が表示されます。

ライセンスのフロッピー ディスク転送

ネットワークに接続されていない PC 間でライセンスを転送するには、フロッピー ディスク転送を使用します。

注: ネットワーク上の他の PC へのライセンスの転送については、CA は直接転送をお勧めします。

フロッピー ディスク転送の場合は、準備が必要です。転送処理を始める前に、転送先 PC（ライセンスのないバージョン）のトライアル ライセンスの期限が切れている必要があります。これは、ライセンスを転送する前に転送先 PC を登録する必要がありますが、トライアル ライセンスが失効するまでは登録ができないためです。

フロッピー ディスクによるライセンス転送には、180 バイトの空き領域がある普通のフロッピー ディスクが必要です。

以下に、自宅と職場の PC の間で相互にライセンスを転送する方法について説明します。

転送の準備

ライセンスを転送する前に、次の作業を行う必要があります。

転送の準備

1. 職場の PC に **CA Plex** をインストールし、グループ モデルのライセンス コードを入力します。
2. 自宅の PC に **CA Plex** およびグループ モデルをインストールし、グループ モデルにアクセスします。トライアル ライセンスの期限が切れるまで待ちます。

ライセンスの転送

自宅と職場の PC 間でグループ モデル ライセンスを転送するには、次の手順に従います。

自宅の PC にグループ モデル ライセンスを転送

自宅の PC で

1. **CA Plex** を起動し、転送先のグループ モデルにアクセスします。すると、自宅の PC の登録に必要な新しいファイル一式が **Oblicense** ディレクトリ内に作成されます。
2. 空のフロッピー ディスクを自宅の PC に挿入して、「**A:¥HOME**」および「**A:¥OFFICE**」という 2 つのディレクトリを作成します。
注: このディレクトリの名前は自由です。ここでは、**HOME** ディレクトリと **OFFICE** ディレクトリとします。
3. ホスト モデルにログインし、ライセンス グループ モデルを選択して、[ヘルプ] メニューから [グループ モデル バージョン情報] を選択します。
4. [グループ モデル ライセンス] ダイアログで [ライセンス] をクリックした後、[転送] をクリックします。
5. [CA Plex ライセンスの転送] ダイアログの [転送先 PC の登録] をクリックします。
6. [ライセンス転送の登録] ダイアログに、「**A:¥HOME**」と入力し [OK] をクリックします。

これで自宅の PC が登録されました。登録ファイルが **HOME** ディレクトリに置かれました。**OFFICE** ディレクトリは空のままです。

職場の PC で

1. 作成したフロッピー ディスクを職場の PC に挿入します。CA Plex を起動して、[CA Plex ライセンスの転送] ダイアログを開きます。
2. [CA Plex ライセンスの転送] ダイアログで、[ライセンスをフロッピーに転送] をクリックします。
3. [ライセンスをフロッピーに転送] ダイアログに、「A:¥HOME」と入力し、[OK] をクリックします。

メッセージ ボックスに転送の進行状況が表示されます。

4. [OK] をクリックします。

この時点ではライセンスはフロッピー ディスク上にあり、職場の PC 上のグループ モデルのライセンスは無効です。翌日、職場の PC にライセンスを戻すには、この段階で職場の PC を登録する必要があります。

5. 表示中の [CA Plex ライセンスの転送] ダイアログで、職場の PC の登録手順を行います。
6. [ライセンス転送の登録] ダイアログに、「A:¥OFFICE」と入力して、[OK] をクリックします。

この時点で、フロッピー ディスク内の両方のディレクトリにデータが含まれます。HOME ディレクトリにはライセンスがあります。OFFICE ディレクトリには職場の PC の登録情報があります。登録情報は職場の PC にライセンスを戻すときに必要です。

注: この段階で、職場の PC を再び登録することはできません。再び登録しようとする、次のメッセージが表示されます。

「ソース ディレクトリにはすでに登録ファイルがあります。」

また、ここで職場の PC に他のライセンスを転送することはできません。それをしよう、とすると、元のライセンスを戻せなくなります。職場の PC に他のライセンスを転送すると、職場の PC のサイト コードが変わるためです。職場の PC に元のライセンスを戻そうとしても、サイト コードが変わるため使用できません。

自宅の PC で

1. フロッピー ディスクを自宅の PC に挿入し、[グループ モデル ライセンス] ダイアログを開きます。
2. [転送] をクリックします。
3. [CA Plex ライセンスの転送] ダイアログで、[ライセンスを PC に転送] をクリックします。
4. [ライセンスを PC に転送] ダイアログ ボックスに、「A:¥HOME」と入力し、[OK] をクリックします。

フロッピー ディスク上のライセンスが、自宅の PC に転送されます。この時点で HOME ディレクトリは空で、職場の PC にはライセンスがありません。

5. 職場の PC にライセンスを戻すには、ライセンスを OFFICE ディレクトリに転送します。

第 5 章: パターン ライブラリ

パターン ライブラリは再利用可能なデザイン オブジェクトのセットです。このライブラリに基づいてアプリケーションを開発することができます。

CA Plex は、次のパターン ライブラリおよびクラス ライブラリのセットを提供します。

- パターン ライブラリ
- クラス ライブラリ (OBASE ファミリー)

バージョン名とレベル名

CA Plex r7.0 で提供されるパターン ライブラリ バージョンは、V7.0 Patterns です。

パターン ライブラリのインストール

パターン ライブラリは、CA Plex のベース プロダクトの標準インストールで組み込まれます。「コンパクト」（最小限）インストールでも組み込まれます。インストール手順については、本書の「CA Plex のインストール」の章を参照してください。

クラス ライブラリのインストール

クラス ライブラリをインストールするには、「カスタム」（ユーザ指定）インストール（または既存のインストール内容を変更）を行う必要があります。インストール手順については、本書の「CA Plex のインストール」の章を参照してください。

クラス ライブラリの使用法については、CLASSLIBS.chm ヘルプ ファイルを参照してください。

ネットワーク上へのパターン ライブラリのインストール

一般的なワークグループ環境（「CA Plex のインストール」の「標準構成」を参照）では、パターン ライブラリは、すべての開発者がアクセス可能な 1 つのネットワーク ドライブ上にインストールされます。これにより、すべての開発者が同一バージョンのライブラリを使用することができ、モデル管理が簡略化されます。

ネットワーク上に、パターンあるいはクラス ライブラリをインストール

1. サーバ マシン上で、セットアップ プログラムを実行します。

詳細は、「CA Plex のインストール」の「CA Plex ベース プロダクトのインストール」を参照してください。

2. ライブラリのインストール先とするディレクトリを指定します。

注: ライブラリのロケーションとして UNC パス（例：`¥¥MyServer¥MyShare¥Libraries`）を使用することができます。つまり、ネットワーク上のライブラリにアクセスするすべての開発者が、同一のパスを使用することもできます。UNC パスを使用するかどうかは任意です。ライブラリにアクセスする必要があるのに、インストール時に指定したパス名が認識されない場合、CA Plex では有効なパスの入力を促すプロンプトが表示されます。

3. インストール オプションとして [カスタム] を選択します。
4. インストールしたいライブラリを選択し、その他のオプションの選択を解除します。

第 6 章: System i コンポーネント

本章では、System i コンポーネント、特に System i クライアント/サーバ、System i 5250 および Java クライアント System i サーバ製品のインストールおよび構成方法を説明します。

注: CA Plex は、System i での Java サーバ アプリケーションの開発をサポートします。設定手順については、「Java コンポーネント」の章を参照してください。

製品に含まれるもの

次の CA Plex for System i クライアント/サーバと CA Plex for System i 5250 の製品ライブラリは、CA Plex CD にバイナリ ファイルとして格納されています。

- リモート構成とランタイム オブジェクトを含む PLEX700J ライブラリ
- YTUTORIAL (CA Plex チュートリアル ライブラリ)
- YTUTREFER (CA Plex リファレンス チュートリアル ライブラリ)
- APPINTOBJ (CA Plex アプリケーション インテグレータ ライブラリ)

System i 上の開発に必要な最小要件

System i 上の開発環境には、以下の最小要件が必要です。

- リモート構成とランタイム オブジェクトを含む PLEX700J ライブラリ
- TCP/IP
- i5/OS V5R3 以上。
- RPG/400 コンパイラまたは ILE RPG/400 コンパイラ

RPG400 (または SQLRPG400) システム値を使って定義されているファンクションは、RPGIII 構文で生成され、RPG/400 コンパイラを必要とします。

RPGIV (または SQLRPGIV) システム値を使って定義されているファンクションは、RPGIV 構文で生成され、ILE RPG/400 コンパイラを必要とします。

- DDS コンパイラ
- UIM コンパイラ (System i 5250 ジェネレータのみ)
- System i CPIC 通信用の QSYS2 ライブラリ (ライブラリ リストのシステム部分に含まれていること)

- CA Plex 製品用の 4MB のディスク スペース（アプリケーション開発およびランタイム実行のため）
- 生成および構築する System i オブジェクト格納のための十分なスペース
- 「PC 上の開発環境」に記載された要件を満たす PC（System i クライアント/サーバ用アプリケーションのみ）

System i 上の展開に必要な最小要件

このセクションでは、System i 上の以下の実行に必要な要件を示します。

- System i 5250 アプリケーション
- System i クライアント/サーバ アプリケーション
- Java RPG/400 アプリケーション

必要なコンポーネントは以下のとおりです。

- 5250 タイプのターミナル、または 5250 エミュレーション パッケージを実行する PC（System i 5250 用アプリケーションのみ）
- i5/OS
- ファンクションを呼び出す際には、ライブラリ リストに以下のライブラリが含まれている必要があります。

物理ファイルと論理ファイルが存在するデータ ライブラリ

生成したプログラムが存在する生成ライブラリ

PLEX700J ライブラリ

注: データ ライブラリと生成ライブラリは 1 つのライブラリにまとめることができます。

ライブラリ リストに関する注意事項

実行時、ライブラリ リストで PLEX700J ライブラリがデータ ライブラリや生成ライブラリよりも下位になるようにする必要があります。

CD から System i への製品ライブラリの転送

CA Plex System i の製品ライブラリをリストアする前に、ファイルを CD から System i に転送する必要があります。以下の手順説明は、System i 上に TCP/IP がインストールおよび構成されており、アクティブになっていることを前提にしています。そうでない場合は、先へは進めません。TCP/IP のインストールに関して System i システム管理者に問い合わせてください。

CD から System i への System i (バイナリ) ファイルの転送

1. System i システム管理者に問い合わせ、転送先 System i の IP アドレス（たとえば、nnn.nnn.nnn.nnn）または識別名（myas400.mycompany.com）を調べます。
2. お使いの PC にログ オンします。
3. アクセス可能な CD-ROM ドライブに、CA Plex CD を挿入します。
4. PC 上でコマンド ライン ウィンドウを開きます。
5. ドライブを CD-ROM ドライブに変更します。
6. 「cd as400libs」と入力し、Enter キーを押します。
7. 「ftp」と入力し、Enter キーを押します。

FTP セッションが始まります。

8. 「open nnn.nnn.nnn.nnn」（nnn.nnn.nnn.nnn は System i の TCP/IP アドレス - IP アドレスのかわりに識別名も使用できます）と入力し、Enter キーを押します。

接続が完了すると、ユーザ プロファイルの入力を促すプロンプトが表示されます。

注: 接続した後、アイドル状態が 5 分続くと、システムは自動的に接続を閉じます。セッションを再開するために、手順 7 からやり直す必要があります。

9. プロンプトに System i ユーザ プロファイルを入力して、Enter キーを押します。
10. プロンプトに System i パスワードを入力して、Enter キーを押します。

注: パスワードを入力している間、カーソルは動きません。

「xxx logged on」（xxx はユーザ プロファイル）という応答メッセージが表示されます。

11. 「quote site namefmt 1」と入力し、Enter キーを押します。

注: 「namefmt」と「1」の間にスペースが1つ必要です。

「Now using naming format "1"」という応答メッセージが表示されます。

注: 名前割当てフォーマットには1を指定する必要があります。

12. 「bin」と入力し、Enter キーを押します。これで、バイナリ フォーマットが設定されます。

13. 「cd qgpl.lib」と入力して、Enter キーを押します。

注: QGPL の代わりに他の既存のライブラリも指定できます。.lib 拡張子は必須です。

「Current library changed to QGPL (現在のライブラリが QGPL に変更)」という応答メッセージが表示されます。

14. 「put plex700j.savf」と入力し、Enter キーを押します。ファイル転送時間は、ネットワークの速度に応じて変わります。

注: .savf 拡張子は必要です。

「File transfer completed successfully (ファイル転送が完了しました)」という応答メッセージが表示されます。

15. ytutorial.savf、ytutrefer.savf、および appintobj.savf について、手順 14 を繰り返します。

16. 「quit」と入力し、Enter キーを押します。

FTP セッションが中止されます。

17. コマンドライン セッションを終了します。

これで、*SAVF ファイルをリストアできるようになりました。

保管ファイル(*SAVF)からの製品ライブラリのリストア

CA Plex for System i クライアント/サーバ、または CA Plex for System i 5250 用のコンポーネントを保管ファイルからインストールするには、下記の手順に従います。

1. QSECOFR、または QSECOFR と同等の資格でログ オンします。
2. CA Plex の旧バージョンからアップグレードする場合は、まず下記のコマンド文字列を入力してそれぞれのライブラリをクリアします。

```
CLRLIB LIB(PLEX700J)
CLRLIB LIB(YTUTORIAL)
CLRLIB LIB(YTUTREFER)
CLRLIB LIB(APPINTOBJ)
```

3. CA Plex の製品ライブラリを以下の手順でリストアします。

```
RSTLIB SAVLIB(PLEX700J) DEV(*SAVF) SAVF
(QGPL/PLEX700J) FRCOBJCVN(*YES *RQD)
```

4. 次のように入力して、チュートリアル ライブラリをリストアします。

```
RSTLIB SAVLIB(YTUTORIAL) DEV(*SAVF)
SAVF(QGPL/YTUTORIAL)
```

5. 次のように入力して、チュートリアル ライブラリをリストアします。

```
RSTLIB SAVLIB(YTUTREFER) DEV(*SAVF)
SAVF(QGPL/YTUTREFER)
```

6. 以下のように入力して、アプリケーション インテグレータ ライブラリをリストアします。:

```
RSTLIB SAVLIB(APPINTOBJ) DEV(*SAVF)
SAVF(QGPL/APPINTOBJ)
```

7. PLEX700J をライブラリ リストの先頭に加えます。

PC – System i 間の通信用ソフトウェア

System i の開発では、PC が TCP/IP を介して System i にリンクされている必要があります。

CA Plex TCP/IP 環境の構成

以下のガイドラインでは、開発者とエンド ユーザの両方のニーズに応じた CA Plex TCP/IP プロトコル サービスの構成方法を説明します。TCP/IP は、QCMN サブシステムを使用せず、また必要としません。したがって、以下の説明は QCMN がないことを前提にしています。

TCP/IP を使用するには、以下を必要とします。

- i5/OS V5R3 以上

System i 用に TCP/IP を構成する際の詳細については、IBM 刊、「TCP/IP Configuration and Reference (SC41-34209-00)」を参照してください。以下の説明は、System i 上に TCP/IP がインストールされ、アクティブになっていることを前提にしています。

- System i 上で構成された開始されている TCP/IP
- System i 上で開始されている TCP/IP バージョン System i CA Plex ディスパッチャ

TCP/IP を使用して CA Plex クライアントが接続するには、以下を必要とします。

- System i 上で開始されている CA Plex TCP/IP ディスパッチャ
- TCP/IP ディスパッチャ プログラムの起動時割り当てられたポート番号

YOBSYTCPDP TCP/IP ディスパッチャ プログラム

CA Plex r7.0 では、YOBSYTCPDP ディスパッチャ プログラムの使用をお勧めします。ディスパッチャのこのバージョンは、以下を提供します。

- 向上したセキュリティ。特にこのディスパッチャは、クライアントとサーバ間通信に単一ソケットを使用します。ファイアウォールを介して開く必要があるのは、指定したポートのみです。
- (IPv4 に加えて) IPv6 プロトコルをサポートします。

CA Plex r7.0 に含まれるディスパッチャ プログラムは、V5R2 以前のバージョンとは互換性がありません。

ディスパッチャ プログラムの代替バージョン

YOBSYTCPDP の V5R3 バージョン

(本書の執筆時点では) V5R3 は IPv6 を完全にはサポートしていません。そのため、V5R3 では YOBSYTCPDP の異なるバージョンを必要とします。このバージョンは 2 つのプログラム YOBSYTCP3C3 そして YOBSYTCPD3 として製品に含まれています。V5R3 互換バージョンをインストールするには、

1. Plex ライブラリ (PLEX700J) 内の既存の YOBSYTCPCT と YOBSYTCPDP プログラムをリネームします。
2. YOBSYTCP3C3 プログラムを YOBSYTCPCT に、YOBSYTCPD3 を YOBSYTCPDP にリネームします。

注: リネームを行う前にすべての実行中のディスパッチャを停止し、リネーム後にリスタートしてください。

V5R3 でのユーザ名制限

V5R3 でのみ YOBSYTCPDP には 1 つの制限があります。それは、発生したジョブ名の Job User パラメータに対する実際のユーザ プロファイルが i5/OS WRKACTJOB 画面上で表示されないことです。この制限は、V5R4 以降のバージョンではありません。

V5R3 でのこの i5/OS 制限のための回避策として、YOBSYTCPDP のかわりに YOBSYTCP ディスパッチャを実行することができます。しかし、YOBSYTCP ディスパッチャは、ファイアウォールとの互換性がないとともに IPv6 もサポートしていないため一般的にはお勧めできません。

YOBSYTCP_R ディスパッチャ

CA Plex の以前のリリースでは、YOBSYTCP_R という名前のディスパッチャ プログラムも含まれていました。このプログラムが提供していた機能を必要とされる場合は、CA テクニカル サポートにご連絡ください。

ユーザ権限の要件

USER パラメータに QSECOFR を設定する必要はありませんが、次の権限を持つユーザ パラメータを設定する必要があります。

- オブジェクト QSYS/QSYGETPH および QSYS/QSYRLSPH の *USE 権限。
- *ALLOBJ 権限は必要ありませんが、接続するクライアント ユーザ プロファイルは、ディスパッチャを開始したユーザ プロファイルに *USE 権限および *OBJMGT (オブジェクト管理) を与える必要があります。この要件は、ディスパッチャを開始したユーザ プロファイルが、接続するクライアント ユーザの i5/OS ユーザ プロファイルおよびパスワードを検証するために必要です。
- *SECADM 権限は必要ではありませんが、接続するクライアント ユーザ プロファイルは、もしディスパッチャを開始したユーザ プロファイルが *SECADM 権限を持たない場合は、自身の期限切れのパスワードを変更することはできません。
*SECADM 権限無しでディスパッチャを実行したい場合、エンド ユーザが自身のパスワードを変更するための代わりのメカニズムを提供する必要があります。詳細は、オンラインヘルプの「System i TCP/IP 接続でのパスワード管理」を参照してください。

注: 接続するクライアント ユーザ プロファイルが、ディスパッチャを開始した同じユーザ プロファイルである場合、自身の期限切れのパスワードをリセットするために *SECADM 権限は必要ありません。

オブジェクト権限

グラント オブジェクト権限 (GRTOBJAUT) コマンドあるいは編集オブジェクト権限コマンドをこれらのオブジェクト権限を与えるために使用することができます。

例:

この例では、STARTDSP ユーザ プロファイルが System i 上でディスパッチャ (YOBSYTCPDP あるいは YOBSYTCP) を開始し、STARTDSP ユーザ プロファイルは *ALLOBJ 権限は持っていませんが、*SECADM 権限を持っているものとします。CONNECTUSR ユーザ プロファイルは接続するクライアント ユーザ プロファイルです。

```
GRTOBJAUT OBJ(QSYS/CONNECTUSR) OBJTYPE(*USRPRF) USER(STARTDSP) AUT(*OBJOPR *READ *EXECUTE)
```

注: AUT (*OBJOPR *READ *EXECUTE) は、「編集オブジェクト権限」(EDTOBJAUT) コマンドを使用する場合の *USE 権限と同等です。

上記の例において、STARTDSP ユーザ プロファイルが、CONNECTUSR ユーザ プロファイルに対してオブジェクト管理 (*OBJMGT) 権限を持たないため、CONNECTUSR ユーザ プロファイルは、期限切れのパスワードをリセットできません。

次の例では、STARTDSP ユーザ プロファイルが CONNECTUSR ユーザ プロファイルに対してオブジェクト管理 (*OBJMGT) 権限を持つので、CONNECTUSR ユーザ プロファイルは、期限切れのパスワードをリセットできるようになります。

```
GRTOBJAUT OBJ(QSYS/CONNECTUSR) OBJTYPE(*USRPRF) USER(STARTDSP) AUT(*OBJMGT
*OBJOPR *READ *EXECUTE)
```

System i TCP/IP ディスパッチャの開始

System i TCP/IP ディスパッチャの開始についての例を実行する前に、前のセクションを確認してください。

注: この例のパラメータ「61000」は割当ポートです。ただし、IANA (Internet Assigned Number Authority) にこれまで登録されていない未使用の番号であれば、どの番号でも使用できます。登録済みのポートは通常 0 から 1023 までの間です。登録済みの番号との競合を避けるために、ポート番号には 3000 から 61000 までの値を使用することをお勧めします。

YOBLISTEN ジョブは、手動あるいはシステム IPL によって終了されるまで実行されています。IPL 後、STRTCP コマンドが発行され、TCP/IP のスタートアップが完了する時間に SCDDTIME パラメータをセットしてください。

例 1:

PLEX700J/YOBSYTCPPD または PLEX700J/YOBSYTCP を起動する独自の CL プログラムを作成して、CL プログラムを IPL の一部として呼び出します。

例 2:

ジョブ スケジュール項目追加 (ADDJOBSCDE) を作成します。

次に、IPL が毎晩発生する場合に使用できる例を示します。

```
ADDJOBSCDE JOB(YOBLISTEN) CMD(CALL )) FRQ(*WEEKLY)
PGM(PLEX700J/YOBSYTCPPD) PARM('61000'
SCDDATE(*NONE) SCDDAY(*ALL) SCDDTIME('hh:mm:ss')
JOBQ(PLEX700J/PLEX) JOBQ(QGPL/QINTER) USER(QSECOFR)
TEXT(Plex TCPIP C/S Dispatcher')
```

注: 前出のステートメントの PARM 値には一重引用符 (') が必要です。これがないと、呼び出しは成功しません。

ADDJOBSCDE コマンドの投入後、次の IPL 後の予定時刻までディスパッチャは起動しません。

次の IPL より前にディスパッチャを起動

次の IPL より前に、ディスパッチャを開始することができます。

次の IPL より前にディスパッチャを起動

1. コマンド ラインから次のコマンドを実行します。

```
WRKJOBSCDE
```

2. ジョブの次に「10」と入力して、Enter キーを押します。

ジョブがただちに実行されます（通常、これは、ディスパッチャを設定した日に 1 度行う必要があるだけです）。

追加の TCP/IP ディスパッチャの起動

テストの目的で、別のポート上に追加の CA Plex TCP/IP ディスパッチャをいつでも起動できます。たとえば、TCP/IP 用の CA Plex を初めて System i に構成する際、ジョブを起動ルーチンに加える前に、手動でディスパッチャを起動させる必要がある場合などです。

CA Plex TCP/IP ディスパッチャのポート上での手動による起動

1. コマンド ラインから下記のコマンドを実行します。

```
SBMJOB CMD(CALL PGM(PLEX700J/YOBSYTCPDP) PARM('61000')) JOB(YOBLISTEN)
JOBBD(PLEX700J/PLEX)
JOBQ(QGPL/QINTER)
```

YOBSYTCPDP（または YOBSYTCP）プログラムに渡されるパラメータには、未使用の利用可能なポート番号が必要です。

2. ポートが起動したことを確認するには、コマンド ラインに下記を入力します。

```
WRKTCPSTS *CNN (割り当てたポート番号のローカル ポート カラムをチェックしてください)
```

注: 接続要求を行う CA Plex クライアントが対話型の応答を得られるように、ジョブ待ち行列に QINTER を使用してください。CA Plex TCP/IP ディスパッチャを起動するジョブ待ち行列は、要求された際に個別のクライアント ジョブの投入に使用されます。バッチ ジョブ待ち行列を使用すると、CA Plex クライアント ジョブがバッチ ジョブ待ち行列内にとどまって、クライアント側にタイムアウトと「ハング」状態を引き起こすことがあります。

製品ライブラリの別名でのリストア

製品ライブラリは、デフォルトの **PLEX700J** とは別の名前でリストアすることができます。その場合、以下のセクションで説明するようにいくつかのライブラリ名への参照を手動で更新する必要があります。

- ジョブ記述の更新
- PC ファイルの修正

ジョブ記述の更新

デフォルトの **PLEX700J** とは別の名前で製品ライブラリをリストアするために、手動でジョブ記述を更新しなければなりません。

ライブラリのリストア後、**JOB**D を更新するために次のコマンドを入力します。

```
CHGJOB D JOB(ライブラリ名/PLEX)
INLLIBL(QTEMP ライブラリ名 YTUTORIAL YTUTREFER APPINTOBJ
QGPL)
```

PC ファイルの修正

以下に、デフォルトの名前 (**PLEX700J**) の代わりに指定した **CA Plex System i** の製品ライブラリの名前 (この例では「ライブラリ名」) を使用するに必要な **CA Plex** のいくつかのデフォルト設定を変更する方法について説明します。

PC ファイルの修正

1. ワードパッドなどのテキスト エディタを開きます。
2. 新しいバージョンで使用するために割り当てた一意の **System i** ライブラリ名を反映させるために、以下のファイルを修正します。

注: 次のファイルは **CA Plex** を初めて起動した際に、「マイ ドキュメント」フォルダ (**CA\Plex\7.0** サブ フォルダ内) にコピーされています。既に **CA Plex** を実行していた場合は、「マイ ドキュメント」フォルダ内のコピーも更新してください。

- C:\Program Files\CA\Plex\7.0\Plex.bld

[remote configuration] という名前のセクション下にある次の行を

```
jobdesc lib=PLEX700J
```

下記のように変更します。

```
jobdesc lib=ライブラリ名
```

- C:\Program Files\CA\Plex\7.0\Plex.INI

[Remote] という名前のセクション下にある次の行を

Library=PLEX700J

下記のように変更します。

Library=ライブラリ名

- C:¥Program Files¥CA¥Plex¥7.0¥Bin¥Obsyrt.ini

[Remote] という名前のセクション下にある次の行を

Library=PLEX700J

下記のように変更します。

Library=ライブラリ名

3. Obsyrt.ini と同じように次のファイルを変更します。

- ¥Bin¥Ob600rc.INI
- ¥Bin¥Ob600rcd.INI

第 7 章: Windows C++ サーバ コンポーネント

本章では、CA Plex の Windows C++ サーバ コンポーネントの構成方法を説明します。

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だけに提供されます。CA は Windows サーバ プラットフォームのための新規 Plex アプリケーションの開発には、C# または Java ジェネレータの使用を強くお勧め致します。既存の WinNTC アプリケーションをお持ちのお客様は C# または Java への移行をご検討ください。

Windows C++ サーバの開発要件

Windows C++ サーバに必要なソフトウェアは以下のとおりです。

- Windows Server 2008 の 64 ビット エディションをサポートとしています。Plex WinNTC アプリケーションは、WOW64 環境で 32 ビット アプリケーションとして実行されます。純粋な 64 ビット アプリケーションのためには、C# または Java ジェネレータを使用する必要があります。
- Microsoft Visual Studio 2005 (Professional Edition または Standard Edition)
- CA Plex Windows C++ アプリケーション サーバ SDK
- Microsoft SQL Server または Oracle

注: Microsoft Visual Studio 2005 は「管理者として実行」で設定する必要があります。devenv.exe のプロパティの「互換性」タブを開き、下部にある「管理者として実行」チェック ボックスから設定します。

更なるシステム要件についての詳細は、リリース ノートを参照してください。

Windows C++ サーバ コンポーネントのインストール

Windows サーバ アプリケーションを開発するためには、開発バージョンの **CA Plex** アプリケーション サーバをインストールしなければなりません。

Windows クライアントの要件

各開発者のマシンで **CA Plex** インストール プログラムを実行し、クライアント コンポーネントをインストールする必要があります。**CA Plex** の標準インストールには、必要なコンポーネントが含まれています。

さらに、**CA Plex Windows C++** アプリケーション サーバ **SDK** をサーバ マシン上にインストールしなければなりません。

アプリケーション サーバ

アプリケーション サーバは、**Windows** サーバ上の **CA Plex C++** アプリケーションをサポートするサービスおよびコンポーネントの名前です。

アプリケーション サーバは以下を含んでいます。

- 環境マネージャ
- 構築サービスおよび構築サービス マネージャ
- ディスパッチ サービスおよびディスパッチ サービス マネージャ
- 印刷サービス

これらのコンポーネントは、**Plex¥AppServer¥Bin** ディレクトリにインストールされます。

アプリケーション サーバのインストール

Windows Server 2008 オペレーティング システム上にアプリケーション サーバをインストールするには、次の権限が必要です。

- **CA Plex** が以前にドメイン内のどこかにインストールされていた場合は、ローカル管理者の権限が必要です。
- **CA Plex** がこれまでにインストールされたことがない場合は、ドメイン管理者の権限が必要です。サーバは、ドメインの一部でなければなりません。

CA Plex ベース プロダクトのインストール時にアプリケーション サーバをインストール

1. インストールするためのセットアップ タイプを指定する際に、[カスタム] をクリックします。
2. すでにチェックされているものに加え、CA Plex Windows C++ アプリケーション サーバ SDK をチェックします。

CA Plex ベース プロダクトのインストール後にアプリケーション サーバをインストール

1. [スタート] - [設定] - [コントロール パネル] の順にクリックします。
2. [アプリケーションの追加と削除] アイコンをダブルクリックします。
3. 製品のリストから [CA Plex] を選択します。
4. [変更と削除] をクリックします。
5. InstallShield Wizard で、[変更] を選択してから [次へ] をクリックします。
6. CA Plex Windows C++ アプリケーション サーバ SDK のチェック ボックスを選択します。
7. [次へ] ボタンをクリックすると、インストールが始まります。

ジョブ ステータス データベース

構築サービスをサポートするために、アプリケーション サーバのインストール プログラムは、自動的に以下を行います。

- CA Plex 構築に関するステータス情報を格納するための Microsoft Access データベースの作成。このジョブ ステータス データ ベースの位置は、¥AppServer¥Jobsts¥Objobsts.mdb です。
- ジョブ ステータス データベース用の ODBC データ ソースの設定。

構築サービスおよびディスパッチ サービスの構成

インストールされたサービスに対してスタートアップ情報を構成することができます。インストール時には、デフォルトでスタートアップの種類は「手動」に設定されています。これを、サービスがブート時に開始するように「自動」に変更できます。

詳細については、オンライン ヘルプ (Ntservices.chm) を参照してください。

Oracle のサポート

Windows サーバ (WinNTC) ジェネレータを使用して、Windows サーバ上の Oracle データベースのデータにアクセスするアプリケーションを作成できます。ランタイム アプリケーションは、高速データ アクセスにネイティブの Oracle Call Interface (OCI) を使用します。

CA Plex を使用する前に、Oracle データベースがサーバ上に適切にインストールされていて (詳細は Oracle マニュアルを参照してください)、基本ユーザ (SCOTT などのデフォルト ユーザ) が次の権限を備えて構成されていることを確認してください。

- Connect
- Resource
- Tablespace

Windows クライアント コンポーネント (Microsoft RPC) のインストール

Windows クライアント コンポーネント (Microsoft RPC) をインストールする前に、次の点を考慮してください。

- CA Plex クライアントは Microsoft RPC ランタイム サービスを使用してアプリケーション サーバに接続します。
- CA Plex の環境マネージャを使用して、各 CA Plex 開発者に対して、Windows サーバ上で適切な権限付きのユーザ プロファイルを作成してください。詳細については、オンライン ヘルプの「ユーザ環境の作成」を参照してください。
- データベース設定、ユーザ権限付与、データ ソース名構成の詳細は、Microsoft SQL サーバのマニュアルを参照してください。

Windows C++ アプリケーション サーバ

CA Plex のインストール オプションには、Windows C++ アプリケーション サーバ オプションがあります。このオプションを選択すると、アプリケーションの展開に必要なコンポーネントだけがインストールされます (ディスパッチ サービス、印刷サービスおよび環境マネージャ)。

第 8 章: Java コンポーネント

本章では、CA Plex の Java コンポーネントのインストールと構成方法について説明します。詳細については、オンラインヘルプの「Java プラットフォーム」セクションを参照してください。

Java の開発要件

CA Plex は、クライアントとサーバの両方における Java アプリケーションの生成をサポートします。以下の構成において、Java アプリケーションの開発および展開を行うことができます。

- Java クライアントと RPG (System i) サーバ
- Java クライアントと Java サーバ
- Java クライアントまたは Java サーバと C# .NET サーバ
- C# .NET クライアントまたは C# .NET サーバと Java サーバ
- Windows クライアントと Java サーバ

CA Plex は、任意のプラットフォーム上の任意の Java 仮想マシン下で実行される 100% 純粋な Java のコードを生成します。また、CA Plex で生成した既存の Java アプリケーションの場合、ランタイムは下位互換性を持つため、既存のアプリケーションを再生成することなく新しいバージョンの CA Plex Java ランタイムを適用することができます。

CA Plex Java クライアントは、主に Windows 上でテストされています。Java クライアント アプレットのテストには、Sun の Java Plug-in が使用されます。CA Plex Java クライアントが GUI コンポーネントをサポートするためには、Sun の Swing クラスが必要であることに注意してください。

以下の表に、CA Plex Java 開発に必要なインストール要件をまとめてあります。展開要件は、オンライン ヘルプの「Java プラットフォーム」セクションに記載されています。

コンポーネント	説明
Java クライアント	■ CA Plex Java コンポーネント ■ Sun Java SE JDK 7.0。他のバージョンの JDK も CA Plex で動作しますが、完全にテストされているものではありません。無償でダウンロードすることができます ■ JAVA_HOME 環境変数が定義されているはずです。
Java サーバ	Java クライアントの要件に加えて、 ■ サードパーティの DBMS および JDBC ドライバ (ランタイム データ アクセス用) ■ ODBC ドライバ (データ ベース構築用、実行時には必要ありません) System i Java サーバ開発の場合 ■ PLEXJVA700 ライブラリ ■ IBM Client Access (またはそれと同等のもの) ■ IBM System i Toolbox for Java

Java コンポーネントのインストール方法

CA Plex Java コンポーネントは、CA Plex をインストールすると自動的にインストールされます。インストールされるコンポーネントには以下が含まれます。

- ObJava ディレクトリ
- Apache ANT
- ObJava ディレクトリ内の obJavadsp.bat ファイル

Java コンポーネントのインストールおよび構成を行うには、以下を実行します。

1. Sun Java SE 開発キット (JDK) の必要となるバージョンをダウンロードしてインストールします。
2. JAVA_HOME 環境変数をセットします。
3. ランタイム用のサーバを構成します。
4. サーバに Java サーバ ファクション (JAR ファイル) を転送します。
5. Java ディスパッチャを開始します。

JAVA_HOME の環境変数を定義

JAVA_HOME 環境変数は、Plex Java 構築に使用される Java コンパイラ(Javac.exe)を決定します。Plex は、この環境変数を自動的にセットしません。ユーザは自由にコンパイラ バージョンを切り替えることができます。この設定は、生成されたアプリケーションの実行時には必要ありません。

JAVA_HOME の環境変数を定義

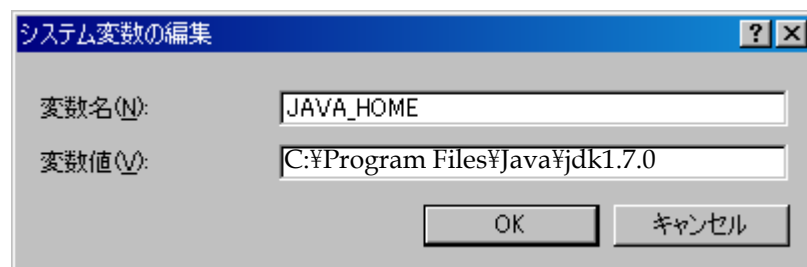
1. Windows のコントロール パネルで、[システム] アイコンをダブルクリックします。
2. [詳細設定] タブの、[環境変数] ボタンをクリックします。

JAVA_HOME をユーザ環境変数あるいはシステム環境変数のいずれかとして定義することができます。

3. [新規] ボタンをクリックして、「JAVA_HOME」の変数名を入力し、変数値に JDK がインストールされているフォルダのフルパスを入力します。例えば JDK 1.7 の場合は、C:¥Program Files¥Java¥jdk1.7.0

注: JAVA_HOME には、JDK 導入フォルダを指定し、サブフォルダの Bin を指定しないよう注意してください。

例:



4. [OK] をクリックします。

JAVA_Home の環境変数を定義しました。

ランタイム用のサーバの構成

以下を行い、ランタイム用のサーバを構成します。

1. クライアント マシンの CA Plex ディレクトリからリモート サーバへ、ObjJava ディレクトリをコピーします。

注: いくつかの ObjJava ランタイム アプリケーションのサブフォルダは必要ありません。例えば、Apache-Ant と src サブフォルダをコピーする必要はありません。

2. Java アプリケーションのサーバ ファンクションをクライアントに構築した後、サーバへ JAR ファイルを移してください。

3. 次のいずれかの方法で、CA Plex Java ディスパッチャを管理します。

製品に含まれているサブレットを処理する Web サーバを使用

セッション内で Java VM インタープリタを実行するバッチ ファイルを使用

バッチ ファイルを使用するのが、最も簡単な方法ですが、アプリケーション運用には、Web サーバの使用が適しています。

ディスパッチャの開始手順についての詳細は、オンライン ヘルプの「バッチ ファイルを使用した Java ディスパッチャの起動」を参照してください。

Web サーバについては、オンライン ヘルプの「Web サーバのセットアップ」を参照してください。

Java ディスパッチャの代わりに、EJB ランタイム プロキシを使用して配布した Java アプリケーション内での通信を管理できます。詳細は、オンライン ヘルプを参照してください。

System i での Java の設定

CA Plex Java のファンクションを System i 上で実行するには、PLEXJVA700 ライブラリおよび CA Plex Java コンポーネントを System i 上にインストールする必要があります。

これらの手順では、System i Toolbox for Java を使用していることを前提としています。System i 上の IFS (Integrated File System) にアクセスするために、IBM Client Access (またはそれと同等のもの) を使用することもお勧めします。

注: RPG サーバ アプリケーションを使用する Java クライアントを開発している場合には、このセクションを飛ばしてもかまいません。System i 上の RPG サーバ ファンクションと接続するために、Java クライアントは PLEX700J 製品ライブラリに含まれる System i TCP/IP ディスパッチャを使用します。インストール プロセスは Windows クライアントの場合と同じで、「System i コンポーネント」の章で説明されています。

PLEXJVA700 ライブラリ

PLEXJVA700 ライブラリは、CA Plex の CD-ROM 上にある AS400 Libraries フォルダに保管ファイルとして提供されています。これを System i に転送し、他の CA Plex System i ライブラリと同じ方法でリストアする必要があります。詳細については、本書で前述した「CD から System i への製品ライブラリの転送」および「保管ファイル (*SAVF) からの製品ライブラリのリストア」を参照してください。

System i への Objava ディレクトリのコピー

System i へ Objava ディレクトリをコピーするには、以下の手順を実行してください。

1. 本章の「Java コンポーネントのインストール方法」セクションの説明のとおり、CA Plex Java コンポーネントを Windows マシン上にインストールします。
2. System i IFS のトップレベルにディレクトリを作成します。以下の手順では、そのディレクトリの名前を「Plex」とします。
3. Objava ディレクトリおよびその内容を Windows マシンから System i へコピーし、Plex/Objava サブディレクトリを作成します。

System i 上の Java ディスパッチャを構成

次の手順は、System i Toolbox for Java の JDBC ドライバを使用することを前提としています。ディスパッチャは、System i Toolbox for Java クラスをクラスパス上で利用可能にするために必要な JDBC 設定で構成される必要があります。

Java ディスパッチャを構成するには、以下の手順を実行してください。

1. IFS のトップレベルにディレクトリを作成します。例えば、/PlexJava を作成します。
2. obsrv.properties ファイルの設定を変更します。これらの設定に関する説明は、CA Plex のヘルプを参照してください。必要な変更を加えた一般的な例を示します（太字の箇所）。各エントリは 1 行で記述する必要があります。

```
Environment.Default.Driver=  
    com.ibm.as400.access.AS400JDBCDriver  
Environment.Default.DataSource=  
    jdbc:as400://マシン/ライブラリ  
Environment.Default.User=ユーザ ID  
Environment.Default.Password=パスワード
```

Java クラスの最適化

System i 上で正常に動作するように、システムの各 .class ファイルを最適化しなければなりません。これは 1 回だけ実行してください。

CRTJVAPGM コマンドを、単一のファイル、JAR ファイル全体、またはディレクトリ全体（ワイルドカードを使用）に対して使用します。最適化を 4 つレベルの中から選択することができます。特に理由がある場合を除き、完全最適化（レベル 40）を使用してください。このコマンドは、System i Java 仮想マシンによってロードされる各 .class ファイルについて最適化された隠しファイルを 1 つずつ作成します。このコマンドの詳細については、System i のオンライン ヘルプを参照してください。

以下のファイルについて CRTJVAPGM をバッチ処理として実行します。

- obrun.jar
- JT400.jar (このファイルをデフォルトの場所から、以前に作成した「Plex」ディレクトリにコピーすることによって、パスを短くすることができます)

System i 上の Java ディスパッチャの起動

YSTRJVADSP コマンドを使用して、ディスパッチャを起動することができます。また、RUNJVA コマンドを使用して、サービスを開始することができます。

YSTRJVADSP コマンドを使用して、Java ディスパッチャを開始

1. ライブラリ リストに PLEXJVA700 を追加します。
2. YSTRJVADSP コマンドを使用して、ディスパッチャを開始します。

注: このコマンドは、ポート番号、ランタイムとプロパティ ファイルの位置についてのパラメータを含みます。JT400.jar ファイルへのパスを、「Path To Additional Class」パラメータで参照する必要があります。

このコマンドの詳細については、CA Plex のオンライン ヘルプのキーワード YSTRJVADSP を参照してください。

サービスを開始するための 2 番目の方法は、RUNJVA コマンドを使用する方法です。

RUNJVA コマンドを使用して、サービスを開始

1. System i のコマンド行に「RUNJVA」と入力してから F4 キーを押します。
2. Class パラメータに、サービスを必要とするランタイム クラス名を入力します。
ObRun.ObComms.ObService - Java ディスパッチャの起動用
注: クラス名は大文字と小文字が区別されます。
3. Parameters パラメータに、ポート番号およびプロパティ ファイルへのパス、例えば、「'1998' '/Plex/Objava'」を入力します。
4. Classpath プロンプトに、CA Plex Java ランタイムへのフルパス、例えば、「/Objava/lib/obrun.jar」を入力します。

第 9 章: CA 2E データ移行

本章では、CA 2E データ移行のインストール方法を説明します。

2E データ移行製品は、CA 2E から CA Plex へのリレーショナル データ モデリング メタ データの移行をサポートします。アクション ダイアグラム コードとパネル デザインの移行はサポートしていませんのでご注意ください。

製品に含まれるもの

CA 2E データ移行製品には、CA Plex 製品 CD-ROM の System i ライブラリ ディレクトリ内にある次の移行ライブラリが含まれます。

- YOMF
- YOMFVENG
- YOMFDB

CA 2E データ移行の要件

PC および System i の要件については、以下を参照してください。

- PC 開発環境
- 必要最小限の System i 開発要件

このセクションでは、CA 2E データ移行に特有の要件のみを説明します。

メモリ

CA 2E データ移行に最小限必要なメモリは、1 GB RAM です。

データ移行は、大規模な CA Plex モデルを高速で充填できるので、大きなサイズの CA 2E モデルをインポートする場合は、開発に使用する PC のメモリ (RAM) 増設を検討してください。

System i に必要な最小要件

System i には以下の要件があります。

- i5/OS
- PC - System i 間の通信用ソフトウェア パッケージ（通信用ソフトウェア固有の要件の詳細については、リリース サマリの「CA Plex の旧バージョンからのアップグレード」を参照してください）
- CA 2E データ移行製品のための 20MB の空き容量

製品ライブラリのリストア

CA Plex オブジェクトを製品 CD から System i へリストアするには、次の手順を実行します。

1. 移行ファイル YOMF.savf、YOMFVENG.savf、YOMFDB.savf を CD から System i に転送します。詳しい手順については、「CD から System i への製品ライブラリの転送」を参照してください。
2. 「保管ファイル (*SAVF) からの製品ライブラリのリストア」の手順を実行してください。

YOMF.savf、YOMFVENG.savf、および YOMFDB.savf ファイルをリストアします。

3. CA Plex の旧バージョンをアップグレードしている場合は、データ移行用の製品ライブラリ名が変更されていることに注意してください。導入手順としては必要ありませんが、古いライブラリを削除したい場合は、次のコマンド文字列を使用して行えます。

```
DLTLIB LIB(YIMPS2OBSY)
DLTLIB LIB(YIMPS2VENG)
DLTLIB LIB(YOBSCS)
DLTLIB LIB(YOBSCSVENG)
```

注: PC - System i 間の通信用ソフトウェアの手順を完了していない場合は、以下のコマンド文字列を入力します。

```
CLRLIB LIB(PLEX610J)
CLRLIB LIB(YTUTORIAL)
CLRLIB LIB(YTUTREFER)
```

4. CA Plex のランタイム ライブラリをリストアします。
V3R6 あるいはそれ以降のものを使用している場合は、以下のように入力します。

```
RSTLIB SAVLIB(PLEX610J) DEV(device) ENDOPT(*LEAVE) RSTLIB(PLEX610J)
FRCOBJCVN(*YES *RQD)
```

5. 次のように入力して、チュートリアル ライブラリをリストアします。

```
RSTLIB SAVLIB(YTUTORIAL) DEV(device) ENDOPT(*LEAVE)
```

6. 次のコマンドを入力して、チュートリアル リファレンス ライブラリをリストアします。

```
RSTLIB SAVLIB(YTUTREFER) DEV(device) ENDOPT(*REWIND)
```

7. PLEX610J をライブラリ リストの先頭に加えます。

8. 以下のコマンドを利用して、移行用主製品ライブラリをリストアします。

```
RSTLIB SAVLIB(YOMF) DEV(device) ENDOPT(*LEAVE) RSTLIB(PLEX610J)
FRCOBJCVN(*YES *RQD)
```

9. 以下のコマンドを利用して、移行用ユーザ インターフェース ライブラリをリストアします。

```
RSTLIB SAVLIB(YOMFVENG) DEV(device) ENDOPT(*LEAVE) RSTLIB(PLEX610J)
FRCOBJCVN(*YES *RQD)
```

10. 以下のコマンドを利用して、移行用マッピング データベースをリストアします。

```
RSTLIB SAVLIB(YOMFDB) DEV(device) ENDOPT(*LEAVE) RSTLIB(PLEX610J)
FRCOBJCVN(*YES *RQD)
```

CA 2E 並行インストール

CA 2E を並行インストールする場合は、YIMPS2OBSY/QBATCH のジョブ記述ライブラリ リストを、環境に合わせて設定しなおす必要があります。次の場合には、正しいライブラリを指すようジョブ記述を設定してください。

- CA 2E オブジェクトを新たなライブラリに格納した場合
- CA 2E オブジェクトを統合し、ライブラリの数を減らした場合
- CA 2E ライブラリの名前を変更した場合

第 10 章: アプリケーション インテグレータ

本章では、アプリケーション インテグレータのインストール方法について説明します。アプリケーション インテグレータにより、CA Plex では、さまざまなソースからのリバース エンジニアリング情報が得られます。

アプリケーション インテグレータには、複数のデータ ソースをインポートすることができます。それぞれは、個別にライセンスを必要とする製品です。

- **Schema/ODBC** — どんな ODBC データ ソースからでもデータスキーマを統合できます。これは以前の ODBC インポート ユーティリティに置き換わるものです。
- **Schema/400** — System i からの DDS 定義データスキーマを統合できます。これは、以前の DB2/400 インポート ユーティリティに置き換わるものです。
- **アプリケーション インテグレータ/400** — i5/OS プログラム（ユーザが手動でコーディングしたもの、またはその他のツールによって生成されたもの）を統合できます。
- **アプリケーション インテグレータ/2E** — CA 2E モデルから直接に CA 2E ファンクションを統合できます。

アプリケーション インテグレータの使用の詳細については、オンライン ヘルプで説明されています。

製品に含まれるもの

CA Plex 製品 CD には、以下のものが含まれています。

- アプリケーション インテグレータ ツール
- **APPINT System i** ライブラリ（アプリケーション インテグレータ ディレクトリに保管ファイルとしてインストールされます）

システム要件

アプリケーション インテグレータに特有の要件は、以下のとおりです。

- PC 開発環境。インポートが大規模になると、推奨する最小限のメモリよりも多くのメモリが必要になることがあります。

注: PC 開発環境の詳細については、リリース ノートの「システム要件」セクションを参照してください。

- 必要最小限の **System i** 開発要件。この要件には以下のものが含まれます。
 - i5/OS
 - TCP/IP 接続 (**System i** 接続用)

注: 必要最小限の **System i** 開発要件の詳細については、本書の「**System i** コンポーネント」の章を参照してください。

アプリケーション インテグレータのインストール

ベース プロダクトをインストールする時、あるいはインストール後にアプリケーション インテグレータをインストールするには、以下の手順を実行します。

CA Plex ベース プロダクトのインストール時にアプリケーション インテグレータ ツールをインストール

1. どのタイプのセットアップをインストールするのか指定するように求められた場合には、[カスタム] をクリックします。
2. すでにチェックされているものに加えて、アプリケーション インテグレータをチェックします。

CA Plex ベース プロダクトをインストールした後にアプリケーション インテグレータ ツールをインストール

1. [スタート] - [設定] - [コントロール パネル] の順にクリックします。
2. コントロール パネルで、[アプリケーションの追加と削除] アイコンをダブルクリックします。
3. 製品のリストから [CA Plex] を選択して、[変更と削除] をクリックします。
4. InstallShield Wizard で、[変更] を選択してから [次へ] をクリックします。
5. アプリケーション インテグレータのチェック ボックスを有効にして、[次へ] ボタンをクリックすると、インストールが始まります。

ODBC データ ソースの設定

インストール プロセスの最後に、システム上の ODBC ファイル DNS 用のディレクトリが指定されていないことを示す警告メッセージが表示されることがあります。警告メッセージが表示された場合、手動で以下の手順を実行してください。

ODBC データ ソースの設定

1. Application Integrator ディレクトリで Appint.dsn ファイルを探します。
2. このファイルを、ODBC ファイル DSN が格納されているシステム内のデフォルトディレクトリへコピーします。通常このディレクトリは **C:\Program Files\Common Files\ODBC\Data Sources** ですが、ODBC アドミニストレータを使用してこれを変更することができます。

注: 警告メッセージが表示されなかった場合、これらの手順は必要ありません。

System i からのインポートの準備

System i ベースのデータ ソースからインポートする場合には、まず CA Plex 製品ライブラリ、PLEX700J、およびアプリケーション インテグレータ製品ライブラリ、APPINT を System i 上にインストールしなければなりません。Schema/ODBC または Schema/Biz のモジュールのみを使用している場合には、このセクションを飛ばしてもかまいません。

CA Plex オブジェクトを、次の手順で製品 CD から System i にリストア

1. 保存ファイルを CD から System i に転送します。「System i コンポーネント」の章の「CD から System i への製品ライブラリの転送」セクションの手順を実行してください。

appint.savf ファイルは、アプリケーション インテグレータ ディレクトリにインストールされます。plex.savf は、製品 CD の AS400 Libraries フォルダに収録されています。

2. 製品 CD を使用している場合には、「System i コンポーネント」の章の「保管ファイル (*SAVF) からの製品ライブラリのリストア」セクションの手順を実行してください。
3. APPINT ライブラリをリストアするには、以下のように入力します。

```
RSTLIB SAVLIB(APPINT) DEV(device) ENDOPT(*REWIND) FRCOBJCVN(*YES)
```

CA Plex の複数バージョンの実行

複数のバージョンの CA Plex を実行している場合は、APPINT/APPIN のジョブ記述を、それぞれ適切な PLEX ライブラリおよび APPINT ライブラリのライブラリ名を使用するように変更してください。

第 11 章: よくある質問と回答 (FAQ)

このセクションは、よくある質問とその回答 (FAQ) です。

質問と答え

Q: グループ モデルはいつバックアップしたらよいでしょうか？

A: グループ モデルは毎日定期的にバックアップするようにしてください。グループ モデルは、中核となる設計リポジトリであり、非常に大事なものです。誤ってローカル モデルを破壊してしまった場合、グループ モデルから最後に抽出してから以降に行った作業はすべて失われます。グループ モデルを誤って破壊した場合は、そのグループ モデルの作業はすべて失われます。データの全部または一部をローカル モデルから (XML インポート/エクスポート機能を使用して) 回復させることができる場合がありますが、すべての場合において、回復が保証されているわけではありません。詳細については、オンライン ヘルプの「バックアップ」を参照してください。

Q: 更新頻度はどのくらいですか？

A: CA では、少なくとも 1 週間に 1 回、できればそれ以上の頻度でグループ モデルへの変更を更新することをお勧めしています。ワークグループ環境では、更新を頻繁に行うことによって、他の開発者が変更内容を利用できるようになります。さらに重要なのは、グループ モデルに対する変更を最新のものにしておくと、変更内容が中央リポジトリの一部になり、より安全になるという点です。更新間隔が長くなると、更新にかかる時間に影響することに留意してください。ローカル モデルとグループ モデル間の違いを CA Plex が処理するために必要な作業が増えるほど、時間が長くなることになります。

重要: 数週間も間隔を開けないようにしてください。

Q: ファンクション、フィールド、テーブル、ビューに 2 つの実行名があるのはなぜですか？

A: あなたと誰か別の人がファンクションとパネルまたはテーブルまたはビューを生成して構築した後、グループ モデルを更新したためです。あなたと他の誰かが実行名とファイル名のトリプルとオブジェクトをグループ モデルに追加しましたが、一方のセットを他方のセットに置き換えないでこれを実行しています。このような状態になった場合は、重複した名前を手動で削除しなければなりません。

2 番目の人がグループ モデルを更新しようとしたときにグループ モデルの矛盾が発生することに留意してください。この時点更新しないように選択できます。前もってこのような矛盾が起こらないようにするには、頻繁に変更を更新することが必要です。特にワークグループ内の他の開発者に影響を及ぼすような変更を行った後には必ず更新が必要です。詳細については、オンライン ヘルプの「矛盾の防止と解決」を参照してください。

Q: オブジェクトを生成しようとしたのですが、「実行用言語を指定してください」という内容のエラーが表示されました。何が間違っているのでしょうか？

A: パターン ライブラリ（OBASE など）からオブジェクトを継承した場合には、ローカル モデルの構成が適切であることを確認してください。そのためには、[ファイル] メニューから [構成] を選択し、次に [モデル/ライブラリ] ボックスにある各モデルを順番に選択します。それぞれについて、[バリエーション] オプションが [Base] に設定されていないことを確認してください。またそうする代わりに、現在の作業に対し適切なバリエーションを選択することもできます。

Q: グループ モデルにおけるレベルとバージョンの違いはどのようなものですか？

A: バージョンは、指定された順序におけるレベルの集合です。作業をいろいろなレベルに割り当てると、一般的なフィックスや顧客固有のフィックスを同時に追跡することができます。たとえば、FunkyApp というアプリケーションにおいて、リリース 1.0 の後、顧客 A と顧客 B 向けに 2 つのフィックスを作成し、その後、リリース 1.1 でフィックスの一般リリースを行ったというような場合です。この結果、リリース 1.0 の顧客 A フィックス、顧客 B フィックス、およびリリース 1.1 の両者の 4 つのレベルが存在することになります。次の表は、バージョンとレベルの可能な組み合わせをまとめたものです。詳細については、オンライン ヘルプの「バージョンとレベル」を参照してください。

バージョン	含まれるレベル	説明
リリース 1.0	リリース 1.0	全顧客向けの FunkyApp 1.0 の最初の一般リリース
顧客 A 1.0	リリース 1.0 顧客 A フィックス	FunkyApp 1.0 リリース、顧客 A 専用
顧客 B 1.0	リリース 1.0 顧客 B フィックス	FunkyApp 1.0 リリース、顧客 B 専用
リリース 1.1	リリース 1.0 リリース 1.1	FunkyApp 1.1 一般リリース、A および B 以外の全顧客対象
顧客 A 1.1	リリース e 1.0 顧客 A フィックス リリース 1.1	FunkyApp 1.1 リリース、顧客 A 専用
顧客 B 1.1	リリース 1.0 顧客 B フィックス リリース 1.1	FunkyApp 1.1 リリース、顧客 B 専用

Q: アクション ダイアグラムで定数やリテラル値を使えません。なぜでしょうか？

A: 確かに、コード内のどこにでもリテラルを入力できれば、プログラミングは速くなるでしょう。たいていのプログラミング言語ではこれが行えます。では、なぜ **CA Plex** ではこのような使い方を許していないのでしょうか？理由は保守性ということにあります。さまざまな言語で記述されたきわめて多くのプログラムが存在しますが、これらのプログラムには、その文字や数値の重要性についてまったく説明がないままプログラムの中にコーディングされたリテラルが含まれています。**CA Plex** は、企業向け開発ツールとして、元の開発者以外の人物でも容易にアプリケーションを保守できるように設計されています。このため、簡単にプログラムを理解できるよう、すべてのリテラルに常にラベルを貼り付けることが義務づけられています。

Q: ライブラリ ファンクションの **+++** メタオペレーションとはどのようなものですか？

A: メタ コードは、モデル内のトリプルについて質問し、予想される機能を提供するためにパターン ライブラリで使用するものです。たとえば、トリプル「**MyField default MyDefault**」を含むフィールドは、そのデフォルト値を設定するメタ コードをアクション ダイアグラムでサポートしていないと、デフォルト値を提供しません。メタ コードのメタ オペレーションの詳細については、オンライン ヘルプの[検索]ボタンをクリックし、[目次]タブの関連項目を参照してください。他の人が書いたコードをコピーしたり変更したりすることをとおして、付属のライブラリ コードのカスタマイズに取り組んでみてください。

Q: コードのパターンへの抽象化はいつすべきでしょうか？

A: パターンはわかりやすい場合もありますが、継承可能でカスタマイズ可能なわかりやすいパターンにするためには特別な作業が必要となるので、2 回しかこれを実行しないのであれば、無駄な労力となります。その一方で、明らかにパターンであるべきものは、抽象化が難しいことがあります。ほとんどの場合、答えは、一度実行してみれば（最初に抽出に関して学習するときに実行できます）、これを抽出すべきかどうか、またどのようにしてパターンに変換できるかが、より明確かつ容易に理解できるようになるということです。いつ抽象化すればよいかは、経験を積むことで、簡単にわかるようになります。

Q: アクション ダイアグラムのデバッガにフィールド名が表示されません。なぜでしょうか？

A: **C++** 構築オプションで、[選択したオブジェクトすべての構築の強制]を選択し、次にファンクションを再構築してください。これが正常に動作しない場合は、**Obj** サブディレクトリ全体を削除してから再構築してください。詳細については、オンライン ヘルプの「アクション ダイアグラム デバッグ用の生成と構築」を参照してください。

Q: どうしたらパラメータの対応付けを簡略化できますか？

A: 最も重要な手法の 1 つは、フィールド ドメインの管理です。フィールド ドメインで、CA Plex によるパラメータの自動対応付けを制御します。手動でマッピングを実行すると、フィールド ドメインによって利用可能なフィールドのリストも制限されるので、必要なものを検出しやすくなります。詳細については、オンライン ヘルプの「フィールド ドメイン」を参照してください。

もう 1 つの手法は、[パラメータ マッピング] ダイアログ ボックスでデフォルトのマッピングを使用することです。デフォルトのマッピングにより、ファンクション呼び出し間のパラメータをマッピングするときに、アクション ダイアグラムは自動的に最適な推測を行うことができるので、各パラメータを個別にマッピングする手間を省くことができます。詳細については、オンライン ヘルプの「デフォルトの対応付けの設定」を参照してください。

Q: 前ポイント、後ポイント、編集ポイントの違いは何ですか？ どれを使用したらよいでしょうか？

A: 編集ポイントは、前のバージョンの CA Plex から引き継がれたものです。収集ポイント（前ポイントと後ポイントの総称）は、コードの保管場所を改良したものです。違いは、編集ポイントでは継承ツリーの下に異なるレベルで互いに上書きしますが（ネストには十二分に注意する必要があります）、収集ポイントは協調的で、コードを順番に集めて一緒にすればよいという点です。いずれのアクション ダイアグラムにおいても、常に収集ポイントの末尾にコードを追加することができます。そのコードが前に使われているかどうかには関係しません。編集ポイントには、一度これを使用すると継承ツリーのそれより下位の編集ポイントを使用できなくなるという制限がありました。したがって、今後は編集ポイントを使用しないようにしてください。

それでは、後ポイントと前ポイントのどちらを使用すればいいのでしょうか？ 明白な場合があります。編集ポイント、前ポイント、あるいは後ポイントにおいて、別のコード部分の前または後にコードを置きたい場合、このコードを配置する場所は明らかです。コードが前ポイントまたは後ポイントのいずれに配置されてもかまわない場合はどうなるでしょうか？ その場合、コードは常に、後ポイントに入ります。誰かがファンクションからコードを継承し、これを追加したい場合、そのコードが後ポイントにあれば、コードをその前（前ポイント）にでも、あるいは後にでも追加できます（コードは、それが継承階層の上位で使われているかどうかに関係なく、常に収集ポイントの最後に追加できることを思い出してください）。コードを前ポイントに追加するのがあなたの習慣であれば、コードは、継承階層の下にレベルであなたのコードの後（前ではなく）に追加することしかできません。結論として、特に理由がない限り、常に後ポイントを使用してください。詳細については、オンライン ヘルプの「編集ポイントと収集ポイント」を参照してください。

Q: 使用するのはどの後ポイントですか？

A: これは初心者にとっておそらく、**CA Plex** における、最も難しい問題でしょう。コンピュータ・アソシエイツでは、以下の手順を実行することをお勧めしています。

1. よく考えて推測します。
2. 後ポイントの前後でコードを展開します。
3. これが正しい後ポイントかどうかを考えます。
4. テストして、結果を確認します。
5. うまく動作しない場合には、手順 **3** をもう一度実行し、次に手順 **1**～**5** を繰り返します。

使用する後ポイントを知っていることが、**CA Plex** のファンクションを迅速に記述するためのキーとなることは間違いありません。他のファンクションを調べてみて、開発者が選択したその特定のポイントにローカル修正が追加されている理由を判断します。クラスライブラリとパターン ライブラリは、編集ポイントと収集ポイントの使用をベースにしているので、調査して手がかりを探せるということを忘れないようにしてください。場合によっては、ライブラリからアクション ダイアグラムを開くことも役立ちます。たとえば、**UISTYLE/EditDetailGrid** を調べると、**UISTYLE/Detail** と **UISTYLE/Grid** の統合方法や **UISTYLE/UIBasicShell** の使用方法がわかります。

他の部分コード、とりわけライブラリを調べれば、いつどの後ポイントを使用すべきかについての総合的な理解度を向上させることができます。これによって、**CA Plex** 開発者としての能力が高まります。

Q: 行をウィンドウの最上段の編集ボックスまでドラッグし続けなければならないのはなぜですか？

A: そうする必要はありません。キーボードの **Insert** キーを押してみてください。これでインライン編集が可能になります。複数行コメントについては、**Ctrl** キーと **Enter** キーを同時に押してアクション ダイアグラムに行を入力する必要がありますが、通常の単一行のコードについては、編集が終わった時点で **Enter** キーを押すだけです。

Q: ファンクションからフィールドや変数を削除するにはどうしたらよいでしょうか？

A: モデル エディタでトリプルを削除するとフィールドや変数を削除できます。変数パレットで **Delete** キーを押すだけでは削除できません。ファンクションの先頭行または変数パレット（両方とも、ファンクション名が表示されています）で、**F12** キーを押します。これにより、モデル エディタが呼び出されて、このファンクションがフォーカスされます。削除できるフィールドや変数があれば、そのリストがトリプル内に表示されます。継承フィールドと継承変数は一覧表示されないことに留意してください。継承フィールドを削除するには、継承階層のより上位にあるファンクションが規定しているトリプルを、このファンクションから削除する必要があります。ライブラリ ファンクションのフィールドと変数は削除できません。**F5** キー（リフレッシュ）を押すと、変数パレットが再描画されるので、フィールドと変数が削除されていることがわかります。

Q: どうしたらコメントを記録し、コードに追加することができますか？

A: **Comment** 構造体および **Seq** 構造体がよく使われます。**CA Plex** の長いオブジェクト名を活用し、意味のある名前を付けてください。サブルーチンと編集ポイントの命名には注意が必要です。必要に応じて説明ブロックを使用してください。

Q: フィールドがデュアル（参照による呼び出し）のパラメータであることをどうすれば指示できますか？

A: モデル エディタで、トリプルを **MyFunction input MyField** から **MyFunction dual MyField** に変更してください。アクション ダイアグラマからは変更できません。

Q: パネル デザイナでウィンドウ パネルのサイズを変更しましたが、実行時にサイズが変わりません。どうして変わらないのでしょうか。

A: おそらく、パネルの「**Save Placement**」プロパティが「**Yes**」に設定されていて、前のサイズがアプリケーションの **.ini** ファイルに保存されたままだと思います。変更を確定するには、**.ini** ファイルのエントリを削除する（または**[Exe を作成]**コマンドを使って再構築する）か、「**Save Placement**」を「**No**」に設定してください。

Q: コードは記述されているのに、パネル上のボタンが機能しません。なぜでしょうか？

A: 通常考えられるのは、パネル内の物理イベントが **CA Plex** の論理イベントに対応付けられていないということです。物理パネル イベントは、誰かがパネル上の何かに対してクリック、ドラッグ、押す、ダブルクリック、または選択という操作を行った結果です。たとえば、ボタンを押したりメニュー項目を選択したり、あるいはテキストを入力したりすることはすべて物理イベントとみなされます。これらはすべて、目に見えるコンピュータ画面上で行われるからです。

CA Plex のこれ以外のイベントは、論理イベントです。論理イベントは、**CA Plex** の内部でのみ行われるイベントです。このイベントは、「目に見えるもの」ではありません。その代わりに、**CA Plex** では、物理イベントを（クリックして）論理イベントに結びつけることができます。アクション ダイアグラマと、特にイベント ハンドラは、論理イベントにしか応答できません。

パネル デザイナでボタンを右クリックし（このボタンが選択されているとき）、**[イベントの対応付け]**を選択すると、**[イベントの対応付け]** ダイアログ ボックスが表示されます。左側のリスト ボックス（物理イベント）で「**Pressed**」をクリックし、右側のリスト ボックス（論理イベント）で、対応するイベント（たとえば、**MyButtonPressed** など）が青色で選択されていることを確認します。強調表示されていない場合は、選択後、**[OK]** をクリックして、ダイアログ ボックスを閉じます。アクション ダイアグラマに戻り（**F11** キーを押すとファンクションとそのパネルの間で高速に切り替わります）、**F5** キーを押してリフレッシュします。イベント コードを探してください（**Events Handler** 構造体の中の **Event: EventMyButtonPressed** 構造体）。これは、ボタンを押したときに実行されるはずです。

どの物理イベントがどの論理イベントに対応付けられるかを調べるには、パネル デザイナでパネル パレット内の「**Events**」フォルダを見てください。**1** つの論理イベントに **1** つまたは複数の物理イベントが割り当てられている場合、その論理イベントを展開すれば、割り当てられている物理イベント（**1** つまたは複数）が表示されます。

Q: MDI 親ウィンドウが MDI 子ウィンドウでトリガされた共通イベントに応答しません。なぜでしょうか？

A: おそらく、メニュー項目の「Menu ID」プロパティが設定されていないためです。パネル デザイナで [MyMDI] の [MenuShell] パネルを開き、パネル パレットでメニュー バー フォルダを展開します。すべての子ウィンドウで複製されるが、親ウィンドウのみが応答するメニュー アイテムを見つけて、それらが番号の「MenuId」プロパティを持つことを確認します。1~500、54016、または 54107 の番号は使用しないでください。

Q: Menu ID を設定しているのに、MDI 子パネルが MDI のパネル イベントをトリガできません。なぜでしょうか？

A: MDI の子パネルと MDI の親パネルはすべて、共通の MenuShell パネルから継承されたものであるため、そのすべてを再コンパイルするまでは、子のファンクションが、設定した Menu ID のプロパティ番号を選択することは期待できません。子を再コンパイルすると、個々のパネルは Menu ID 番号を認識できるようになり、これを親に渡すことができます。MDI 親パネルも再コンパイルすることを忘れないでください。親もその Menu ID 番号を認識する必要があるからです。

Q: メニュー項目はどのようにして削除するのでしょうか？

A: Delete キーを押してください。右クリックして開くコンテキスト メニューからメニュー アイテムを削除するオプションはありません。

Q: ステータス フィールドに入力した、新しい値を確認できません。なぜでしょうか？

A: モデル エディタでフィールドに値を追加し、パネルに移動しても、追加された値はフィールド上に表示されません。パネル デザイナを使ってフィールドを選択し、ポップアップ メニューの [値のリフレッシュ] を選択します。値はリフレッシュされ、モデル エディタ内のトリプルと一致するようになります。

Q: メッセージ ログ内のメッセージの全文を表示する方法はありますか？

A: あります。メッセージを右クリックしてください。

Q: 実行した変更が他のウィンドウに反映されません。なぜでしょうか？

A: リフレッシュ ボタン (F5) をクリックすると変更内容が表示されます。場合によっては、変更内容を確認するために、パネルとその関連アクション ダイアグラムの両方をいったん閉じてから (変更を保存するかどうかを尋ねられたら [はい] と答えます)、再度これらを開く必要があります。

Q: モデル エディタでトリプルを追加しましたが、見つけることができません。どうなっているのでしょうか？

A: モデル エディタに表示される特定のトリプルを決定する要因はいろいろあります。それぞれの要因をチェックしてください。詳細については、オンライン ヘルプの「モデル エディタの表示」を参照してください。さらに、以下についても留意してください。

- モデル エディタには、継承された（暗黙の）トリプルは表示されません。
- 継承されたオブジェクトに追加したトリプルは、フォーカスされていないモデル エディタ ウィンドウには表示されない場合があります。これは、継承オブジェクトを参照するトリプルがないからです。このような場合、オブジェクト ブラウザから継承されたオブジェクトをドラッグ アンド ドロップして、モデル エディタをフォーカスします。

Q: CA Plex では不完全な、あるいは非論理的なデータを入力できますが、なぜですか？

A: CA Plex では構文の正しい入力及要求されますが、整合性や完全性については、生成しない限り要求されません。したがって、モデルに、たとえばトリプル「Credit limit length Customer」を含めることはできませんが、「Credit limit length 6」と「Credit limit length 7」は含めることができます。CA Plex では、次のような理由から、このようにしています。

- 設計時には、オブジェクトのプロパティの一部しかわからないことがよくあります。このような状況では、すべてがわからなければ何も記憶できないようなツールは、役に立ちません。
- ワークグループ環境で、複数の異なるプロパティ値を同時に記録しなければならないような場合にも、このような状況が発生します。つまり、2 人のチーム メンバがクレジット限度額フィールドの長さについて意見を異にしている場合があります。CA Plex では、両方のフィールド長を記録し、不一致を後で解消できるようになっています。
- ワークグループ環境では、規則を対話的に強制するのは非効率、あるいは単に不可能だという事実が挙げられます。ある時点で、必要なトリプルがすべてローカル モデルに存在しているとはかぎりません。

Q: モデル エディタに、利用可能な目的語オブジェクト タイプのリストがありません。なぜでしょうか？

A: 入力バー上の目的語オブジェクト タイプは、常に選択した動詞によって値が決まるため、入力することはできません。

Appendix A: Acknowledgments

This appendix provides the third-party license agreements.

Sun Microsystems, Inc. Binary Code License Agreement

SUN MICROSYSTEMS, INC. ("SUN") IS WILLING TO LICENSE THE SOFTWARE IDENTIFIED BELOW TO YOU ONLY UPON THE CONDITION THAT YOU ACCEPT ALL OF THE TERMS CONTAINED IN THIS BINARY CODE LICENSE AGREEMENT AND SUPPLEMENTAL LICENSE TERMS (COLLECTIVELY "AGREEMENT"). PLEASE READ THE AGREEMENT CAREFULLY. BY DOWNLOADING OR INSTALLING THIS SOFTWARE, YOU ACCEPT THE TERMS OF THE AGREEMENT. INDICATE ACCEPTANCE BY SELECTING THE "ACCEPT" BUTTON AT THE BOTTOM OF THE AGREEMENT. IF YOU ARE NOT WILLING TO BE BOUND BY ALL THE TERMS, SELECT THE "DECLINE" BUTTON AT THE BOTTOM OF THE AGREEMENT AND THE DOWNLOAD OR INSTALL PROCESS WILL NOT CONTINUE.

1. DEFINITIONS. "Software" means the identified above in binary form, any other machine readable materials (including, but not limited to, libraries, source files, header files, and data files), any updates or error corrections provided by Sun, and any user manuals, programming guides and other documentation provided to you by Sun under this Agreement. Programs mean Java applets and applications intended to run on the Java 2 Platform, Standard Edition (J2SETM platform) platform on Java-enabled general purpose desktop computers and servers.

2. LICENSE TO USE. Subject to the terms and conditions of this Agreement, including, but not limited to the Java Technology Restrictions of the Supplemental License Terms, Sun grants you a non-exclusive, non-transferable, limited license without license fees to reproduce and use internally Software complete and unmodified for the sole purpose of running Programs. Additional licenses for developers and/or publishers are granted in the Supplemental License Terms.

3. RESTRICTIONS. Software is confidential and copyrighted. Title to Software and all associated intellectual property rights is retained by Sun and/or its licensors. Unless enforcement is prohibited by applicable law, you may not modify, decompile, or reverse engineer Software. Licensee acknowledges that Licensed Software is not designed or intended for use in the design, construction, operation or maintenance of any nuclear facility. Sun Microsystems, Inc. disclaims any express or implied warranty of fitness for such uses. No right, title or interest in or to any trademark, service mark, logo or trade name of Sun or its licensors is granted under this Agreement. Additional restrictions for developers and/or publishers licenses are set forth in the Supplemental License Terms.

4. LIMITED WARRANTY. Sun warrants to you that for a period of ninety (90) days from the date of purchase, as evidenced by a copy of the receipt, the media on which Software is furnished (if any) will be free of defects in materials and workmanship under normal use. Except for the foregoing, Software is provided "AS IS". Your exclusive remedy and Sun's entire liability under this limited warranty will be at Sun's option to replace Software media or refund the fee paid for Software. Any implied warranties on the Software are limited to 90 days. Some states do not allow limitations on duration of an implied warranty, so the above may not apply to you. This limited warranty gives you specific legal rights. You may have others, which vary from state to state.

5. DISCLAIMER OF WARRANTY. UNLESS SPECIFIED IN THIS AGREEMENT, ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT THESE DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

6. LIMITATION OF LIABILITY. TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL SUN OR ITS LICENSORS BE LIABLE FOR ANY LOST REVENUE, PROFIT OR DATA, OR FOR SPECIAL, INDIRECT, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER CAUSED REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF OR RELATED TO THE USE OF OR INABILITY TO USE SOFTWARE, EVEN IF SUN HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. In no event will Sun's liability to you, whether in contract, tort (including negligence), or otherwise, exceed the amount paid by you for Software under this Agreement. The foregoing limitations will apply even if the above stated warranty fails of its essential purpose. Some states do not allow the exclusion of incidental or consequential damages, so some of the terms above may not be applicable to you.

7. SOFTWARE UPDATES FROM SUN. You acknowledge that at your request or consent optional features of the Software may download, install, and execute applets, applications, software extensions, and updated versions of the Software from Sun ("Software Updates"), which may require you to accept updated terms and conditions for installation. If additional terms and conditions are not presented on installation, the Software Updates will be considered part of the Software and subject to the terms and conditions of the Agreement.

8. SOFTWARE FROM SOURCES OTHER THAN SUN. You acknowledge that, by your use of optional features of the Software and/or by requesting services that require use of the optional features of the Software, the Software may automatically download, install, and execute software applications from sources other than Sun ("Other Software"). Sun makes no representations of a relationship of any kind to licensors of Other Software. TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL SUN OR ITS LICENSORS BE LIABLE FOR ANY LOST REVENUE, PROFIT OR DATA, OR FOR SPECIAL, INDIRECT, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER CAUSED REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF OR RELATED TO THE USE OF OR INABILITY TO USE OTHER SOFTWARE, EVEN IF SUN HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Some states do not allow the exclusion of incidental or consequential damages, so some of the terms above may not be applicable to you.

9. TERMINATION. This Agreement is effective until terminated. You may terminate this Agreement at any time by destroying all copies of Software. This Agreement will terminate immediately without notice from Sun if you fail to comply with any provision of this Agreement. Either party may terminate this Agreement immediately should any Software become, or in either party's opinion be likely to become, the subject of a claim of infringement of any intellectual property right. Upon Termination, you must destroy all copies of Software.

10. EXPORT REGULATIONS. All Software and technical data delivered under this Agreement are subject to US export control laws and may be subject to export or import regulations in other countries. You agree to comply strictly with all such laws and regulations and acknowledge that you have the responsibility to obtain such licenses to export, re-export, or import as may be required after delivery to you.

11. TRADEMARKS AND LOGOS. You acknowledge and agree as between you and Sun that Sun owns the SUN, SOLARIS, JAVA, JINI, FORTE, and iPLANET trademarks and all SUN, SOLARIS, JAVA, JINI, FORTE, and iPLANET-related trademarks, service marks, logos and other brand designations ("Sun Marks"), and you agree to comply with the Sun Trademark and Logo Usage Requirements currently located at <http://www.sun.com/policies/trademarks>. Any use you make of the Sun Marks inures to Sun's benefit.

12. U.S. GOVERNMENT RESTRICTED RIGHTS. If Software is being acquired by or on behalf of the U.S. Government or by a U.S. Government prime contractor or subcontractor (at any tier), then the Government's rights in Software and accompanying documentation will be only as set forth in this Agreement; this is in accordance with 48 CFR 227.7201 through 227.7202-4 (for Department of Defense (DOD) acquisitions) and with 48 CFR 2.101 and 12.212 (for non-DOD acquisitions).

13. GOVERNING LAW. Any action related to this Agreement will be governed by California law and controlling U.S. federal law. No choice of law rules of any jurisdiction will apply.

14. SEVERABILITY. If any provision of this Agreement is held to be unenforceable, this Agreement will remain in effect with the provision omitted, unless omission would frustrate the intent of the parties, in which case this Agreement will immediately terminate.

15. INTEGRATION. This Agreement is the entire agreement between you and Sun relating to its subject matter. It supersedes all prior or contemporaneous oral or written communications, proposals, representations and warranties and prevails over any conflicting or additional terms of any quote, order, acknowledgment, or other communication between the parties relating to its subject matter during the term of this Agreement. No modification of this Agreement will be binding, unless in writing and signed by an authorized representative of each party.

RSA Security

This product includes code licensed from RSA Security, Inc.

Some portions licensed from IBM are available at
<http://oss.software.ibm.com/icu4j/>

decNumber 3.32

This product contains Runtime Modules of decNumber (c) Copyright IBM Corporation 2001, 2004. All Rights Reserved.

Apache ANT

Portions of this product include software developed by the Apache Software Foundation. The Apache software is distributed in accordance with the following license agreement:

Apache License

Version 2.0, January 2004

<http://www.apache.org/licenses/>

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

'License' shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

'Licensor' shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

'Legal Entity' shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition,

'control' means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

'You' (or 'Your') shall mean an individual or Legal Entity exercising permissions granted by this License.

'Source' form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

'Object' form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and versions to other media types.

'Work' shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

'Derivative Works' shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

'Contribution' shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, 'submitted' means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as 'Not a Contribution.'

'Contributor' shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

(a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

(b) You must cause any modified files to carry prominent notices stating that You changed the files; and

(c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

(d) If the Work includes a 'NOTICE' text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions.

Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an 'AS IS' BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.