CA Plex

Getting Started

r6.1

CA Product References

This document references the following CA products:

- CA Plex

Contact Technical Support

Contact Technical Support

For online technical assistance and a complete list of locations, primary service hours, and telephone numbers, contact Technical Support at http://ca.com/support (http://www.ca.com/support).

Provide Feedback

If you have comments or questions about CA product documentation, you can send a message to techpubs@ca.com (mailto:techpubs@ca.com).

If you would like to provide feedback about CA product documentation, please complete our short customer survey (http://tinyurl.com/6j6ugb), which is also available on the CA Support website.

Contents

Chapter 1: Introduction

Chapter 2: Installing CA Plex

Chapter 3: Develop Your First Java Application in 20 Minutes

Chapter 4: Group Model Licensing

Chapter 5: Pattern Libraries

Chapter 6: System i Components

Chapter 7: Windows C++ Server Components

Chapter 8: Java Components

Chapter 9: CA 2E Data Migration

Chapter 10: Application Integrator

Chapter 11: Frequently Asked Questions

Appendix A: Acknowledgments

Chapter 1: Introduction

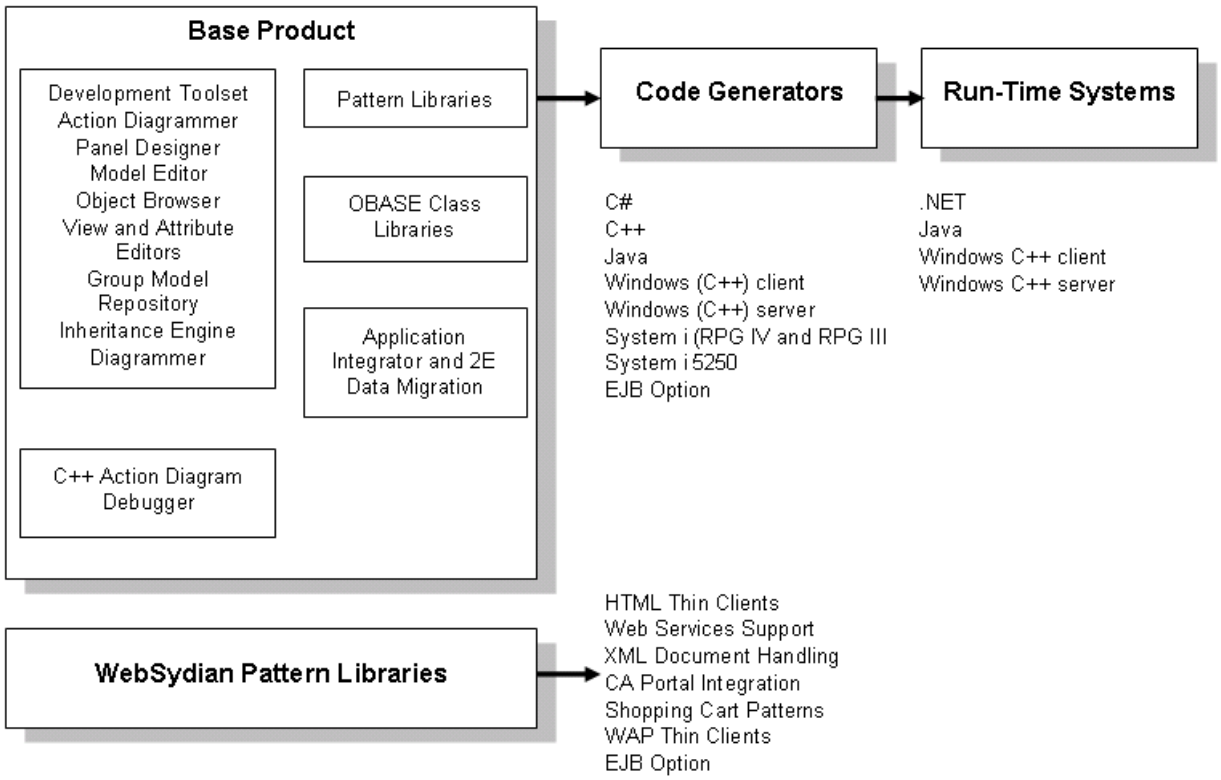
CA Plex is an Architected Rapid Application Development (ARAD) tool that uses software patterns, models, and generators to accelerate the design and maintenance of business applications for the Java, J2EE, .NET, and System i platforms.

Begin Quickly

Follow the basic installation steps described in the chapter "Installing CA Plex" to begin quickly. You will then be able to run through the introductory tutorial in the chapter "Develop Your First Java Application in 20 Minutes."

Product Components

The following diagram shows the main components of CA Plex.



CA Plex includes the following:

- A Windows-based visual development environment, complete with GUI screen designers, a language-neutral business logic editor, diagrammer, and impact analysis tools.
- A multi-developer repository with built-in configuration management for storing design models across multiple versions, languages, and platforms.
- Code generators that automatically create 100 percent of C#, Java, C++, HTML, RPG, or SQL code required for implementing applications for the Web, wireless, Java, System i, and .NET platforms. CA Plex generates the entire application, including HTML and GUI clients, 5250 host screens, server programs, and database objects.

CA Plex minimizes the need for platform expertise, which insulates developers from technology changes and lets them focus on solving business problems rather than technical problems.

Applications generated by CA Plex support a rich set of industry standards and technologies including XML, J2EE, J2SE, WAP, Web Services, Microsoft SQL Server, Oracle, DB2, and Linux.

Supported Platforms

CA Plex creates native, n-tier code for the following platforms:

- Java Server—Java Servers with JDBC data access are supported across multiple platforms including Linux, Solaris, and i5/OS. EJB Option for J2EE support.
- .NET Server—C# code generation with a .NET runtime system that uses OLE DB for data access to industry standard databases.
- System i Server—System i Servers are supported with RPG IV or RPG III code generation and DDS access to DB2/400.
- XML-based web services—XML document-based web services support through the Websydian pattern libraries.
- HTML web client—HTML web clients through the Websydian pattern libraries.
- Rich GUI client—Rich GUI client supports both Win32 and Java (Swing) platforms and ActiveX and JavaBean GUI components. It supports access to all server platforms, plus stand-alone client data access through ODBC or JDBC to the industry standard databases.

- System i 5250—System i 5250 with RPG IV or RPG III code generation and DDS access to DB2/400. It includes drag-and-drop 5250 screen designer.
- Windows C++ Server—Windows C++ Servers provide support for Oracle (through OCI) and Microsoft SQL Server (through ODBC). They also include extensive COM integration support.

Additional Resources

After reading this *Getting Started*, you can refer to the numerous resources available to you for additional information. In particular, there are three tutorials that take you step-by-step through the creation of a simple application:

- Tutorial for Windows
- Tutorial for System i 5250
- Develop Your First Java Application in 20 Minutes

CA Plex offers detailed online help that provides comprehensive information and detailed instructions about how to use CA Plex.

Chapter 2: Installing CA Plex

For the minimum hardware and software requirements for developing applications with CA Plex, see the readme.

Installation Checklist

Installation requirements for CA Plex vary depending on your target development platforms and include third-party products that are not supplied with the product.

Note: For more information about detailed version and other system requirements, see the readme.

Compatibilities are subject to change. For the latest information on CA Plex compatibility with third-party products, check the support website http://ca.com/support/, and select CA Plex from the Product List drop-down menu.

Use the following table to determine which components you need to install and where to find the corresponding installation instructions.

Component	Description	See
Base Product		
CA Plex base product	Includes the CA Plex development environment, help files, sample models and tutorial models.	Install the Base Product in this chapter
CA License Key	License keys are required to use CA Plex and its optional components. The optional components are as follows: ■ EJB Option ■ System i 5250 generator ■ Websydian pattern libraries	Entering the CA License Key in this chapter

Component	Description	See
Pattern and Class Libraries		
Pattern libraries	The latest pattern libraries, recommended for most application development with CA Plex.	"Pattern Libraries" chapter in this guide
Class libraries (OBASE)	The first set of pattern libraries developed for CA Plex; not recommended for new development projects, except for the System i 5250 platform.	classlib.chm help file
Tools and Utilities		
Application Integrator	Used for reverse engineering information from a variety of data sources into CA Plex.	Install the Application Integrator in this guide
CA 2E data migration	Used for importing schema from CA 2E data models into CA Plex.	CA 2E Data Migration Requirements in this guide
Microsoft Word Viewer	Used for viewing and printing generated Windows reports.	Specifying your Print Program in the online help's *User Guide*
Adobe Acrobat	Used for viewing the online versions of the tutorial guides.	Install Adobe Reader in this guide

Component	Description	See
Platform-Specific Requirements		
C# server generator	■ Microsoft .NET Framework Version 2.0 or later. Microsoft .NET Framework Version 3.5, required for WCF development, is supplied on the CA Plex product CD. ■ DBMS (Microsoft SQL Server, Oracle, IBM DB2) and OLE DB Provider. ■ ODBC driver for database builds, not required at run time **Note:** Microsoft Visual Studio 2005 is *not* required for C# development. Optionally, you may find it useful for debugging and other development activities. Microsoft Visual Studio 2008 is the version recommended for WCF development.	ObNET.chm help file.

Component	Description	See
Java generator	■ Sun J2SE Development Kit 6.0 ■ Apache ANT (automatically installed with CA Plex) ■ Third-party DBMS such as Microsoft SQL Server, Oracle, IBM DB2, or other third-party DBMS and JDBC driver. ■ ODBC driver for database builds, not required at run time. ■ The JAVA_HOME environment variable must be defined to enable Java builds.	"Java Components" chapter in this guide.
EJB Option	■ J2EE application server such as WebLogic, JBOSS, or WebSphere. The J2EE SDK includes an application server (the Reference Implementation) that is useful for testing purposes. ■ Java 2 Enterprise Edition SDK 1.4. ■ J2SE SDK. This is the same as the requirement for the Java generator. ■ JDBC driver and DBMS. This is the same as the existing requirement for the Java generator.	"EJB Option" in the online help Java Platform Guide

Component	Description	See
Windows C++ client	■ Microsoft Visual Studio 2005 Standard edition or higher **Note:** The Express edition of Visual Studio is not compatible with CA Plex. ■ A deployment platform of Windows XP or Windows Vista ■ CA Plex WinC Run-time system ■ ODBC driver and database (Optional)	Microsoft Visual C++ Compiler
Windows C++ server (WinNTC generator)	■ CA Plex Windows Application Server ■ Microsoft Visual Studio 2005 ■ DBMS (Microsoft SQL Server, Oracle, IBM DB2) ■ Windows Server 2003 or Windows Server 2008	Install the Windows Server Components in this guide
System i (System i 5250 and System i client/server generators)	■ CA Plex System i libraries ■ TCP/IP ■ For more information about i5/OS supported version, see the readme ■ RPG IV or RPG III compiler	"System i Components" chapter in this guide
Open Database (ODBC)	Requires an ODBC driver and DBMS.	"ODBC Components" chapter in this guide

Note: For more information about upgrading from earlier versions, see the *Release Notes*.

Before installing CA Plex (or any of its components) or migrating from a previous product release, be sure to review the following section carefully to determine if there are additional installation or configuration steps you need to follow for this release.

Install CA Plex with Unicenter Software Delivery

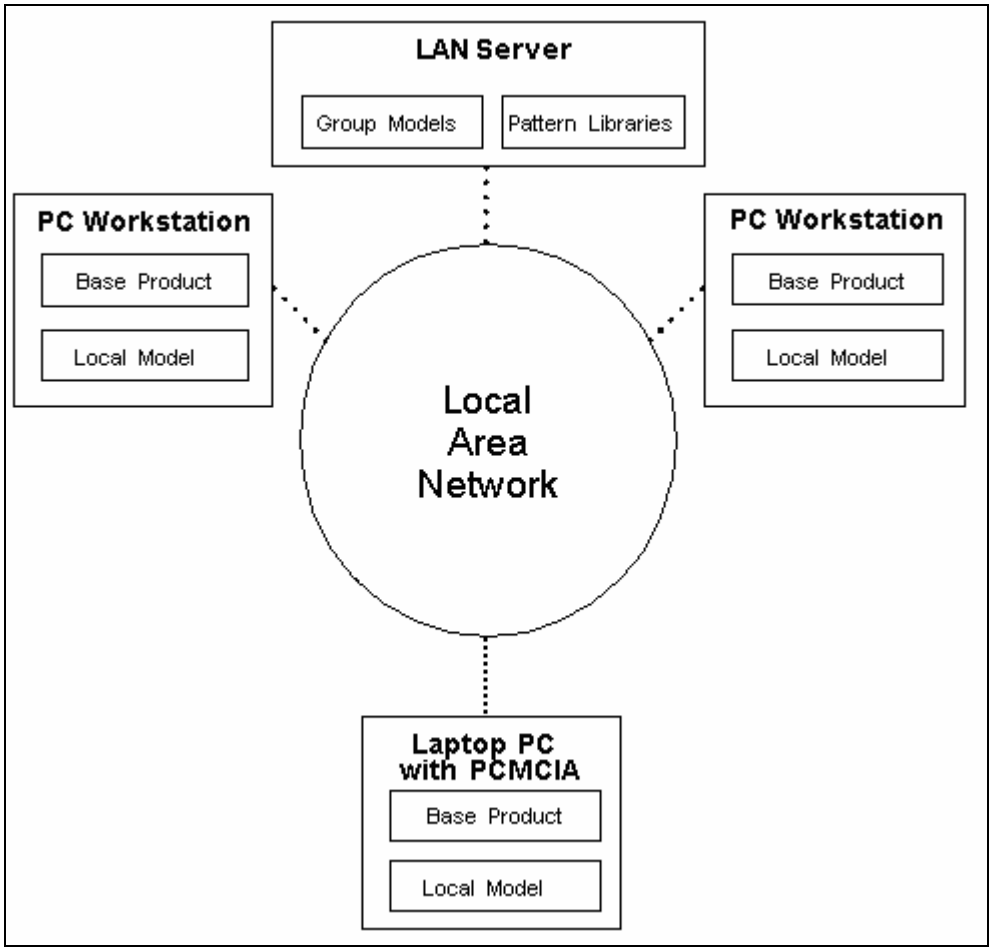
You have an option of installing CA Plex with Unicenter Software Delivery (USD). A USD package is provided for this purpose on the CA Support web site at http://ca.com/support.

To install the CA Plex with USD

1. Download the USD package from the CA Support web site.
2. Unzip the USD Package to your SDO Server.
3. Register the USD Package with your SDO Server.
4. Deploy the USD Package to your desired system.

Typical Configuration

You can configure CA Plex in various ways. The following diagram shows a typical placement of CA Plex components in a LAN environment:



Your group models and pattern libraries reside on the LAN server. The base product and the local models reside on the developer workstations. They may also reside on a laptop with a connection to the network. You only need to connect to the network when you are working on the group model.

Install the Base Product

You must install the base product to start working with the product and develop your first application.

Note: You must have administrator rights to install CA Plex. If you try to install CA Plex without being logged as an administrator, you can get an error 1628: Failed to complete installation from InstallShield.

To install the CA Plex base product on a development workstation

1. Insert the CA Plex product CD into a CD-ROM drive.

   The CA Plex Installation CD window appears.

2. Click the Read button to read the CA Plex documentation, including the readme, and review the installation and upgrade information.

   Note: The readme contains additional, late-breaking information not covered in this guide.

3. Click the Install button to begin the installation.

4. Initiate one of the following installations from the displayed menu:

   CA Plex

   Installs the CA Plex base product.

   CA Plex Windows C++ Application Server

   Installs the CA Plex Windows C++ Application Server necessary to deploy a WinNTC application on a Windows server. This option excludes build tools required for development. To install the build tools, run a custom installation of the base product.

   Websydian Web Libraries

   Installs third-party libraries to enable your CA Plex applications for the Web.

   Microsoft .NET Framework 3.5

   Installs Microsoft .NET Framework 3.5. You may need to reboot at the end of the install. Microsoft .NET Framework 3.5 installation is mandatory for the WFC service generation feature.

Word Viewer

Installs a version of Word Viewer, which you can use to view and print CA Plex C++ run-time reports.

Note: If you have Microsoft Office installed, there is no need to install Word Viewer.

Java JDK 6.0

Links to the Sun website to download the Java Development Kit (JDK) 6.0, which is a mandatory component for Java development within CA Plex. It is not required for C#, Windows C++, or System i RPG development.

If you clicked *CA Plex,* the InstallShield Wizard window appears.

5. Click Next.

   The License Agreement window appears.

6. Read the License Agreement. Select *I accept the terms of this license agreement option* if you accept the terms of the agreement, and click Next.

   The Personal Information dialog appears.

7. Enter your name and your company name, and Click Next.

   The Choose Destination Location dialog appears

8. Specify which type of setup you want to install:

   ***Typical* configuration**

   Installs all the program files, including help files, tutorial files, sample files, and pattern libraries. With a Typical installation, the OBASE class libraries and Windows C++ Server components are not installed by default; You must select Custom install to install them.

   ***Compact* configuration**

   Saves disk space by installing only the program files and the pattern libraries, but not the help files, tutorial files, or sample files.

   ***Custom* configuration**

   Lets you specify which product components to install.

9. Specify a destination location, and Click Next. Either you can accept the default, C:\Program Files\CA\CA Plex r6.1, or you can click Change to install to a different directory.

   In addition to the specified destination directory, CA Plex also installs a copy of Sun's JRE in the Program Files\CA\SharedComponents\Jre directory.

   Note: The installation program prevents you from installing this release into the same directory as the previous release of CA Plex. Instead, you should install this release into a different directory or uninstall the previous version.

10. If you selected Custom, select the components to install. The default settings for the Custom installation are the same as for the Typical installation, but you can also select Class Libraries, Application Integrator, and CA Plex Windows C++ Application Server SDK.

11. Select which language you want to use as the default for generated Windows clients. You can change this setting later.

    Note: For more information, see the online help topic Specifying the National Language Library.

12. Click Back to return to any dialog in which you want to change the settings, or click Next to begin the installation.

13. Review your settings before copying files, and click Finish when the installation is complete.

    CA Plex is installed on your computer.

14. If prompted, select whether to restart your computer now or later and click OK.

If you experience an installation error when running CA Plex setup, enter the following command from Windows command prompt:

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

To run this command on Windows Vista, you must start the Windows Command Prompt as an Administrator. This command creates a PlexInstall.log file in the root of the C: drive containing information that helps diagnose the problem.

In the previous example, the F: drive is the CD/DVD drive containing the CA Plex installation CD.

In the case of the CA Plex Windows C++ Application Server, the command line is as follows:

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

Silent Installation

A silent installation is useful when you need to install the product automatically without any user intervention. To install the product in silent mode, use the /s command line switch with the setup.exe command. For example, enter the following command from the command line:

```
d:\Plexsetup\setup.exe /s
```

or

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

where d:\ is the drive letter for your CD-ROM.

The first command performs a typical installation of the CA Plex product to the default installation directory on 32-bit Windows. The second command performs an installation on 64-bit Windows.

If you run the silent install using a command-line window on Windows Vista or Windows 2008, you must open the command-line window prompt as an administrator, even if you are already logged on as an administrator.

To uninstall CA Plex in silent mode, use the following command.

On 32-bit Windows:

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

On 64 bit Windows:

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

Recording your own iss file

The shipped setup.iis file is recorded so as to install CA Plex to the default location with the *Typical* install option. However this may not meet your requirements.

You can record your own setup.iss file by using the /r command-line switch with the setup.exe command. Typically, the setup.exe will create the setup.iss file in the C:\Windows folder.

You can rename the setup.iss file and specify it with the /f1 command-line switch when running setup.exe.

Modify the System Path

The CA Plex base product installation automatically makes changes to your system path. Specifically, the CA Plex Bin directory (typically, C:\Program Files\CA\AllFusion\CA Plex r6.1\Bin) is added to the system path.

To view or change the system path, double-click the System icon in the Control Panel and alter the environment settings.

Side-by-side installation of CA Plex r6.1 with CA Plex r6.0

The CA Plex r6.1 C++ runtime is backwards compatible with CA Plex r6.0. This means that C++ functions generated with CA Plex r6.0 can be used with CA Plex r6.1 version of the runtime without needing to be recompiled.

Note: There is no forward compatibility - do not use the CA Plex r6.0 runtime with CA Plex r6.1 functions because runtime errors will occur.

Both the 6.1 and 6.0 C++ runtime DLLs have the same names. For example, ob600lc.dll, ob600nwi.dll and so on. When the CA Plex install modifies the system path, the Plex 6.1 Bin folder is put at the beginning of Path statement. This means that, by default, both 6.0 and 6.1 WinC applications will run with the 6.1 WinC runtime DLLs on your PC.

Install Adobe Reader

To view the online versions of the tutorials, you must install Adobe Reader.

To install Adobe Reader

1. Click Documents, Install Adobe Acrobat on the CA Plex Installation CD window.
2. Follow the instructions displayed on the dialog.

Install the Java Development Kit

To develop applications using Java within CA Plex, you must download and install the J2SE Development Kit (version 5.0).

To install Java Development Kit

1. Click Install to display more installation options on the CA Plex Installation CD window.
2. Click the Download JDK 6.0 button.
3. Follow the instructions on the website to download and install JDK 6.0.

Note: If you install the JDK component after installing CA Plex, you must perform any Java-dependent CA Plex configuration changes manually.

Enter the CA License Key

To complete the licensing of CA Plex, place the information in the Execution Key from the ALP Key Certificate in the ca.olf file on the machine running CA Plex. If you have access to the World Wide Web, you can get your updated ca.olf file by going to http://ca.com/support/. Otherwise, follow the directions in this section.

Create the CA License Key File

If the ca.olf does not exist in the CA_LIC folder (usually located in Program Files\CA\SharedComponents), create a new ca.olf file.

To create a ca.olf file

1. Create a new file using a text editor.
2. Copy all of the information from the Execution Key into the new file.
3. Click Save and select the CA_LIC directory.

   The ca.olf file is created. Now follow the instructions to edit an existing CA license key file.

Edit an Existing CA License Key File

If the ca.olf file exists, open it using a text editor of your choice and make the edits.

To edit the ca.olf

1. Open the ca.olf file using a text editor, such as Notepad.
2. Replace all lines beginning with ID_ with the ID_ lines indicated in the Execution Key.
3. Go to the bottom of the file and immediately following any existing FEATURE lines, add the FEATURE line from the Execution Key.

   Note: Do not remove any existing FEATURE lines.
4. Select Save As from the File menu to save the edited ca.olf file in the CA_LIC folder.

Note: The FEATURE line can wrap to a second line on the certificate, but it must be entered on a single line with no carriage return in the ca.olf file.

Frequently Asked Questions on CA Licensing (ALP)

Q: What is ALP?

A: ALP is the Automated License Program, the newest phase of CA licensing and order fulfillment. Rather than requiring clients to use a client-side application to identify hardware eligible to run CA software and request/create the appropriate license file, ALP products are shipped with a printed certificate representing their license file (based on the product and hardware information recorded in our license database), and are also given the opportunity to get their license keys electronically using our support website.

Q: How can I get my ALP license keys online?

A: You can get ALP keys from http://www.ca.com/support. To log on to the secure area of the website, you need a valid user ID and password.

To get ALP keys

1. Log on to supportconnect using your user ID and password.
2. Click the Licensing link under Downloads on the left pane.

   The CA Licensing page appears.
3. Click the ALP keys link under What's New to view your ALP keys.

   The ALP Keys page appears.

   Note: You can also install your ALP keys directly to your local machine by clicking the Install button.

Q: Why does CA licensing sometimes install into the Program Files folder and not C:\CA_LIC as it always has?

A: The latest CA licensing installation conforms to the CA installation standard by installing in the C:\Program Files\CA\SharedComponents\CA_LIC folder if no previous C:\CA_LIC licensing folder is found on a machine. If, however, a C:\CA_LIC folder is found on the machine, the new licensing will continue installing into that folder. Licensing packages that follow the installation standard are versions 1.52 and higher.

The following issue may occur when downgrading licensing:

If a 1.52 or higher version of licensing is installed on a machine with no previous licensing installed on it, then an earlier version of licensing is installed (downgrade of licensing), the earlier version automatically installs into the C:\CA_LIC folder, which creates two CA_LIC folders (C:\CA_LIC and C:\Program Files\CA\SharedComponents\CA_LIC). To fix the problem, install the latest posted version of licensing from the Total License Care (TLC) support site. This merges the two folders into C:\Program Files\CA\SharedComponents\CA_LIC. Over time, C:\Program Files\CA\SharedComponents\CA_LIC becomes the folder where licensing is managed.

Q: How do I use the mergeolf utility?

A: Syntax: MERGEOLF -n <new_olf> -c <current_olf> -o <output_olf> -b <backup_olf> -d

Where:

-n: Is the name of the new OLF file to merge

-c: Is the name of the current OLF file to merge
Default: ca.olf

-o: Is the name of the new OLF file to create
Default: ca.olf

-b: Is the name of the backup of the current OLF file
Default: ca.bak

-d: Enables debugging (mergeolf.log)

Example:

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

Q: What is the correct format of an ALP key?

A: Following is an example of an ALP Execution Key:

```
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
```

The actual encrypted Execution Key is line 7, which is referred to as the FEATURE line, the ID and Server and Daemon lines are header lines that should only appear once in the file, at the beginning. Additional instances of these lines can invalidate your license file.

If you are having problems with your ALP key, verify that the following:

- There are no carriage returns interrupting the FEATURE lines in the ca.olf file.
- There are no blank lines in between the FEATURE lines in the ca.olf file.
- The ID lines of the newly generated ALP Execution Key replace any pre-existing ID lines in the olf.ca file.
- The name of the OLF is ca.olf and does not have an extra extension in the name; for example, ca.olf.txt or ca.olf.olf (This is most common on Windows systems when hide known file extensions is enabled in the Explorer).

Q: When I download certain CA data files and utilities from the support site, they come with an extension of .caz. What is a .caz file?

A: Files downloaded with a .caz extension are compressed with CAZIPXP, available at http://supportconnectw.ca.com/public/ca_common_docs/ca_utility_supp.asp. To uncompress the file, use the command: **cazipxp.exe -u <*filename.caz*>**, which extracts the compressed files to the local directory.

Q: I have installed my ALP license key and I am receiving the following error:

```
- 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
```

A: Typically, this is due to an invalid or corrupt lic98.cap file: open the lic98.cap file with a text editor, the file should be comprised of a listing of detected model types in the format of:
SYS (detectedmodel) (tier) (checksum)

- If the file is improperly formatted, you can copy the lic98.cap from your installation source or download the latest data files at our support website
- If the lic98.cap file does not exist, you may not have a current ALP license run-time installed. Contact TLC for assistance
- If the lic98.cap exists and appears to be valid, contact TLC for assistance.

Modify, Repair, or Remove the Base Product

After installing the CA Plex base product, you can rerun the installation to modify, repair, or remove the base product.

To modify, repair, or remove the CA Plex base product

1. Click Start, Settings, and Control Panel from your computer.

   The Control Panel window appears.

2. Double-click the Add/Remove Programs icon.

   The Add or Remove Programs window appears.

3. Select CA Plex from the list of products, and click Add/Remove Programs.

   The Modify, repair, or remove the program dialog appears.

4. Select Modify, Repair, or Remove, as required.

   The Select Features dialog appears.

5. Click Next, and follow the instructions displayed on the dialog.

Install Other Components

The following sections explain how to install other components.

Install the Windows Application Server

You can choose to install the application server tool when you install the CA Plex base product or after you install the CA Plex base product. This product option is required for Windows C++ Server development. It is not required for C# development.

To install the application server tool when you install the CA Plex base product

1. Click Custom when asked to specify which type of setup to install.
2. Check CA Plex Windows Application Server in addition to what is already checked.

To install the application server after you install the CA Plex base product

1. Click Start, Settings, and Control Panel.
2. Double-click the Add/Remove Programs icon on the Control Panel.
3. Select CA Plex in the list of products.

4. Click Change/Remove.
5. Select Modify and click Next on the InstallShield Wizard window.
6. Click CA Plex Windows Application Server to check it.
7. Click Next to start the installation.

Install the Application Integrator

You can choose to install the Application Integrator tool when you install the CA Plex base product or after you install the CA Plex base product.

To install the Application Integrator tool when you install the CA Plex base product

1. When asked to specify which type of setup to install, click Custom.
2. Check Application Integrator in addition to what is already checked.

To install the Application Integrator tool after you install the CA Plex base product

1. Click Start, Settings, and Control Panel.
2. Double-click the Add/Remove Programs icon on the Control Panel.
3. In the list of products, select CA Plex.
4. Click Change/Remove.
5. On the InstallShield Wizard window, select Modify, and click Next.
6. Click Application Integrator to check it.
7. Click Next to start the installation.

Microsoft Visual Studio 2005

The Microsoft Visual Studio 2005 C++ compiler must be installed to generate and build C++ code (WinC and WinNTC objects) in CA Plex. The Standard edition, Professional edition, and other higher editions of Visual Studio are compatible with CA Plex.

CA Plex only requires a typical installation for Microsoft's Visual Studio. For more information about different configurations, see the Visual C++ documentation.

Note the following:

- The C#, System i 5250, and Java generators do not require the Microsoft Visual C++ compiler.
- If you are building Windows server (WinNTC) functions, you must install Visual Studio on the machine on which you are building the server functions.

The Microsoft Visual Studio 2005 Express edition is not compatible with CA Plex C++ builds because it does not support MFC.

Chapter 3: Develop Your First Java Application in 20 Minutes

CA Plex is an Architected RAD tool that uses patterns to accelerate the design, creation, and maintenance of software applications. This chapter provides an introduction to some of the innovative features of CA Plex.

In this chapter, you define a very simple project management application. The pattern libraries in CA Plex are used to streamline the process. The Panel Designer is used to make changes to a dialog. Before the end of this chapter, you will have the first part of the application up and running.

The example in this chapter creates a Java application. You do not need to know anything about Java programming to complete the example. The development process is very similar regardless of which platform you are developing for.

Prerequisites

Before creating a Java application, you must perform the following tasks:

- Install CA Plex
- Install Sun JDK 6.0, as described in the "Java Components" chapter of this guide
- Define the JAVA_HOME environment variable as described in the "Java Components" chapter of this guide.

Additional Prerequisites for 64-bit Windows

If you want to run this tutorial for 64-bit Windows, consider the following additional prerequisites:

- **The tutorial uses a 32-bit Microsoft access ODBC driver and the JDBC—ODBC bridge. Hence, you must install and use the 32-bit version of the JDK to execute the tutorial application.**
- These considerations do not apply to typical production applications on 64-bit systems that use Oracle, IBM DB2, or Microsoft SQL Server. In such cases, the 64-bit JDK and native 64-bit JDBC drivers are recommended.

Additional Prerequisites for Windows Vista and Windows Server 2008

When you run the tutorial application on Windows Vista and Windows Server 2008, you may encounter a problem with updates to records failing. This occurs due to limitations associated with the use of the default Microsoft Access database. Though it does not prevent you from completing the tutorial, you can use an alternative database such as Microsoft SQL Server Express edition to avoid this limitation.

Concepts

The following section explains the concepts used in the application environment.

Reusable Business Patterns

CA Plex developers model applications by reusing *patterns*. A pattern describes a solution to a recurring problem in software systems. Patterns are abstract descriptions that can be reused in many contexts. An example of a pattern is Structure.

The Structure pattern provides the database schema, user interface designs, and programs that can be used to implement hierarchical data structures such as a bill of materials, an organization chart, or a chart of accounts. CA Plex includes libraries of such patterns, and additional libraries are available from third-party vendors.

Several of the patterns from the CA Plex pattern libraries are used to build the sample application.

Workgroup Development

CA Plex provides a model-based *workgroup environment* in which teams of software developers can collaborate on the design and construction of applications. At the heart of this environment is a repository whose facilities include the following:

- Multiple developer support that enables developers to work offline and then update their changes to the central repository
- Built-in configuration management that enables a model to store a single application design in multiple versions, platforms, and national languages
- The integration of designs stored across multiple models

For simplicity, the sample in this chapter focuses on a single-user environment. The single-user model, called a *local model*, has already been created for you.

Code Generation

Based on the designs held in the repository, CA Plex automatically generates 100 percent of the code to implement applications and database objects across several platforms. Currently supported implementation options include the following:

- Multitiered web applications with Java or HTML clients and Java, .NET , or System i servers
- Multitiered client/server applications with Win32 C++ clients and Java, .NET, or System i servers
- System i 5250 character-based terminal applications

CA constantly improves and expands the available generators to keep pace with customer demand and advances in technology. Generators insulate you from the underlying technology and its implementation details—CA Plex users can take advantage of new platforms simply by regenerating their existing designs.

Terms

In this chapter, you create an *entity* called Project. The Project entity uses *fields* to store information, such as its start and end dates. You inherit *functions* to enable end users to create, modify, and delete projects. Entities, fields, and functions are all types of CA Plex *objects*.

A *local model* is the file that stores the design of the application you are building. In your local model, you create and define objects.

The *Object Browser* displays all of the objects in a model. In it, you can see the pattern library objects available, as well as the objects that you define in your model.

Object Types

CA Plex uses many types of objects. The following shows the icon, the object type, and the abbreviation, and a description of the objects that CA Plex uses to identify each object type that you encounter while building the application in this chapter.

Object Type	Abbreviation	Icon	Description
Entity	ENT		Entities are the tangible objects, such as customers and orders, and intangible objects, such as projects and job titles, about which you want to record information.
Field	FLD		Fields are the pieces of information that you want to record about an entity, such as employee names, item prices, and order statuses.
Function	FNC		Functions are processes that define the functionality of an application. A function is roughly equivalent to a program or method.
Panel	PNL		Panels are the windows and dialogs that make up the user interface of the application.
Table	TBL		Tables are the physical structures in which data is stored in applications that use relational databases.
View	VW		Views represent all or part of the data in a database table. A view does not contain any data—you can think of it as a window through which you look at the data in the table.
Diagram	DGM		Diagrams visually represent a subset of the objects in a model. They show objects and the relationships between the objects.
Message	MSG		Messages hold text. The text can be used in error messages, as the caption for windows and dialogs, and so on.

Toolbar Icons

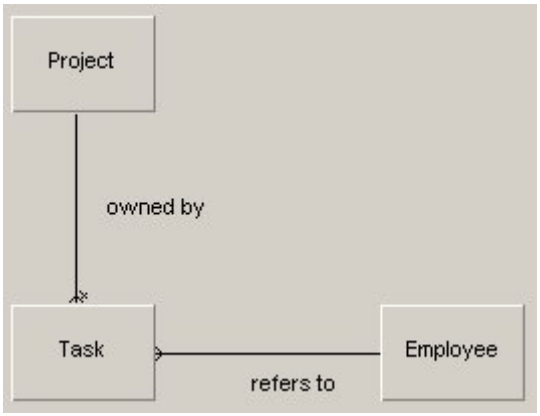
The following list shows a list of toolbar icons and describes each of them.

Icons	Name	Description
	Model Editor	Opens the Model editor.
	Show/Hide Browsers	Shows or hides the object browser.
	Fields	Displays field objects.
	Two Levels	Amount of detail displayed on Model Editor – Depth of two levels.
	One Level	Amount of detail displayed on Model Editor – Depth of one level.
	Clear Focus	Resets the display of the editor.
	Show/Hide Library Objects	Shows or Hides the library objects.
	Entities	Displays Entities.
	Inspect	Shows triples that define only that entity
	New Gen and Build	Opens a New Gen and Build Window.
	Gen and Build	Starts the Generate and Build process for the selected objects.
	WinC Build Summary	Displays the WinC build summary.
	Run	Runs the application.
	Copy	Creates a new project populated with a current project's data.
	Editor	Opens the appropriate editor based on the selection.
	Panel Palette	Displays the panel palette.
	Properties	Displays the properties.
	Check mark	Confirms the entered values in the Panel editor.

Project Management Application

In the following sections, you create a simple project management application. In this sample, a project consists of a group of tasks, and each task has one employee assigned to it. The model contains three entities: Project, Task, and Employee. To save time, the Employee entity is already defined for you.

Here is an entity-relationship diagram of the application created in this chapter. You will draw a diagram like this later.



If you have worked with entity-relationship diagrams before, you can observe the following:

- A project can have more than one task, but a task can only belong to one project.
- A task is owned by a project, which means that if you delete a project, you want all of its tasks deleted too.
- Employees are assigned to tasks—one employee can be assigned to more than one task, but each task can only have one employee assigned to it.
- An employee is not dependent on any particular task, which means that if you delete a task, you do not necessarily want to delete the employee record too.

You can see all of this without having to look at any code. This diagram shows useful information, which CA Plex uses to generate the application.

For the purpose of this sample application, this model is very basic. If you have developed real project management systems before, you may find parts of the model that you would design differently.

Open the Sample Model

First, you must start CA Plex and open the supplied sample model.

To open the sample model

1. Click Start, Programs, CA, CA Plex r6.1, and CA Plex.

   The CA Plex application opens.

2. Select File, Open.

   The Open File dialog appears.

3. Click My Documents (or Documents on Windows Vista) and navigate to the \CA\Plex\6.1 sub-folder. Then, click the Tutorial shortcut folder.

4. Select TUTORIALJAVA.mdl,and click Open.

   The Object Browser appears and the name of the model is displayed in the title bar of the main application window.

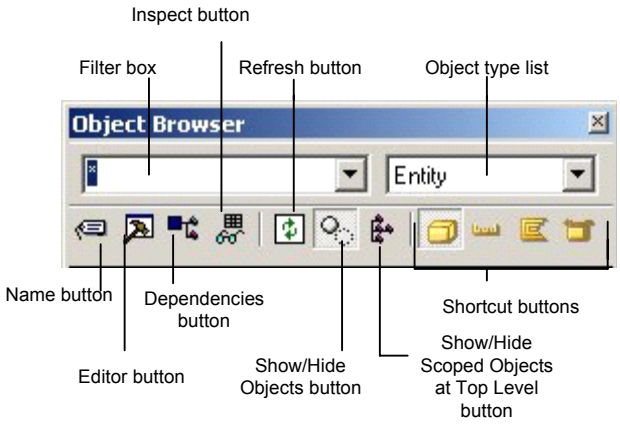
Note: Save your work! If you need to stop working on this tutorial before reaching the end of a chapter, you can save your progress by clicking the Save toolbar button. CA Plex also prompts you to save your changes when you close certain editors after making changes in them.

Object Browser

You can select to show or hide the *object browser* by clicking the Show/Hide Browsers toolbar button.

You can keep the Object Browser open while you work. The object browser lets you use it as a palette from which you can drag-and-drop the objects you need.

The following graphic shows the object browser as it appears in the application:



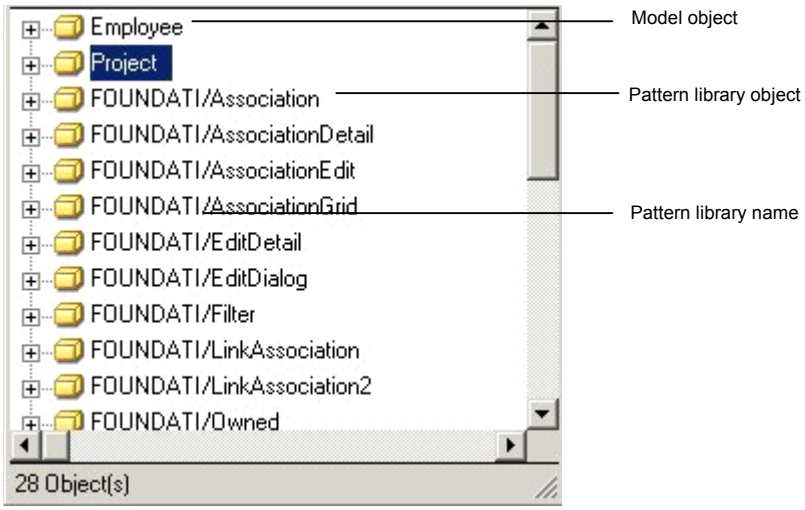
By default, the Object Browser shows you objects of one type at a time. You can see other objects if they are *scoped* by the main object type. For more information, see More About Scope section in this chapter.

The Object Browser has shortcut buttons for displaying entities, fields, and functions. Setting the Object Browser to display functions only displays unscoped functions; functions that are scoped by another object (such as an entity or a view) are displayed when the Object Browser is focused on that object type.

You can also select an object type to view from the object list.

You can click the Show/Hide Library Objects toolbar button to show library objects.

You can tell that an object is in a pattern library because the library name is prefixed to the object name by default. For instance, the third entity in this graphic is FOUNDATI/Association, which means that the entity Association comes from the FOUNDATION pattern library.



Project Entity and Triples

In CA Plex, you define objects and specify relationships between them with *triples*. As the name implies, a triple has three parts: a source object, a verb, and a target object.

For example, in this chapter, you use the following triple to define a unique identifier for the Project entity:

Project ***known by FLD*** Project ID

In this example, Project (an entity) is the source object of the triple, *known by* (or *known by FLD*) is the verb, and Project ID (a field) is the target object.

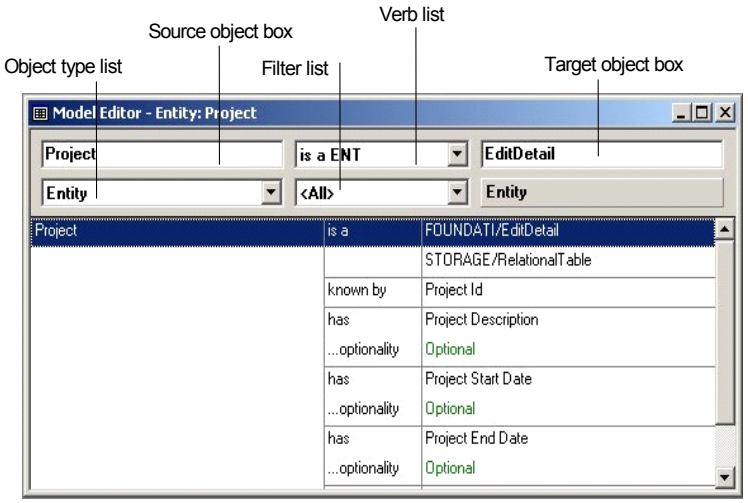
In CA Plex documentation, verbs are always printed in bold and usually with their source and target object types (ENT *known by* FLD, for example).

Triples provide a simple, easily understood language for entering information into CA Plex. Here are some more examples of triples that you will enter while creating your first application:

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

In this sample application, triples are entered to define fields for the Project entity: an identifier, a description, and start and end dates for a project.

You use the Model Editor to view, add, edit, and delete triples. The following is the Model Editor as it will appear when you complete this chapter.



Specify Attributes for the Project Entity

You must add triples to the model before you start working with it. In the next series of steps you add the following triples to the model:

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

To specify attributes for the Project entity

1. Click Model Editor from the Tools menu.

   The Model Editor appears. You can see some triples already entered in the Model Editor.

2. Select Entity from the Object type list.

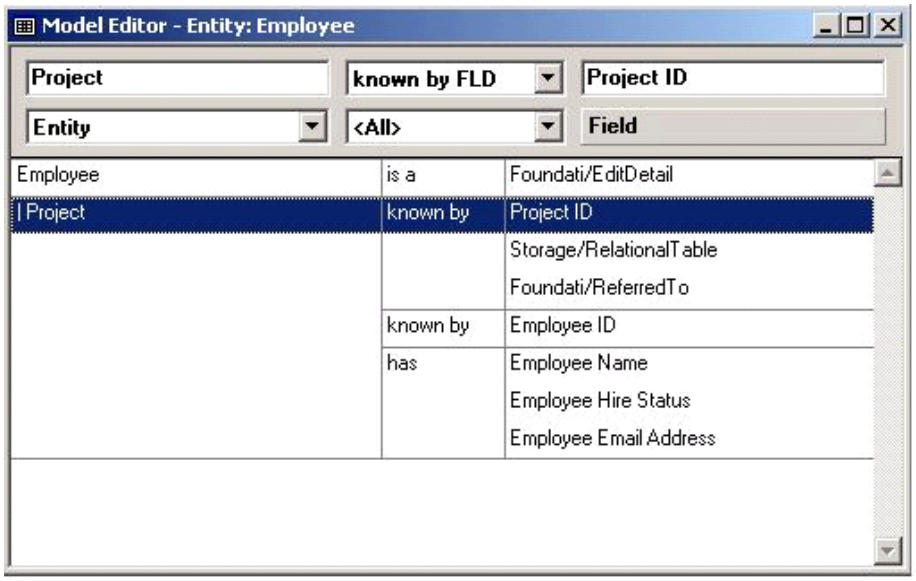
3. Enter **Project** in the source object box. Even though you have not specified anything about projects yet, you can type its name and CA Plex adds the entity to your model.

4. From the Verb list, select *known by* FLD.

   Note: Instead of scrolling in the list, you can click the verb list, and press the first letter of the verb you are looking for until the verb appears. To select the *known by* FLD verb, press the K key once.

5. Enter **Project ID** in the target object box, and press Enter.

   The Model Editor should now look like this:



   You just created the triple Project *known by* Project ID. This triple defines a primary key for the Project entity.

6. If the Object Browser is not open, click the Show/Hide Browsers toolbar button to open it.

7. Click the Fields toolbar button on the Object Browser to display field objects.

   Notice the Project ID field you just added. All of the objects you define in a local model appear at the top of the Object Browser. Pattern library objects appear below the local objects.

Notice that the Employee entity and its fields Employee Email Address, Employee Hire Status, and so on have already been defined in the model you are working with.

Project should still appear in the source object box in the Model Editor.

8. Select **has** FLD from the verb list.
9. Enter **Project Description** in the target object box.
10. Press Enter.

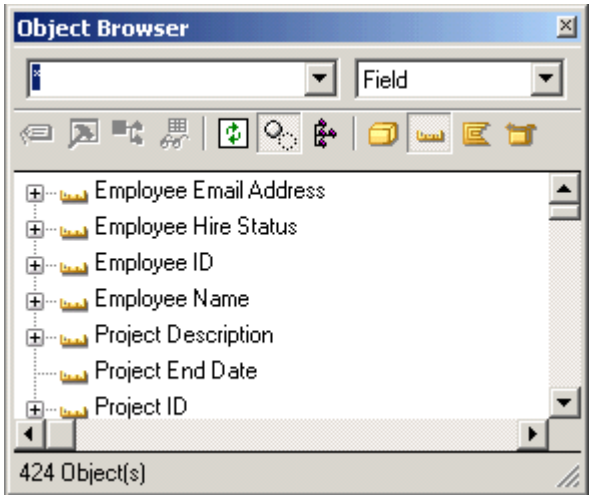
    You have created the triple Project *has* Project Description, which defines the field Project Description for the Project entity. You use this field to store a description of the project. This triple, *ENT has FLD*, creates a *non-key attribute*. The values in non-key attributes do not need to be unique to each entity. For example, you may have more than one project that has the same text in the description field.

11. Repeat Steps 8 to 10 to create the following triples:

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

12. Click the Refresh toolbar button on the Object Browser.

    The Object Browser shows the new fields:



Field Inheritance

You use inheritance to define the properties of fields. Inheritance in CA Plex is defined in a simple way—using the *is a* verb.

Following are the triples you enter in the next series of steps:

- 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

The FIELDS and DATE prefixes indicate that the objects concerned belong to the FIELDS and DATE pattern libraries, respectively.

The fields you defined previously have two different data types: character and date. These represent different kinds of fields. The Project ID field holds code that uniquely identifies a project, the description holds text, and the start and end dates hold dates. Currently, your model only indicates that those fields exist, and that they belong to the Project entity, but has no information about what type of data they store.

Inheritance is the mechanism that enables an object to adopt the properties of another more general (or abstract) object. By inheriting from pattern library fields, you enable your application to do the following:

- Validate data entered in the fields (which ensures, for example, that an end user does not accidentally enter February 31)
- Display data on the screen appropriately (such as displaying dates according to your Windows settings)
- Store data appropriately in the database (creating a text field in the database for the Project Description field, and date fields for the Project Start Date and Project End Date fields)

Use Inheritance to Define Field Properties

You can define properties to the fields in an entity. Properties provide information about the type of data each field stores.

To define the properties of Project's fields

1. Click the Show/Hide Library Objects button to display the library objects.
2. Click the Fields button to display field objects to ensure that the Object Browser is focused on fields.

3. Select the Project ID field in the Object Browser by clicking the name (not the icon to the left of the name) and drag the field from the Object Browser to the source object box of the Model Editor.

   This changes the Model Editor source object type to Field, and changes the verb list so that only verbs appropriate for fields are contained in it.

   Note: The cursor changes to a closed parcel icon when you drag an object. It changes to an open parcel icon when it is over a location where you can drop the object:

   The following table shows the closed and open parcel icons:

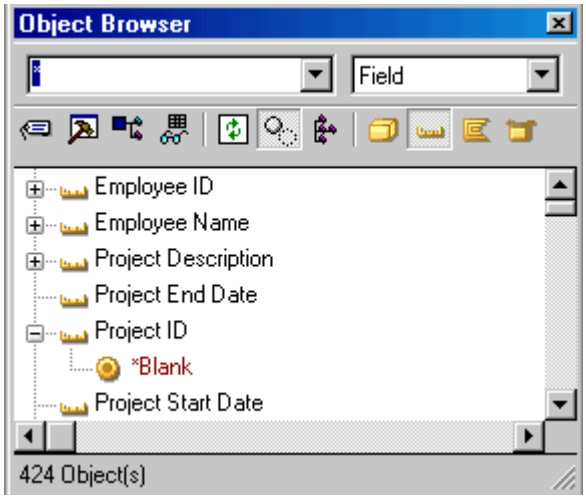
Closed Parcel Icon	Open Parcel Icon
	

4. From the verb list, select *is a* FLD.
5. From the Object Browser, drag the library object FIELDS/Identifier field to the target object box, and press Enter.

   Note: You have to scroll down the Object Browser to find the FIELDS/Identifier field. You can use the filter box at the top of the Object Browser to only show some of the library items. In this case, you could type ***Identifier*** to display only FIELDS/Identifier. Remember to set the filter back to * when you are done.

   You just created the triple Project ID **is a** FIELDS/Identifier.
6. Click the Refresh button on the Object Browser.

   The Project ID field has a plus sign (+) to the left, indicating that it now has scoped objects.
7. Click the plus sign (+) to expand the field:



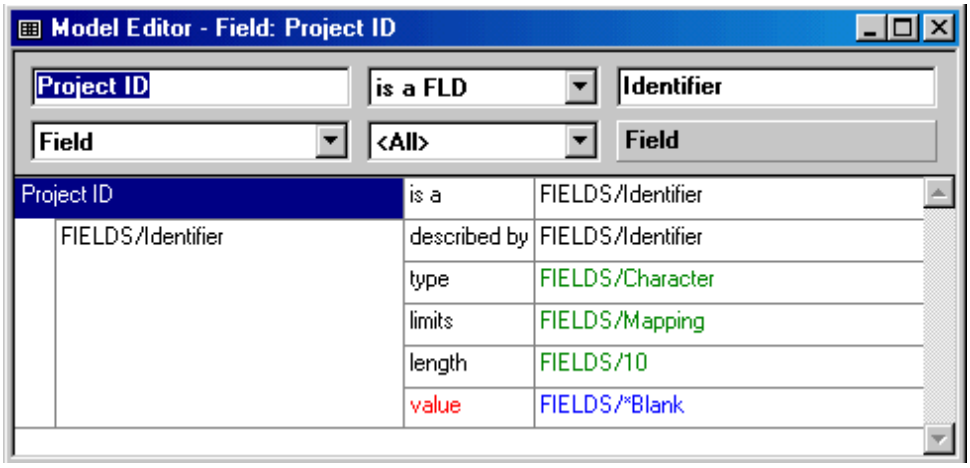
You can see that Project ID now has the value *Blank, but you cannot tell much else about what it inherited from FIELDS/Identifier. Values are another type of CA Plex object.

8. Drag the Project ID field from the Object Browser to the body of the Model Editor. The body is the bottom part of the editor, where the full triples are displayed.

   When you drag one or more objects to the body of the Model Editor, the display changes to show you only the triples that define those objects. This is called *focusing* the Model Editor. When you drag the Project ID field to the Model Editor, it focuses on this field, showing the triple Project ID *is a* FIELDS/Identifier. This still does not give you much information.

9. Click the Two Levels toolbar button to see more about what an object inherits from its ancestor objects.

   The Model Editor shows another level of detail.



Now you can see that Project ID has inherited a data type of character and length of 10, along with the value *Blank (which you saw in the Object Browser in Step 7).

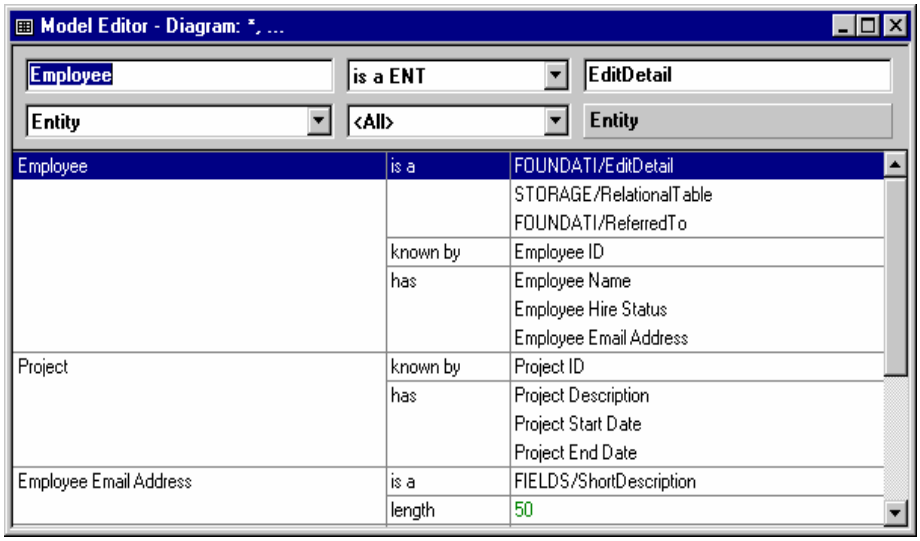
Note: In Step 5, you dragged the library object FIELDS/Identifier from the Object Browser to the target object box in the Model Editor. You can enter the name of the object into the target object box (without the library name) to accomplish the same thing. In Step 5, you would have entered Identifier.

Important! If you entered a wrong object name, you could create a new object with the wrong name. If this happens, find the erroneous object in the Object Browser and delete it, selecting the Ripple Delete check box on the Delete dialog.

10. Click the One Level toolbar button to set the Model Editor to show a single level of information.

11. Reset the Model Editor display by clicking the Clear Focus button.

    All of the triples appear in the model again.



    Note: If your Model Editor displays many more triples than shown in the previous graphic, then you have your model set to display library objects. If this is the case, click the Show/Hide Library Objects toolbar button.

12. Drag the Project Description field from the body of the Model Editor to the source object box. This field is in the third column of the Project *has* Project Description triple.

13. Enter **LongDescription** in the target object box, and press Enter.

    The Model Editor displays the triple as:

    Project Description *is a* FIELDS/LongDescription

    This indicates that you correctly spelled the name of the pattern library field.

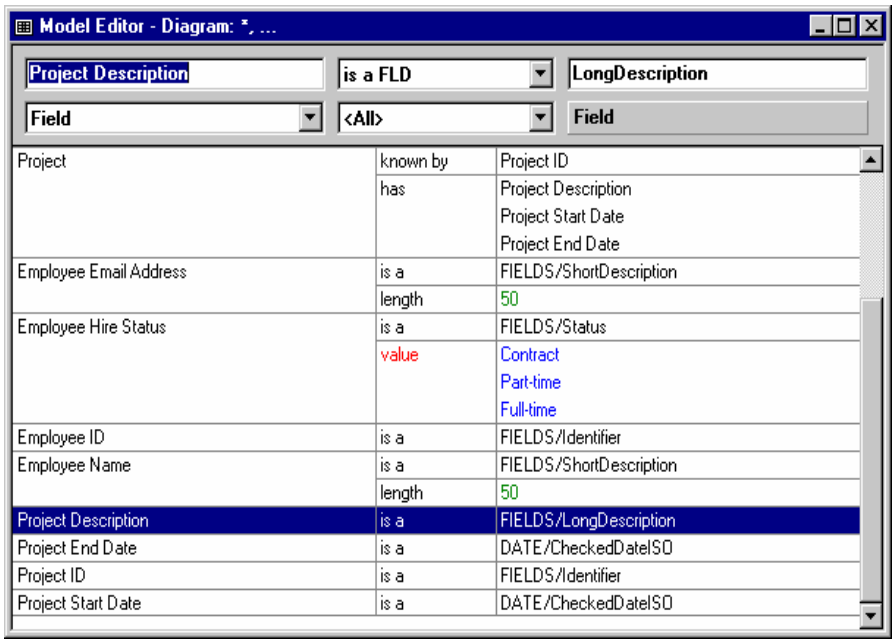
    Note: If you create a new object for a model, and it happens to share the name of a library object, you must rename your object or delete it (if you did not intend to create it).

14. Repeat either Steps 3 to 5 or Steps 12 to 13 to create the following triples:

    Project Start Date *is a* DATE/CheckedDateISO
    Project End Date *is a* DATE/CheckedDateISO

15. Click the Refresh toolbar button (on the main toolbar, not on the Object Browser).

    Your Model Editor should look like this:



    Inheriting from DATE/CheckedDateISO gives the fields functionality to ensure that end users enter valid dates.

16. Use the process explained in Steps 8 and 9 to look at the characteristics that these fields inherit from the pattern library fields.

Continuation Triples

We now introduce an extension to the triple syntax—the *continuation triple*. A continuation triple is like other triples except that the source object is a triple.

In the next series of steps, you enter the following continuation triples to specify that certain fields are optional:

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

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

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

You will enter other continuation triples later in this chapter.

Use Continuation Triples to Make Fields Optional

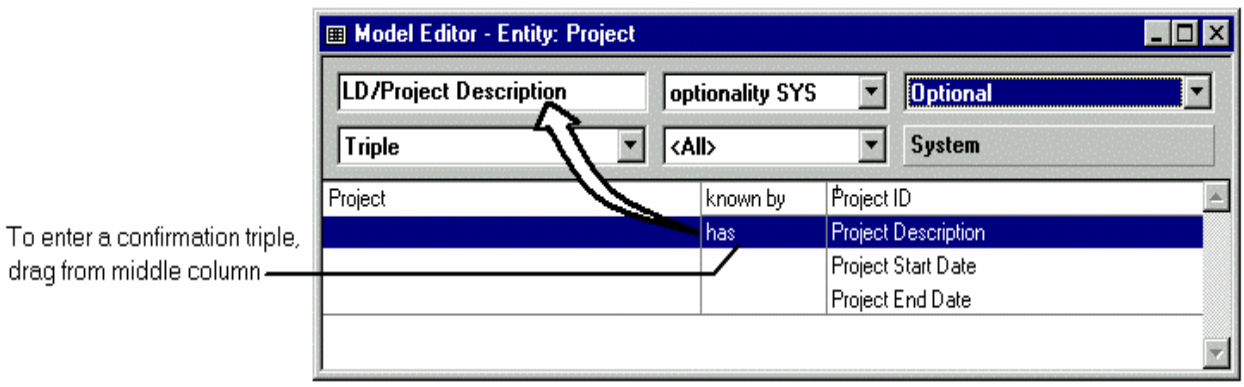
When your end users enter data, they typically enter data in every field. You can specify that some fields are mandatory, while others are optional. If you do not specify optionality, the default is that they are mandatory.

If end users leave a mandatory field blank and then try to close the dialogs, a message dialog prompts them to enter data for the blank mandatory field and does not let them close the dialog until they do. Since this processing is defined as part of the pattern library and not hard-coded into CA Plex, you can adapt it as required.

To use continuation triples to make fields optional

1. Click the Entities toolbar button to set the Object Browser to display entities.
2. Select the Project entity in the Object Browser and click the Inspect toolbar button.

   This focuses the Model Editor on the Project entity, showing only the triples that define that entity.
3. Click in the center of the triple Project has Project Description to select it.
4. Drag it to the source object (top left) box in the Model Editor.



5. Select optionality SYS from the verb (top middle) list, and Optional from the target object (top right) list.
6. Press Enter.

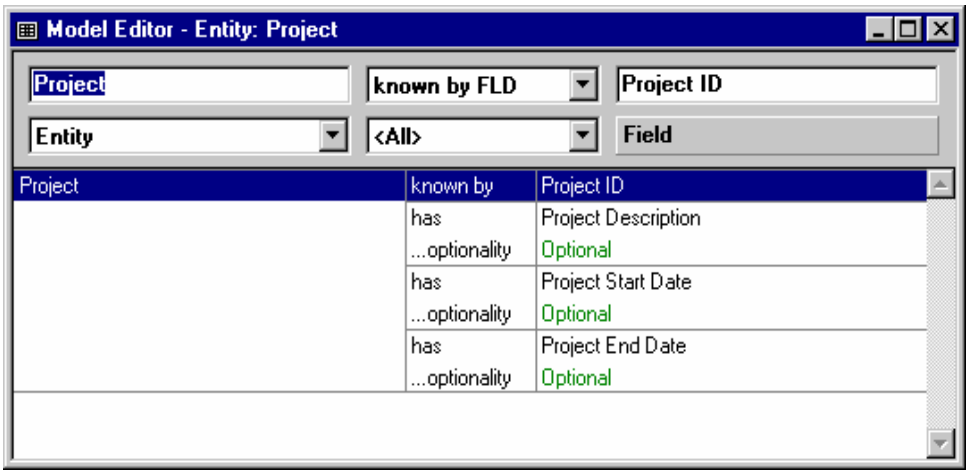
   You have entered the continuation triple:

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

   Note: If you do not see this view, click the One Level toolbar button to set the Model Editor to show a single level of information.

7. Make the Project Start Date and Project End Date fields optional in the same manner.
8. Refresh the Model Editor.

   The Model Editor will look like this:



Patterns and Inheritance

Inheritance is the primary means of reuse when you design an application with CA Plex. Inheriting from pre-built and pre-tested pattern libraries provides tremendous productivity and quality benefits.

Patterns are groups of CA Plex objects that are designed to be reused and customized. In this chapter, you inherit from the included pattern libraries. Additional pattern libraries are available from third-party vendors.

Patterns compared to Templates. The templates or frameworks supplied by other tools provide some of the benefits of patterns. Typically, a template provides a means of copying a predefined solution. Inheritance is not the same as copying. When an object inherits from an CA Plex pattern, the relationship is dynamic—changing a pattern automatically changes all the objects that inherit from the pattern.

Also note that patterns are created with CA Plex rather than being hard coded into the tool. You can customize a pattern library by creating your own patterns that inherit from the supplied patterns, or you can create your own patterns from scratch.

Patterns and Components. Patterns and components (such as COM and EJB components) are complementary technologies. CA Plex patterns can be used to combine and implement groups of components. Components are usually compared to the building blocks of a house. Extending this metaphor, you can think of patterns as the blueprint or plan for the house.

Like patterns, components are intended to maximize software reuse. Components cannot be changed, whereas patterns have an internal structure you can modify in certain places. Another distinction is that patterns are purely a design construct where components form a part of the running application.

Use Inheritance to Define the Project Entity

You have now defined the fields in which the Project entity stores data and specified the pattern library fields that those fields inherit from. In the next step, you give the Project entity a user interface and functionality to interact with a database.

To define the project entity, you again use inheritance. You define the project entity with only two triples:

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

The first inheritance triple gives your entity the ability to display and process a user interface. The second inheritance triple provides the functionality to read Project records from, and write Project records to, a relational database.

The step-by-step instructions that follow help you better understand the consequences of entering these two triples.

Add Functionality to the Project Entity

You can give the Project entity a user interface and functionality to interact with a database using inheritance.

To add functionality to the Project entity

1. Click the Entities toolbar button if the Object Browser is not displaying entities.

   Note: There is no plus sign (+) to the left of the Project entity. This tells you that there are no objects scoped to it.

   To add these triples, you must first set the source object type to Entity in the Model Editor. You could change the object type directly. But, when you drag an object from the Object Browser, it sets the object type *and* shows all of the triples for that object.

2. Drag Project from the Object Browser to the source object box of the Model Editor.

   This is similar to using the Inspect toolbar button; it changes the Model Editor so that it only shows triples related to the Project entity, changes the object type (assuming it was not already set to Entity), and puts Project in the source object box.

3. From the verb list, select **is a** ENT.

   Note: The **is a** verb that you use in this step is different from the one that you used to specify in inheritance for Project's fields. There are several verbs that have the same name, but have a different source and target object. CA Plex only lets you select the verb that matches the target object (in this step, the **is a** ENT verb). For more information about the types of **is a** verbs, search for *is a* in the online help index.

4. Enter EditDetail in the target object box, and press Enter.

   You just created the triple Project **is a** FOUNDATI/EditDetail. This indicates that Project inherits the structure and functionality of the EditDetail pattern in the FOUNDATION pattern library. You can find the FOUNDATION/EditDetail pattern in the Object Browser by making sure that library objects are displayed (by clicking the Show/Hide Library Objects toolbar button) and scrolling down.

   For more information about the EditDetail pattern, select it in the Object Browser, and press Shift+F1.

5. In the Object Browser, click the Refresh toolbar button.

   By inheriting from EditDetail, the Project entity now has some scoped objects.

   For more information about scoped objects, see More About Scope later in this chapter.

6. Click the plus sign (+) to the left of the Project entity and then click the Edit function to expand the Project entity and the Edit function.

   Project inherited one function, Edit, with a scoped panel and a caption, and two views, Fetch and Update. These objects give Project a user interface, and enable it to store data to and retrieve data from a database.

Specifically, CA Plex performs the following actions:

- The Edit function displays the panel scoped by it.
- The Caption and Panel objects, which are scoped by the Edit function, store the layout of the Edit Projects panel. To see what this panel looks like, see the generated application illustration in the section Run the Project.Edit Function later in this chapter.
- The Fetch and Update views scope functions that read and write database records.

Next, indicate how the Project entity stores information. Your application uses a relational database, so you need processing that creates and maintains database tables. Set Project to inherit from STORAGE/RelationalTable for that functionality.

The Model Editor should still have Project in the source object box, and **is a** ENT showing in the verb list.

7. Drag STORAGE/RelationalTable from the Object Browser to the target object box, and press Enter to create the following triple:

   Project **is a** STORAGE/RelationalTable

8. Click the Refresh button to see all the triples that you have defined for the Project entity in the Model Editor.
9. Click the Refresh button on the Object Browser.

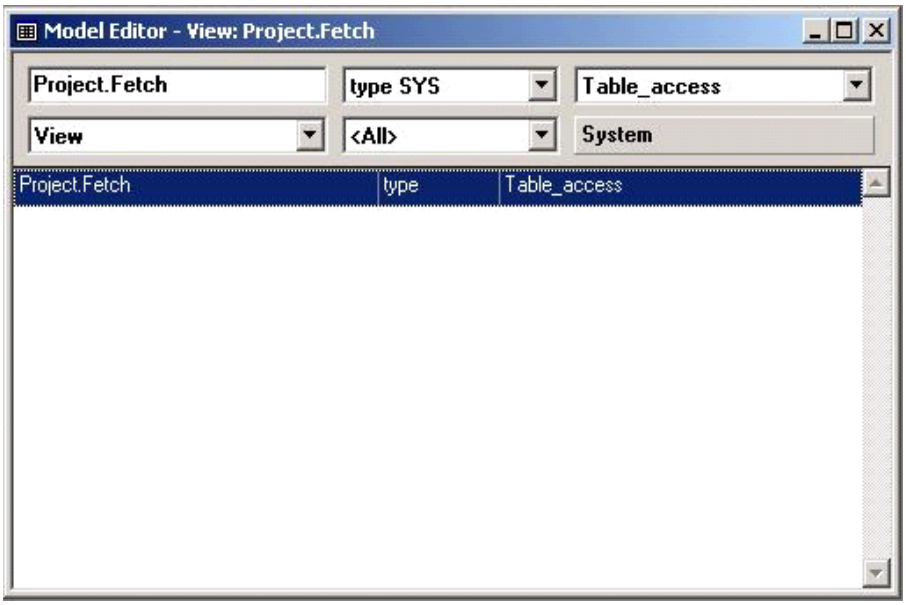
   The object browser will look like this:

Notice that the Project entity has inherited an object called Physical Table from STORAGE/RelationalTable. This defines the table that is created in the database when you generate the application. The fields that you defined for the Project entity are stored in this table.

Microsoft Office Access does not support views. However, the ODBC driver supports the SQL CREATE VIEW statement, and will create a query in Microsoft Office Access.

This means that generating and building a view in CA Plex creates a query in Microsoft Office Access. Microsoft's technical support described this situation as an undocumented feature. We do not recommend that you use it in CA Plex. Instead, set all views to have VW type SYS = Table Access, so that views are actually implemented.

10. Drag the view **Project.Fetch** from the Object Browser to the source object box of the Model Editor.
11. From the verb list, select **type SYS.**
12. Select **Table-access** in the target object box, and press Enter.



13. Create the same triple for the view Project.Update.

More About Scope

A *scoped object* is an object that belongs to another object. A scoped object cannot exist independently of the object by which it is scoped. For example, a panel that is scoped by a function is deleted when that function is deleted.

Some types of objects are unscoped. This means that they exist independently of all other objects in the model. For example, entities are typically unscoped. Conversely, a table is always scoped to an entity. To create a table object, you must specify the entity to which it belongs.

When you refer to a scoped object, you usually need to use its full name to avoid ambiguities. For example, there is an entity called Project that scopes a function called Edit. The function's full name is Project.Edit which distinguishes it from other Edit functions such as Employee.Edit that also exist in this model.

The previous steps demonstrated how scoped objects are usually created when you inherit from a pattern. You can also create scoped objects manually.

Set Up the Generate and Build Options for Java

You must set up the Generate and Build options for Java to generate and build a Java application using CA Plex.

To set up the generate and build options for Java

1. Select Generate and Build from the tools menu.
2. Select Generate and Build options from the Build menu.

   The Generate and Build Options dialog appears.
3. Select System Definitions, and double-click the Local System on the right.

   The System Properties dialog appears.
4. Select the Java Build option.

   The Java Build Options dialog appears.
5. Browse for the directory or type the path where you have installed Java Launcher (java.exe or javaw.exe; these are located in your jdk\bin directory).
6. Click OK.

   The Generate and Build Options dialog appears.
7. Click OK.

   The Generate and Build dialog closes.

Generate and Build

You have now defined fields for the Project entity and specified the properties of those fields. You have also defined functionality for the Project entity, providing a basic user interface and the ability to write to and read from a database. You are now ready to generate and build the Project entity. This process turns your model into source code (generating) and then turns your source code into compiled objects (building).

After you have generated and built the objects in your model, you can run the program to see what you have created.

To generate and build the Project entity

1. Click Generate and Build from the tools menu, or click the New Gen and Build toolbar button.

   The Generate and Build window appears. The Message Log pops up when you open the window.

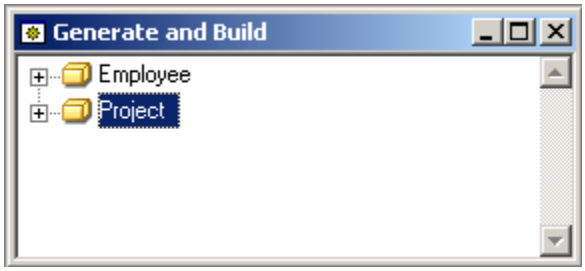
   Note: To generate and build a Java application, you must set up the Generate and Build options for Java.

2. From the Options menu, click Quiet Mode, and minimize the window.

   This keeps the Message Log from popping up every time it has a new message.

3. If the Generate and Build window shows library objects, click the Show/Hide Library Objects toolbar button to hide them.

   The Generate and Build window now shows the Project and Employee entities (Employee was already added to your model before you started).



4. Select the Project entity.
5. Click the Generate and Build toolbar button.

   CA Plex expands the Project and highlights all of the objects that are generated within the Project. Not all of the scoped objects are selected.

   A Confirm Generate dialog appears indicating the number of objects that are generated.

6. Click Yes.

   CA Plex generates those objects and then summarizes the generation process.

   Note: Three warnings appear in the Message Log during the generation and build processes. This is expected and is not a problem. You can expect *one* warning message (BLD9083) to occur during the generation stage.

   For more information, see Message Log in this chapter.

7. When the generation process is complete, the Cancel button becomes an OK button. Click OK to close the Generation Status dialog.

   CA Plex prompts you to compile and build the objects.

8. Click Yes both times.

   The first Java compile dialog is to compile the database tables and views. The second Java compile dialog is for the coalition of the java functions.

   CA Plex opens a new Shell Build window and uses Apache ANT and the Java compiler to build your generated code.

   At the same time, the database table is sent to the ODBC data source being used for this sample application. To make the setup easier, CA Plex created this data source automatically during installation together with the underlying Microsoft Jet database.

   You can see that your Java build is complete when the message *BUILD SUCCESSFUL* appears in the Shell Build window. You can close the Shell Build window.

Native Platform Implementation

CA Plex provides a complete environment for generating and compiling across all supported platforms. For Java implementations, you can see how it seamlessly integrates with Apache ANT.

CA Plex generates Java code that is based on the Java SE platform. Each function in the model is implemented as a set of Java classes that are then packaged into one or more JAR files.

If we chose to implement our application in C#, CA Plex would generate C# code based on Microsoft's .NET Framework. Similarly, we could also implement the application on the IBM System i by generating native RPG and DDS code.

Message Log

Three warnings appear in the Message Log during the generation and build processes. This is expected and is not a problem. If you have additional warnings or error messages this may indicate a problem. The most likely cause is that you have not entered all the necessary triples into the model—go back and check against the instructions.

Errors are also likely if CA Plex did not install properly.

- One warning message (BLD9083) indicates that the Project Description field has a length greater than 255. This is not a problem for the database we are using.
- Two warning messages (BLD9081) indicate that no source for various views has been submitted to the ODBC data source. This is expected because we have set up CA Plex not to implement views in the Microsoft Office Access database being used for this sample application.

Check for Java Build Errors

To check for Java build errors

1. Click the Java Build Summary toolbar button to view the build summary, to make sure there were not any errors.

   You are not prompted to review the build summary any more in this sample application, but if your application does not run correctly, you should review it to see if there were errors.

2. Click the Save toolbar button to save your model.

Your Generated Application

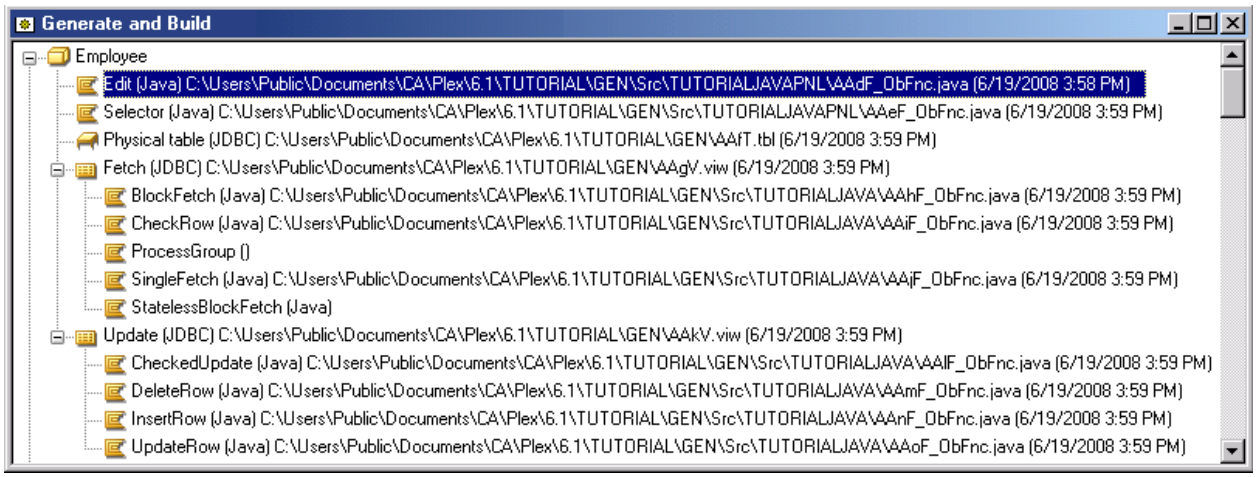
By entering a few triples, you have created a fully functional application—an instance of the EditDetail pattern. You can use your generated application to create, edit, and delete projects. You can create a description for each project and indicate a start and end date.

Run the Project.Edit Function

You have now created a fully functional application. You can run this application to create, edit, and delete projects.

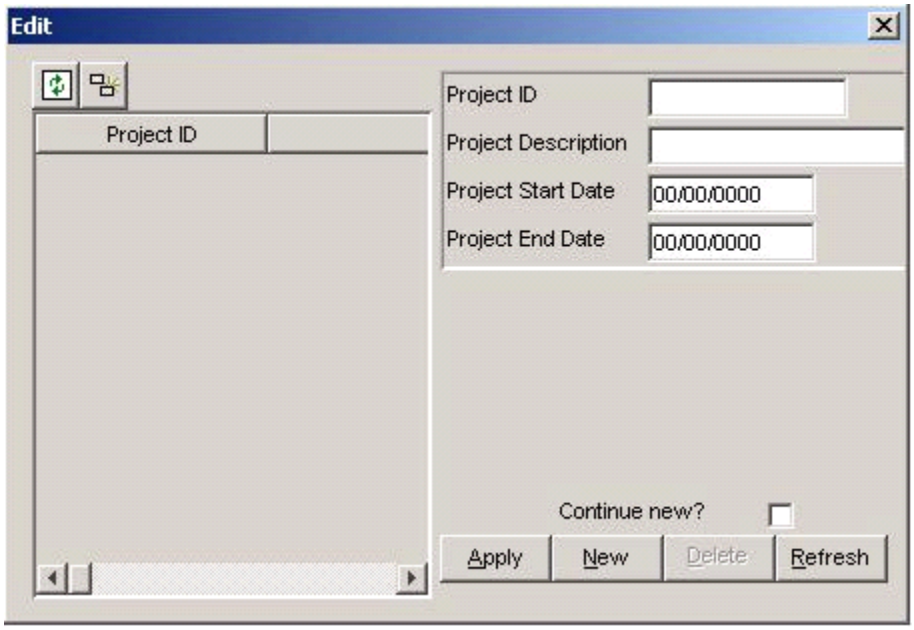
To run the Project.Edit function

1. Select the Edit function under the Project entity in the Generate and Build window.



2. Click the Run toolbar button.

   Your generated application starts. It should look like this:



   Leave this dialog open. The next steps show you how to add projects to your database.

Default Panel Layouts

Based on the design details you entered earlier, CA Plex has automatically added the correct fields to the panel and provided a default layout. You adjust the default layout later. The general appearance and functionality of this panel, with the grid on the left and the fields on the right, is determined by the pattern library from which it is inherited. You can override almost any aspect of this appearance and behavior.

Add a New Project

You will use your created application to add a new project.

To add a new project

1. Enter the following values in the Edit dialog shown previously.

   Note: The format required for the dates vary depending on your regional settings in the Windows Control Panel.

Value	Enter
Project ID	**Proj01**
Project Description	**Temporary description.**
Project Start Date	**12/10/2006**
Project End Date	**01/01/2007**

   For now, you only enter **Temporary description.** in the Project Description field because the field extends past the right edge of the dialog. If you type anything longer, you will not be able to see what you are typing. Since Project Description was defined as an optional attribute, you can leave it blank if you prefer.

   After you finish entering data, modify the panel layout to view the entire description.

2. Click the Apply button.

   The values are posted to the database and the grid region on the left of the window shows the new project.

3. Click the Copy toolbar button to create a new project populated with Proj01's data. Add the following two projects to the database:

Value	Enter
Project ID	**Proj02**
Project Description	**Temporary description.**
Project Start Date	**03/01/2006**
Project End Date	**04/04/2006**

Value	Enter
Project ID	**Proj03**
Project Description	**Temporary description.**
Project Start Date	**05/05/2006**
Project End Date	**06/05/2006**

4. Click the close window button at the upper-right corner of the dialog to exit the application.

   Note: None of the panels that are inherited from the pattern libraries have Cancel buttons on them. If you click the Close Window button, any pending actions are discarded. If you click Apply, though, those changes are completed before the window closes.

100 Percent Code Generation

You have entered 13 triples into the model. From those triples, CA Plex generated over 8,000 lines of Java and SQL code to create a fully functional application. You got one hundred percent code generation!

Feel free to experiment. Examine referential integrity checking inherited from the pattern library. For example, you cannot enter the same Project ID for multiple records.

Preserve Data

By default, each time you build your application, CA Plex rebuilds all of the objects you select in the Generate and Build window, including the tables in your database. Because rebuilding a database table erases all data in the table, if you leave your local model setup as it is, you will lose all of the data you just entered the next time you build.

You can prevent losing this data by entering a TBL **implement** SYS No triple for the table. This keeps the table from being rebuilt the next time you build the entity to which it is scoped (in this case, Project). Only set this triple after you have built the table at least once.

To preserve the data and keep the Project entity table from being regenerated

1. In the Object Browser, select Project.Physical Table, and drag it to the source object box in the Model Editor.
2. From the verb list, select **implement** SYS.
3. From the target object list, select the value No.

4. Press Enter.
5. Save your model.

Note: If you make any changes to an entity that affect its table, such as adding fields to it, you must set the **implement** SYS value back to Yes and regenerate the table. Any data in it is lost. If you want to preserve data entered after rebuilding, make sure you reset the **implement** SYS value to No.

Panel Designer

You may have noticed that when you ran the Project.Edit function, the dialog did not look quite right. The Project Description field ran off the right edge, and there was not enough room to enter more than a couple of words of description.

The Panel Designer changes the panel displayed by Project.Edit. You can change the size of the Project Description field and give it a multiline edit control (so that you can enter multiple lines of text). You can add some spin controls to the date fields too.

Visual Development

You have seen that, unlike visual development tools, the starting point for an CA Plex application design is a design model, not screen layouts. This is because design and modeling are the keys to building large-scale applications successfully. However, just like a visual programming tool, CA Plex provides an easy-to-use editor for designing graphical user interfaces (GUIs). It includes a rich set of native GUI controls plus the ability to use third-party components (ActiveX controls in Windows C++ and JavaBeans in Java). There is another mode of editor that enables the character-based screens of a System i 5250 application to be designed in the same way.

Open the Panel Designer

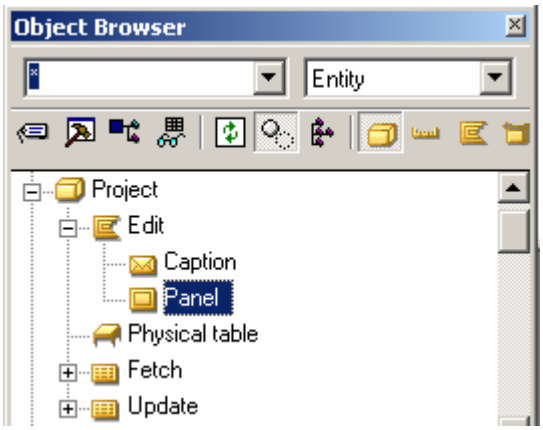
The Panel Designer lets you change the visual appearance of a panel. It comprises three separate windows: Design window, Panel palette, and property sheet.

To open the Panel Designer

1. If the Object Browser is not displaying the entities, click the Entities toolbar button.

   Note: This button is not on the toolbar. It is on the Object Browser. Some buttons are on both the Object Browser and the toolbar.

2. If you do not see the Project entity, click the Refresh toolbar button.
3. Expand the Project entity, expand the Edit function, and select Panel as shown in the following dialog.



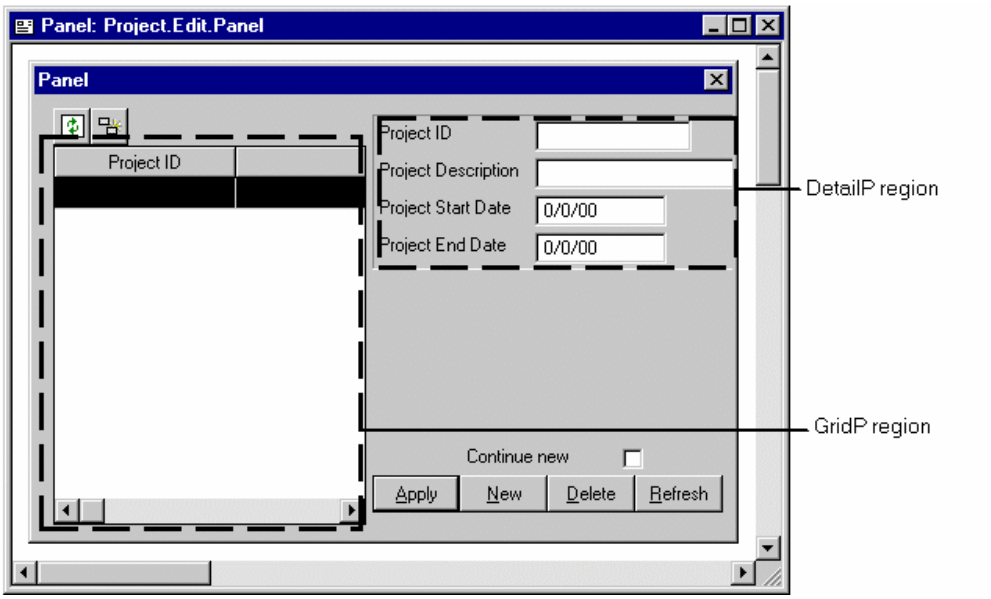
4. Click the Editor toolbar button.

   The Panel Designer appears. You can click the Editor button when you select any object. CA Plex always opens the appropriate editor.

Design Window

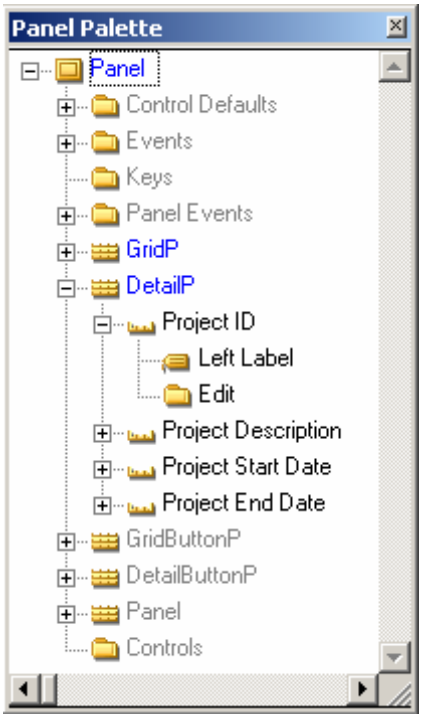
The Design Window is the main window in the Panel Designer and shows you what your panel looks like. When you change a panel's properties, you see how the changes affect what the panel looks like here. You can select, move, and resize buttons, fields, and other user interface elements using this window. When you make visual changes to the panel and its elements using the other windows, the changes appear here.

The elements of a panel are grouped into regions (each region has a name). The following graphic shows the grid and detail regions in the panel displayed by Project.Edit:



Panel Palette

You can open the Panel Palette by selecting Panel Palette from the Windows menu. The Panel Palette shows all the elements of the panel, including fields, labels, and buttons which are grouped under folders ( ) and regions ( ). You can expand the folders and regions to see what they contain. Most of the visible elements are contained in regions. You can see the five regions of the panel (GridP, DetailP, GridButtonP, DetailButtonP, and Panel) in the Panel Palette.



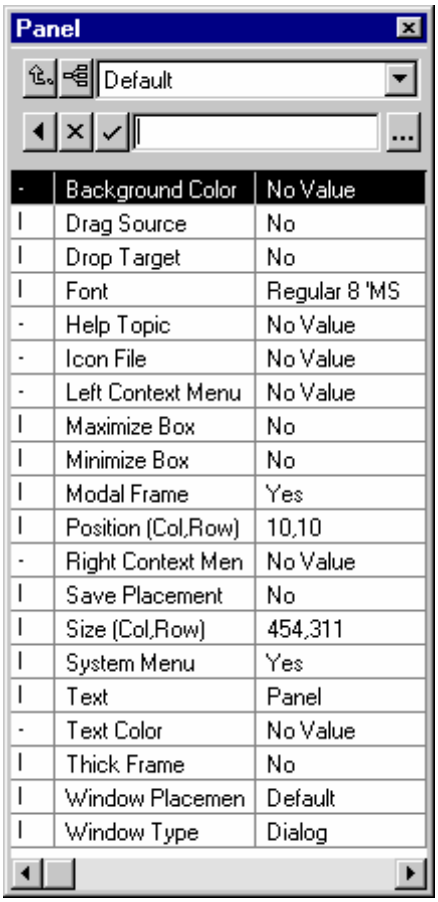
When a region is expanded, as the DetailP region is in this example, you can see the elements contained in that region. In the case of the DetailP region, this includes the Project ID, Project Description, Project Start Date, and Project End Date fields.

When represented on panels, fields typically contain more than one part, usually including at least one label and a control. Notice on the previous graphic that the Project ID field shows a Left Label and an Edit control. The label indicates whether it is a Left Label, a Top Label, or a Right Label. The type of label indicates if it appears to the left or right of the control (left/right label), or as a column heading (top label). The control is the part that the end users interact with. The settings for the control indicate if it is displayed as an edit box, a list box, a combo box, and so on.

Property Sheet

Every element on a panel has a set of properties associated with it, such as its size, color, or position. The Property Sheet shows you the properties of the currently selected element. If no element is selected, the Property Sheet shows you the properties of the panel as a whole.

To see what you can change about an element, select the element in the Design Window or the Panel Palette and check its properties on the Property Sheet. The properties displayed depend on the type of element selected.



For example, you can specify if a button should be included in the tab sequence by setting its Tab Stop property. However, a field label would never be included in a tab sequence, so if you select a label the Property Sheet does not display this property.

Define a Multiline Edit Control

You can modify the panel displayed by Project.Edit so that it displays multiple lines of text and does not extend past the edge of the dialog.

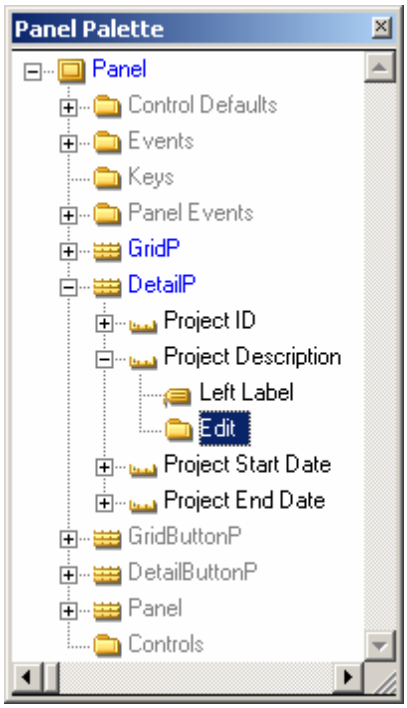
To modify Project.Edit.Panel and to define a multiline edit control

1. Arrange the Panel Designer windows so they do not overlap and you can work with the contents of each.

   You must select an element on a panel to make a change to it. To select only the element, and not the region it is in, use the Panel Palette.

2. Expand Panel to display the regions on the Edit panel if it is not already expanded.

3. Expand the DetailP region and expand the Project Description field.



4. Select the Edit control.

   The values in the Property Sheet indicate the properties you can change for the edit box. Also, notice in the Design Window that only the edit box next to the label Project Description is selected.

5. In the Design Window, drag the edit control below the label so that its left edge lines up more or less with the *j* in the word *Project*. Do not be concerned that it covers up the Project Start Date field—this is fixed later.

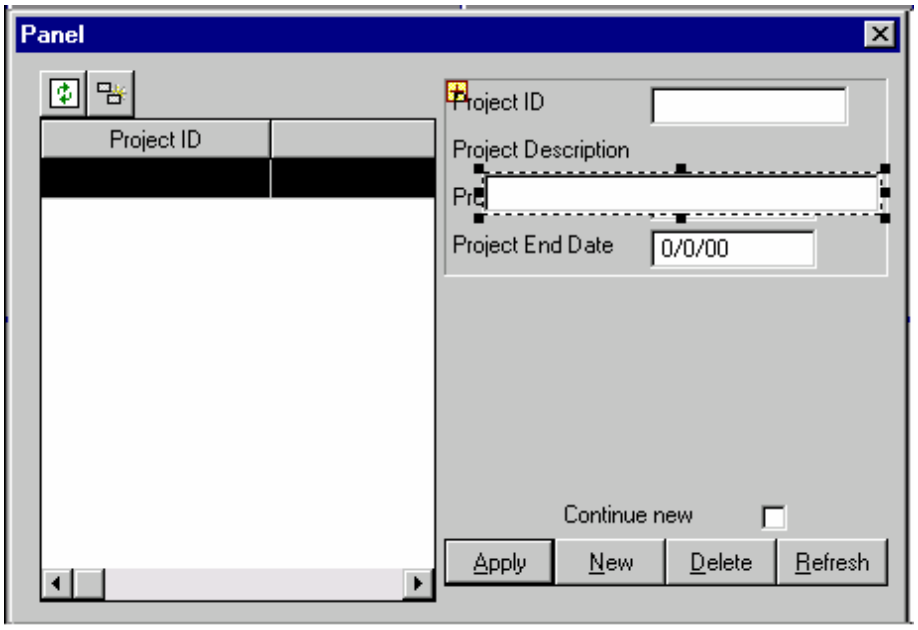
   In the Property Sheet, the Position (Col, Row) property will change as a result of moving the element. If you want to make fine tuning changes, you can specify the exact pixel location in the Property Sheet.

   The field still extends past the right edge of the panel.

6. In the Property Sheet, click the Size (Col, Row) property.

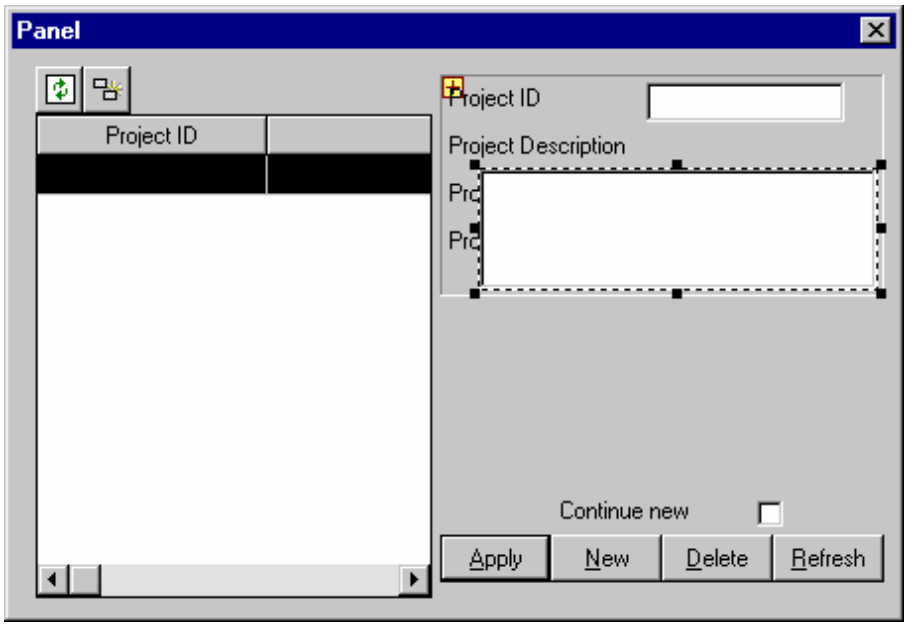
7. In the edit area at the top of the Property Sheet, change the value to 200, 20, and click the check mark button.

   The edit control resizes so that it fits within the panel and the beveled box that surrounds the fields resizes so that it is just slightly wider than the fields.



8. Drag the handle on the bottom center of the edit box to make the box taller. Make it about three times as tall.

   The panel should look something like this:



   You have repositioned and resized the Project Description edit box. Next, set two properties that enable end users to enter more than one line of text in the edit box.

9. In the Property Sheet, double-click the Multi-line property to change its value from No to Yes.

10. Select the Scrolling property in the Property Sheet.

    Note: If the Scrolling property is not available, ensure that the value of the Multi-line property changed from No to Yes.

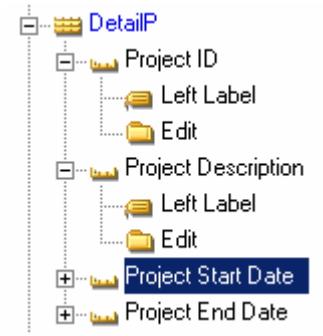
    Notice that the input area at the top of the Property Sheet is changed so you can select a value from a list, but you cannot type in a value.

11. Select Vertical from the list.

    Setting the Scrolling property to Vertical causes the text in the multi-line edit box to wrap to the next line when the cursor reaches the right edge. If you do not set this option, text continues on the first line and extends beyond the edge of the field, rather than wrapping to the next line.

    When you change the size and shape of the Project Description edit box, it covers the Project Start Date and the Project End date fields. Next, move those two fields down.

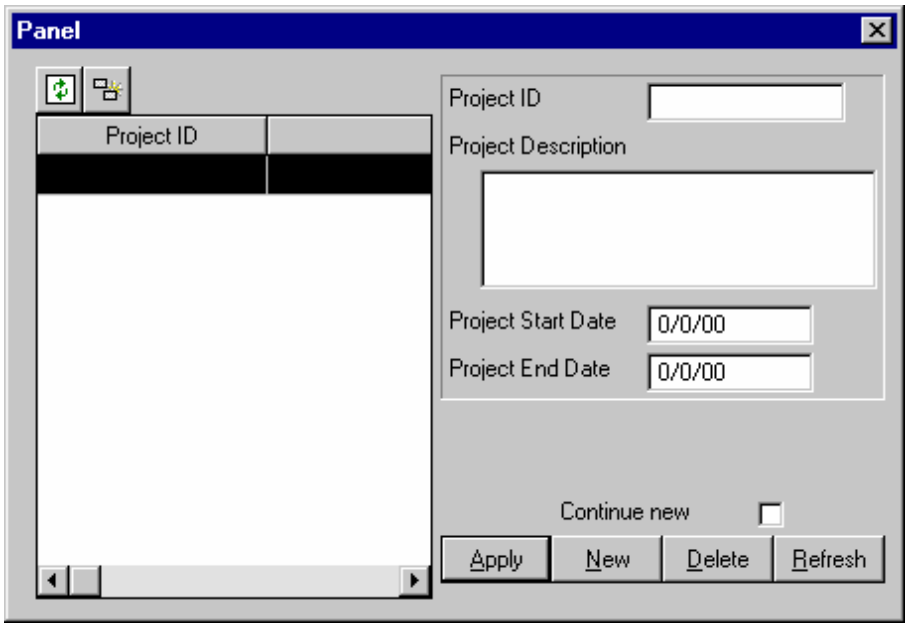
12. In the Panel Palette, select the Project Start Date field in the DetailP region without expanding it.



13. Holding the Ctrl key down, select the Project End Date field. This selects the Project Start Date and the Project End date fields.

14. Drag the fields in the Design Window so they are below the bottom of the Project Description multiline edit box.

    You can see that the beveled box around the fields resizes to fit around all of the fields:

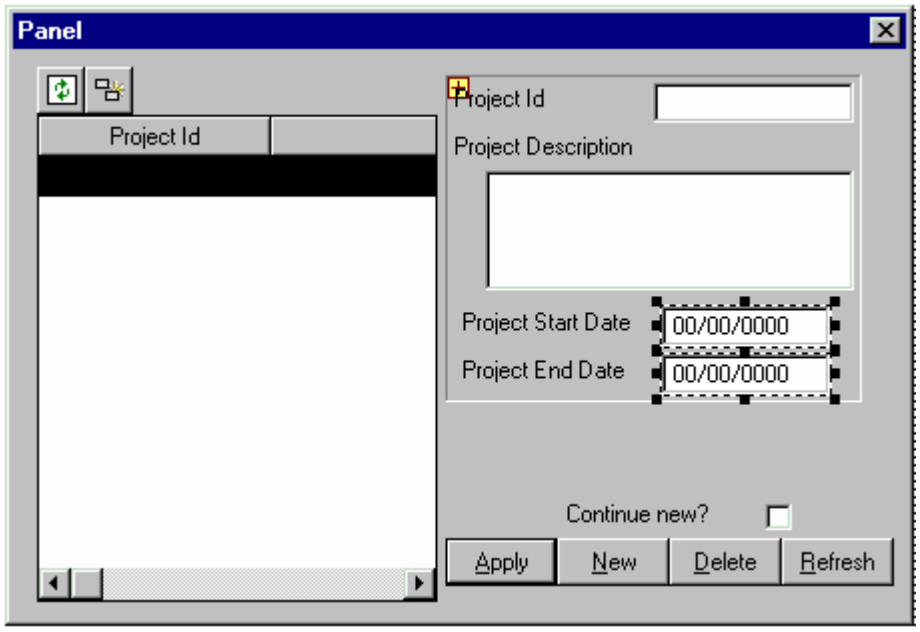


Add Spin Controls

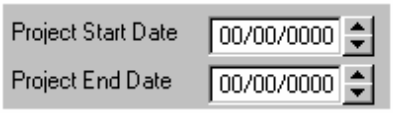
The spin control lets the end user increase or decrease the value in the fields by clicking the arrow buttons instead of typing a value in the edit box.

To modify the two date fields so that they include spin controls

1. On the Design window, hold down Ctrl and click one of the edit controls of the date fields. Holding down Ctrl causes the control to be selected instead of the region in which it is contained.
2. Hold down Ctrl *and* Shift at the same time and click the other date control. Both date controls should now be selected and the Design window should look like this:



3. In the Property Sheet, locate the Spin Control property and set it to Yes.



   Notice how you can select multiple elements on a panel and change properties for all of them.

4. From the File menu, click Save.
5. Close the Design Window.

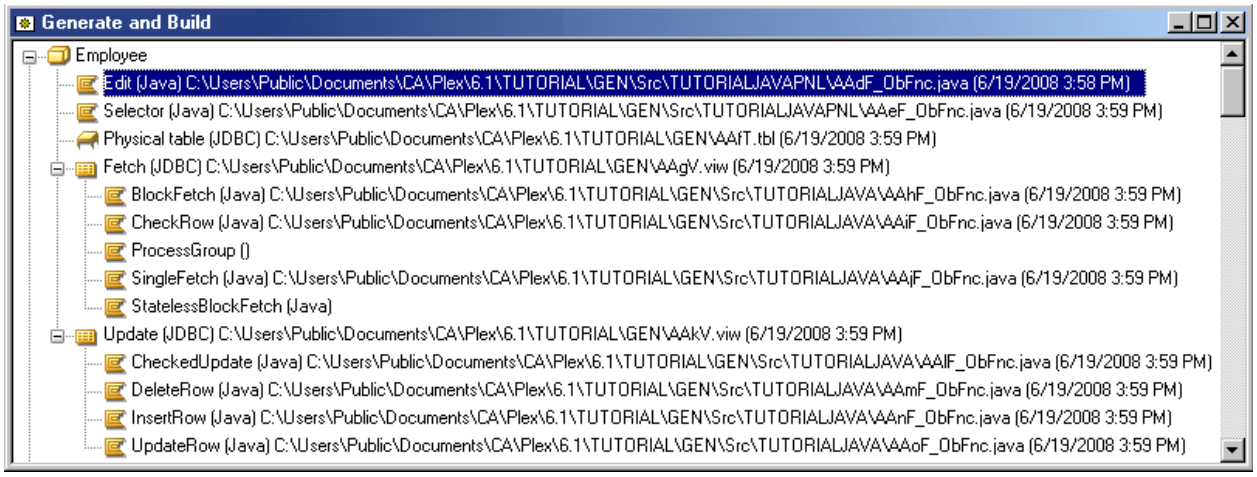
   This closes the whole Panel Designer.

Regenerate the Function

Now that you have modified the panel, you must regenerate and rebuild the function that displays it before you can see the changes.

To regenerate and rebuild the Project.Edit

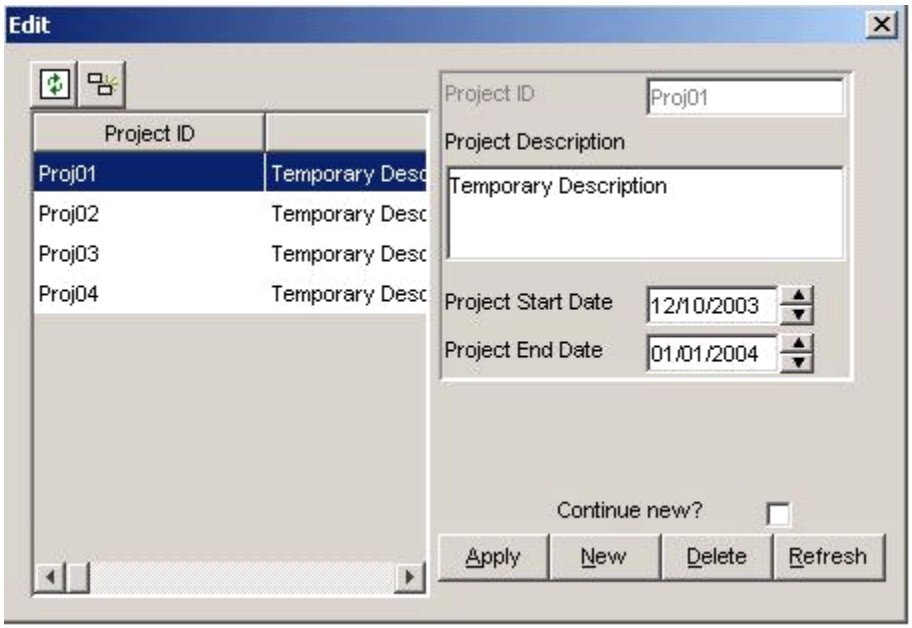
1. If you still have the Generate and Build window open, switch to it. Otherwise, click the New Gen & Build window toolbar button.
2. Select Project.Edit.



To rebuild a panel, you regenerate and rebuild the function that scopes it. Because you have only modified the Project.Edit.Panel object, you only need to regenerate and rebuild Project.Edit. If you had made changes that affected the scoped table or other objects, you would also need to regenerate and rebuild those other objects.

3. Click the Generate and Build toolbar button.
4. Click Yes when prompted to generate and click Yes when prompted to build.
5. When the Generate and Build process is complete, click Project.Edit in the Generate and Build window, and click the Run toolbar button.

The modified dialog appears showing the data you already entered.



6. Select the text in the Project Description field for Proj01 and replace it with a longer description. For example:

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

7. Click Apply.

   Notice that the description is changed in the grid and detail regions.

8. Select Proj02 and Proj03 and repeat this process to change their description to:

   Proj02: Create an email client for internal support staff.

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

9. Close the application.

Review

In this chapter, you have done the following:

- Learned how to use the Model Editor to add triples
- Inherited from pattern library objects to define fields, and data access and user interface functions
- Generated and built the Project entity's database table and the inherited functions, and added some sample data
- Used the Panel Designer to modify the inherited panel layout
- Regenerated and rebuilt the affected objects to see your changes

This chapter introduced the following patterns (and their scoped objects):

	Pattern	Description
	FIELDS/Identifier	A 10-character data field.
	FIELDS/LongDescription	A variable-length character field.
	DATE/CheckedDateISO	A date field, where the date is stored in the ISO standard for dates. Verifies that dates entered are valid.
	FOUNDATION/EditDetail	Lists database records and lets you edit, add, change, and delete records in a single dialog.

This chapter introduced the following triples:

Triple	Description
ENT *known by* FLD	Defines the primary key of an entity
ENT *has* FLD *...optionality* SYS	Defines a data attribute of an entity and whether it is mandatory or optional to provide a value for a field
ENT *is a* ENT	The source entity inherits the properties of the target entity
FLD *is a* FLD	The source field inherits the properties of the target field
TBL *implement* SYS	Specifies whether to generate and build a database table

Chapter 4: Group Model Licensing

Group model licensing is an optional feature that enables Independent Software Vendors (ISVs) to protect their group models from unauthorized use. CA and its business partners must have informed you which of their group models are licensed during the sales transaction.

The first time you reference or open a licensed group model, the licensing dialog appears identifying the group model as licensed. The licensed group models require an authorization code.

If the licensed group model author enables the trial license feature, you will receive a trial license for a temporary period. You must request and enter your authorization code before this period ends. If enabled, the trial license feature lets you immediately use the group model while you wait for your authorization code. If this feature is not enabled, you cannot use the model until you receive your authorization code. After you receive your authorization code, enter it in the Licensing dialog.

You can get your authorization code using the email or fax facility in the CA Plex Licensing dialog.

The information about whom you should get your Authorization Code from is stored in the licensed group model. If you use the email or fax facility, this information automatically appears. You can also select About Group Model from the Help menu to view this information.

For more information about navigating to a licensed group model's licensing dialog, see Access the CA Plex Licensing dialog. Use these instructions, where applicable, to maintain the license of a licensed group model.

Local Model Licensing

In addition to the group model license, the vendor can opt to secure any local models that use a licensed group model. This means that a license check is performed each time a local model is opened that uses objects from the licensed group model. You must have a valid authorization code on your workstation to use such local models.

Access the Group Model Licensing Dialog

Use the Group Model Licensing dialog to enter your authorization code for a licensed group model.

To access the Group Model Licensing dialog

1. Start CA Plex.
2. Log on to your host group model.
3. Select the licensed group model, and then select About Group Model from the Help menu in the Group Model window.
4. Click License.

If a licensed group model is unauthorized, the dialog automatically appears the first time the model is opened.

Group Model License Transfer

Group model licenses are highly secure and specific to a specific PC. You cannot use the same authorization code on more than one PC. Instead, you can transfer a group model license to another workstation. You can either transfer the license directly across the network or indirectly using a floppy disk.

Make a Direct Transfer

Use this method to transfer a license to an unlicensed version of the group model that exists on a different PC on your network.

To perform a direct transfer

1. On the target PC, attempt to access the target copy of the group model (the unlicensed version) at least once, thus creating the license directory. Make sure that you close it before performing the license transfer.

   Important! Never transfer a license to an open copy of CA Plex; it disables the license.

2. On the target PC, share the license directory of the target copy of CA Plex (Windows\Oblicense) with full permissions, so the license files can be copied.

3. Map the directory path for the target copy of the CA Plex license directory.

   Note: UNC path names are not supported; for example, \\MyServer\Windows\Oblicense. You must map to the target PC using a drive letter, for example, K:\Windows\Oblicense.

4. Log on to your host group model on the source workstation.
5. On the Group Model window, select the licensed group model and select About Group Model from the Help menu.
6. Click License, Direct License Transfer.
7. In the Direct License Transfer dialog, enter the path to the target PC's Windows\Oblicense directory (remember, the path must begin with a drive letter).

   Note: This is either the root OBLICENSE directory for CrypKey 4.2, or the OBLICENSE\61 directory for CrypKey 6.1.

8. Click OK to transfer the license.

   A message box displays the results of the transfer.

How to Make a Floppy Disk License Transfer

Use the floppy disk method to transfer the license between two PCs that cannot communicate over the network.

Note: When transferring the license to a different PC on your network, we recommend using the direct transfer method.

Using this method requires advance planning. Before you begin the transfer process, the trial license on the target PC (unlicensed version) must be expired. This is because the target PC needs to be registered before you transfer the license. You cannot register the PC until the trial license has expired.

For this transfer, you need an ordinary floppy disk with 180 bytes of free space.

The following scenario explains the procedure of transferring a license from your office PC to your home PC, and back again to the office PC.

Prepare for the Transfer

Before you can transfer your license, you must prepare your home PC and office PC for the transfer.

To prepare for the transfer

1. Install CA Plex on your office PC and enter the authorization code for the group model.
2. Install CA Plex and the group model on your home PC and attempt to access the group model. Wait until the trial license expires before continuing.

Transfer the License

Use the following procedure to transfer a group model license from your office PC to your home PC.

To transfer a group model license to your home PC

On your home PC:

1. Start CA Plex and attempt to access your target group model. This creates a new set of license files in the Oblicense directory, which are necessary to register the home PC.
2. Insert a blank disk into your home PC and create two directories: A:\HOME and A:\OFFICE.

   Note: You can name these directories anything you want. For these instructions, we are using HOME and OFFICE directories.
3. Log on to your host model, select the license group model and select About Group Model from the Help menu.
4. Click License, and click Transfer in the Group Model Licensing dialog.
5. In the CA Plex License Transfer dialog, click Register Target PC.
6. In the Register License Transfer dialog, enter A:\HOME, and click OK.

Your home PC is now registered. A registration file is placed in the HOME directory. The Office directory is still empty.

On your office PC:

1. Insert the disk into your office PC. Start CA Plex and access the CA Plex License Transfer dialog.
2. In the CA Plex License Transfer dialog, click Transfer License to Floppy.
3. In the Transfer License to Floppy dialog, enter A:\HOME, and click OK.

   A message box displays the status of the transfer.

4. Click OK.

   The group model on your office PC is now unlicensed—the license is on the disk. To return the license to this machine the next day, you must register the office PC now.

5. While still in the CA Plex License Transfer dialog, perform the registration procedure to register your office PC.

6. In the Register License Transfer dialog, enter A:\OFFICE, and click OK.

Both directories on the disk now contain data. The HOME directory contains the license. The OFFICE directory contains the office PC's registration information. The registration information is required for returning the license to your office PC.

Note: At this point, you cannot register the office PC again. If you try to register it again the following message appears:

```
Source directory already contains registration file.
```

You cannot transfer a different license to the office PC at this time. If you do, you will be unable to transfer the original license back to the office PC. This is because the site code on the office PC changes when you transfer a different license to the PC. If you attempt to transfer the original license back to the office PC, it will not work with the new site code.

On your home PC:

1. Insert the disk in your home PC and access the Group Model Licensing dialog.

2. Click Transfer.

3. In the CA Plex License Transfer dialog, click Transfer License to PC.

4. In the Transfer License to PC dialog, enter A:\HOME, and click OK.

   The license on the disk is transferred to your home PC. The HOME directory is now empty and your office PC is not licensed.

5. To return the license to your office PC, transfer the license to the OFFICE directory.

Chapter 5: Pattern Libraries

Pattern libraries are sets of reusable design objects on which you can base the applications you develop.

CA Plex provides the following sets of pattern and class libraries:

- Pattern libraries
- Class libraries (OBASE family)

Version and Level Names

The pattern libraries accompanying CA Plex r6.1 are called V6.0 Patterns.

Install Pattern Libraries

To install the pattern libraries, you must perform a Typical installation of the CA Plex base product. The pattern libraries are included with the Compact installation. For more information about installation instructions, see the chapter "Installing CA Plex" in this guide.

Install Class Libraries

To install the class libraries, you must perform a Custom installation (or modify your existing installation). For more information about installation instructions, see the chapter "Installing CA Plex" in this guide.

For more information about using the class libraries, see the CLASSLIBS.chm help file.

Install Pattern Libraries on a Network

In a typical workgroup environment (see Typical Configuration in the chapter "Installing CA Plex"), the pattern libraries are installed on a network drive where they are accessible by all developers. This ensures that all developers use the same version of the libraries, thus simplifying model administration.

To install the pattern or class libraries on a network server

1. On the server machine, run the setup program

   For more information, see Install the CA Plex Base Product in the chapter "Installing CA Plex."

2. Specify the directory where the libraries are to be installed.

   Note: You can use a UNC path (such as \\MyServer\MyShare\Libraries) as the location for the libraries. This means that all developers accessing the libraries on your network can also use the same path. The use of UNC paths is optional. If you need to access a library, and the path name specified at installation time is not recognized, CA Plex prompts you to enter a valid path.

3. Select the Custom installation option.

4. Select the libraries that you want to install and clear the other options.

Chapter 6: System i Components

This chapter describes how to install and configure the System i components—specifically for the System i client/server, System i 5250, and Java client—System i server products.

Note: CA Plex also supports the development of Java Server applications on the System i. For more information about the setup instructions, see the chapter "Java Components."

What Is Shipped to You

The following CA Plex for System i client/server and CA Plex for System i 5250 product libraries are on the CA Plex Product CD as binary files:

- PLEX610 library containing the remote configuration and run-time objects
- YTUTORIAL (CA Plex Tutorial library)
- YTUTREFER (CA Plex Reference Tutorial library)
- APPINTOBJ (CA Plex Application Integrator library)

Minimum System i Development Requirements

The minimum requirements for the System i development environment include the following:

- PLEX610 library containing the remote configuration and run-time objects
- TCP/IP
- i5/OS V5R3 or later. For more information about the supported version, see the readme
- RPG/400 compiler or ILE RPG/400 compiler

  Functions defined with the RPG400 (or SQLRPG400) system value are generated with RPG III syntax and require the RPG/400 compiler.

  Functions defined with the RPGIV (or SQLRPGIV) system value are generated with RPG IV syntax and require the ILE RPG/400 compiler.

  - DDS compiler.
  - UIM compiler (for System i 5250 generator only).

- QSYS2 library for System i CPIC communications in the system portion of your library list.
- 4 MB of disk space for the CA Plex product (for application development and run-time execution).
- Sufficient space for generated and compiled System i objects.
- A PC that meets the requirements of the PC Development Environment (System i client/server applications only).

Minimum System i Deployment Requirements

This section lists the requirements for a System i running the following:

- System i 5250 application
- System i client/server application
- Java-RPG/400 application

The components you need are as follows:

- 5250-type terminals, or a PC running a 5250 emulation package (System i 5250 applications only)
- i5/OS
- The following libraries must be in your library list when you call your function:
  - A data library, where your physical and logical files reside
  - A generation library, where your generated programs reside
  - The PLEX610 library

    Note: You can combine your data library and generation library into one library.

Library List Considerations

At run time, the PLEX610 library should be lower than your data and generation library in your library list.

Transfer the Product Libraries from CD to System i

Before restoring the CA Plex System i product libraries, you must first transfer the files from the CD to your System i. These instructions assume you have TCP/IP installed, configured, and active on your System i, otherwise, you cannot continue. Either contact your System i System Administrator about installing TCP/IP, or order the CA Plex System i product tape.

To transfer System i (binary) files from the CD to your System i

1. Get the IP address (for example, *nnn.nnn.nnn.nnn*), or the symbolic name (myas400.mycompany.com), of your destination System i from your System i System Administrator.
2. Log on to your PC.
3. Insert the CA Plex CD into an accessible CD-ROM drive.
4. Open a command line window on your PC.
5. Change drives to the location of your CD-ROM drive.
6. Type **cd as400libs**, and press Enter.
7. Type **ftp**, and press Enter.

   This starts an FTP session.
8. Type **open nnn.nnn.nnn.nnn** (*nnn.nnn.nnn.nnn* is your System i TCP/IP address—you can substitute the symbolic name for the IP address), and press Enter.

   You are successfully connected if you are prompted for your user profile.

   Note: If the session is idle for more than five minutes after you have connected, then the system automatically closes the connection. To restart the session, you must start the procedure over at Step 7.
9. Type your System i user profile at the prompt, and press Enter.
10. Type your System i password at the prompt, and press Enter.

    Note: The cursor does not move when you are entering the password.

    The system replies that your user profile is logged on.

11. Type **quote site namefmt 1**, and press Enter.

    Note: There must be a blank space between namefmt and 1.

    The system replies that the user is now using naming format **1**.

    Note: You must specify 1 as the naming format.

12. Type **bin**, and press Enter to set the binary format.
13. Type **cd qgpl.lib**, and press Enter.

    Note: You can specify another existing library in place of QGPL; the .lib extension is required.

    The system replies that the current library is changed to QGPL.

14. Enter **put plex610.savf**, and press Enter. The time to transfer the file varies depending on your network speed.

    Note: The .savf extension is required.

    The system replies that the file transfer is completed successfully.

15. Repeat Step 14 for ytutorial.savf, ytutrefer.savf, and appintobj.savf.
16. Enter **quit**, and press Enter.

    This stops your FTP session.

17. Exit the command line session.

You can now restore the *SAVF files.

Restore the Product Libraries from Save Files (*SAVF)

To install the CA Plex for System i client/server or CA Plex for System i 5250 components from save files

1. Log on as QSECOFR or with QSECOFR equivalent authority.
2. If you are upgrading an existing release of CA Plex, first clear the product libraries by entering the following command strings:

   ```
   CLRLIB LIB(PLEX610)
   CLRLIB LIB(YTUTORIAL)
   CLRLIB LIB(YTUTREFER)
   CLRLIB LIB(APPINTOBJ)
   ```

3. Restore the CA Plex product library by entering the following:

   ```
   RSTLIB SAVLIB(PLEX610) DEV(*SAVF) SAVF
   (QGPL/PLEX610) FRCOBJCVN(*YES *RQD)
   ```

4. Restore the Tutorial library by entering the following:

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

5. Restore the Tutorial library by entering the following:

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

6. Restore the Application Integrator library by entering the following:

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

7. Add PLEX**610** to the top of your library list.

PC-to-System i Communications Software

For System i development, CA Plex requires your PC to be linked to System i through TCP/IP.

CA Plex TCP/IP Environment Configuration

The following guidelines explain how to configure CA Plex TCP/IP protocol services according to the needs of your developers and end users. TCP/IP does not use, or require, the QCMN subsystem. Therefore, these instructions are based on QCMN not being present.

To use TCP/IP with this release of CA Plex, you must have the following:

- i5/OS V5R3 or later

  For more information about configuring TCP/IP for your System i, see the IBM publication, *TCP/IP Configuration and Reference* (SC41-34209-00). These instructions presume that TCP/IP is installed and active on your System i.

- TCP/IP configured and started on your System i
- TCP/IP version of the System i CA Plex Dispatcher started on the System i

For CA Plex clients to connect using TCP/IP, you must have the following:

- The TCP/IP CA Plex Dispatcher started on the System i
- A port number assigned when starting the TCP/IP Dispatcher program

YOBSYTCPDP TCP/IP Dispatcher Program

For CA Plex r6.1, we recommend the use of the YOBSYTCPDP Dispatcher program. This version of the Dispatcher program provides the following:

- Improved security. Specifically, this Dispatcher uses a single socket for all communications between the client and the server so that only the specified port needs to be open through a firewall.
- Support for the IPv6 protocol (in addition to IPv4).

The Dispatcher programs supplied with CA Plex r6.1 are *not* compatible with V5R2 and earlier.

V5R3 version of YOBSYTCPDP

Initially at the time of writing, V5R3 does not fully support IPv6. Therefore, a separate version of YOBSYTCPDP is required on V5R3. This version is supplied as two programs, YOBSYTCPC3 and YOBSYTCPD3. To install the V5R3-compatible version:

1. Rename the existing YOBSYTCPCT and YOBSYTCPDP programs within your PLEX library (PLEX610).
2. Rename the YOBSYTCPC3 program to YOBSYTCPCT and rename the YOBSYTCPD3 program to YOBSYTCPDP.

Note: Any running dispatchers should be stopped before any renaming occurs, then restarted, after the renames have occurred.

User Name Limitation on V5R3

At V5R3 only, one limitation of the YOBSYTCPDP Dispatcher is that it does not display the actual user profile for the Job User parameter of the spawned job name on the i5/OS WRKACTJOB screen. This limitation does not exist at V5R4 or later.

As a workaround for this i5/OS limitation on V5R3, you can run the YOBSYTCP dispatcher instead of YOBSYTCPDP. The YOBSYTCP dispatcher is not generally recommended because it is not firewall compatible and as well as not supporting IPv6.

YOBSYTCP_R dispatcher

Earlier releases of CA Plex also included a dispatcher program called YOBSYTCP_R. If you require the functionality provided by this program, contact CA Technical Support.

User Authority Requirements

It is not necessary to set the USER parameter to QSECOFR, but the user parameter you supply must have the following authorities:

- *USE authority for objects QSYS/QSYGETPH and QSYS/QSYRLSPH.
- *ALLOBJ authority is not required, however, the connecting client user profile must provide *USE authority and *OBJMGT (Object Management) to the user profile that started the Dispatcher. This requirement is necessary because the user profile that started the Dispatcher is then allowed to validate the connecting client user's i5/OS user profile and password.
- *SECADM authority is not required, however, the connecting client user profile will not be allowed to change its own expired password unless the user profile that started the Dispatcher has *SECADM authority. If you prefer to run the Dispatcher without *SECADM authority then you must provide alternative mechanisms for your end users to change their passwords. For more information, see Managing Passwords with System i TCP/IP Connections in the online help.

  Note: If the connecting client user profile is the same user profile that started the Dispatcher, it does not require *SECADM authority for resetting its own expired password.

Object Authorities

The Grant Object Authority (GRTOBJAUT) command or the Edit Object Authority command can be used to grant these object authorities.

Example:

The STARTDSP user profile starts the Dispatcher (YOBSYTCPDP or YOBSYTCP) on the System i, which does not have *ALLOBJ authority but has *SECADM authority. The CONNECTUSR user profile is the connecting client user profile.

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

Note: AUT(*OBJOPR *READ *EXECUTE) is equivalent to *USE authority when using the 'Edit Object Authority' (EDTOBJAUT) command.

In the previous example, the CONNECTUSR user profile will not be to able reset its own expired password because the STARTDSP user profile does not have Object Management (*OBJMGT) authority to the CONNECTUSR user profile.

In the next example, the CONNECTUSR user profile will be able to reset its own expired password because the STARTDSP user profile does have Object Management (*OBJMGT) authority to the CONNECTUSR user profile.

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

Start the System i TCP/IP Dispatcher

Review the preceding sections before you execute the examples on starting the System i TCP/IP Dispatcher.

Note: In these examples the parameter 61000 is the assigned port number. However, it can be any unused port number that is not in the range of port numbers previously registered with the Internet Assigned Number Authority (IANA). These registered ports are typically within the range of 0 to 1023. We recommend using port numbers between 3000 and 61000 to avoid conflicts with lower numbers as they become registered.

The YOBLISTEN job runs until it is ended manually or by the system IPLs. Set the SCDTIME parameter to a time after the IPL, when the STRTCP command was issued to complete the TCP/IP startup.

Example 1:

Write your own CL program to start PLEX610/YOBSYTCPDP or PLEX610/YOBSYTCP, and call the CL program as part of your IPL.

Example 2:

Create a Job Schedule Entry (ADDJOBSCDE).

The following is an example you can use if your IPL happens nightly:

```
ADDJOBSCDE JOB(YOBLISTEN) CMD(CALL )) FRQ(*WEEKLY)
PGM(PLEX610/YOBSYTCPDP) PARM('61000'
SCDDATE(*NONE) SCDDAY(*ALL) SCDTIME('hh:mm:ss')
JOBD(PLEX610/PLEX) JOBQ(QGPL/QINTER) USER(QSECOFR)
TEXT(Plex TCPIP C/S Dispatcher')
```

Note: The PARM value in the previous statements must include the single quotes or the call will fail.

After submitting the ADDJOBSCDE command, the Dispatcher will not start until the scheduled time after the next IPL.

Start the Dispatcher Before the Next IPL

You can choose to start the Dispatcher before the next IPL.

To start the Dispatcher before the next IPL

1. From the command line, enter the following:

   ```
   WRKJOBSCDE
   ```

2. Enter **10** next to the job, and press Enter.

The job will start immediately (typically, you would only do this once on the day you set up the Dispatcher).

Start Additional TCP/IP Dispatchers

You can start additional CA Plex TCP/IP Dispatchers on additional ports at any time for testing purposes. For instance, when first configuring CA Plex for TCP/IP to the System i, you may want to start the Dispatcher manually before adding the job to your startup routine.

To manually start the CA Plex TCP/IP Dispatcher on a port

1. From the command line, enter the following:

   ```
   SBMJOB CMD(CALL PGM(PLEX610/YOBSYTCPDP) PARM('61000')) JOB(YOBLISTEN)
   JOBD(PLEX610/PLEX)
   JOBQ(QGPL/QINTER)
   ```

   The parameter passed into the YOBSYTCPDP (or YOBSYTCP) program must be an available, unused port number.

2. Verify that your port has started by entering the following command on the command line:

   `WRKTCPSTS *CNN` (check the Local Port column for the port number you assigned)

Note: Use QINTER as the job queue so that incoming CA Plex clients get interactive response times. The job queue that starts the CA Plex TCP/IP Dispatcher is used for submitting individual client jobs as they are requested. Using a batch job queue may cause your CA Plex client jobs to sit in your batch job queues, subsequently causing a time-out and hanging behaviors on the client side.

How to Restore the Product Libraries Under a Different Name

You can choose to restore the product library under a different name instead of the default PLEX610. If you change the default name, you will need to manually update several references to the library name as described in the following sections:

- Update the Job Description
- Modify Files on the PC

Update the Job Description

To restore the product library under a different name instead of the default PLEX610, you must manually update the job description.

After restoring the library, enter the following command to update its JOBD:

```
CHGJOBD JOBD(library-name/PLEX610)
INLLIBL(QTEMP library-name YTUTORIAL YTUTREFER APPINTOBJ
QGPL)
```

Modify Files on the PC

The following instructions explain how to change certain default settings in CA Plex to use the name of the CA Plex System i product library that you specified (*Library-name* in this example) instead of the default name (PLEX610).

To modify files on the PC

1. Open a text editor, such as WordPad.
2. Modify the following files on the PC to reflect the unique System i library names that you used for the new version:

   Note: The following files are copied to your personal Documents folder (in the CA\Plex\6.1 sub-folder) when you first run CA Plex. If you have already run CA Plex, you may also want to update the copies in your personal Documents folder.

   - C:\Program Files\CA\Plex\6.1\Plex.bld

     Under the section titled [remote configuration], change the line

     `jobdesclib=PLEX610`

     to:

     `jobdesclib=`*`library-name`*

   - C:\Program Files\CA\Plex\6.1\Plex.INI

Under the section titled [Remote], change the line:

```
Library=PLEX610
```

to:

```
Library=library-name
```

- C:\Program Files\CA\Plex\6.1\Bin\Obsyrt.ini

  Under the section titled [Remote], change the line:

  ```
  Library=PLEX610
  ```

  to:

  ```
  Library=library-name
  ```

3. Using the same change for Obsyrt.ini, modify the following:
   - \Bin\Ob600rc.INI
   - \Bin\Ob600rcd.INI

Chapter 7: Windows C++ Server Components

This chapter describes how to configure the Windows C++ server components for CA Plex.

Note: The Windows C++ Server generator is provided only for maintenance of existing Plex WinNTC applications. CA strongly recommends the use of C# or Java generator for the development of new Plex applications for the Windows Server platform. Customers with existing WinNTC applications are encouraged to consider a migration to C# or Java.

Windows C++ Server Development Requirements

The software requirements for the Windows C++ server are as follows:

- Windows Server 2003 (for detailed version information, see the readme). 64-bit editions of Windows Server are supported. Plex WinNTC applications execute as 32-bit applications in the WOW64 environment. For native 64-bit application support, C# or Java generator must be used.
- Microsoft Visual Studio 2005 (Professional Edition or Standard Edition)
- CA Plex Windows C++ Application Server SDK
- Microsoft SQL Server or Oracle (for detailed version information, see the readme)

For more information about additional system requirements, see the readme.

Install the Windows C++ Server Components

To develop Windows server applications you must install the development version of the CA Plex Application Server.

Windows Client Requirements

You must run the CA Plex installation program on each developer's machine to install the client components. A typical installation of CA Plex includes the necessary components.

In addition, the CA Plex Windows C++ Application Server SDK must be installed on the server machine.

Application Server

The *Application Server* is the name for the services and components that support CA Plex C++ applications on Windows servers.

The Application Server includes the following:

- Environment Manager
- Build Service and Build Service Manager
- Dispatch Service and Dispatch Service Manager
- Printing Service

These components are installed in the Plex\AppServer\Bin directory.

Install the Application Server

To install the Application Server on a Windows Server 2003 operating system, the following authorities are required:

- Local Administrator authority is required if CA Plex has previously been installed anywhere in the domain.
- Domain Administrator authority is required if CA Plex has not been previously installed. The server must be part of a domain.

To install the Application Server when you install the CA Plex base product

1. When asked to specify which type of setup to install, click Custom.
2. Check CA Plex Windows C++ Application Server SDK in addition to what is already checked.

To install the Application Server after you install the CA Plex base product

1. Click Start, Settings, and Control Panel.
2. Double-click the Add/Remove Programs icon.
3. In the list of products, select CA Plex.
4. Click Change/Remove.
5. On the InstallShield Wizard window, select Modify, and click Next.
6. Select the CA Plex Windows C++ Application Server JDK check box.
7. Click Next to start the installation.

Job Status Database

To support the Build Service, the Application Server installation program automatically does the following:

- Creates a Microsoft Access database for storing status information about CA Plex builds. This job status database is located at \AppServer\Jobsts\Objobsts.mdb.
- Configures an ODBC data source for the job status database.

Build Service and the Dispatch Service Configuration

You have the option of configuring the startup information for both the installed services. The default installation sets the startup type to be manual. You can change this to automatic so that the services start at boot up.

For more information, see the online help (Ntservices.chm).

Oracle Support

You can use the Windows Server (WinNTC) generator to create applications that access data in Oracle databases on Windows Servers. The run-time application uses the native Oracle Call Interface (OCI) for fast data access.

Before using CA Plex, ensure that Oracle is properly installed on the server (see your Oracle documentation for details) and that a basic user (like the default user SCOTT) has been configured with the following privileges:

- Connect
- Resource
- Tablespace

Windows Client Components—Microsoft RPC Installation

Consider the following before installing the Windows Client Components—Microsoft RPC:

- CA Plex clients connect to the Application Server using the Microsoft RPC run-time services.
- Using the CA Plex Environment Manager, create a user profile with proper authorities on the Windows Server for each CA Plex developer. For more information, see Creating User Environments in the online help.
- For information on setting up a database, authorizing users, and configuring a data source name, see your Microsoft SQL Server documentation.

Windows C++ Application Server

The CA Plex installation has a Windows C++ Application Server option. When selected, this installs only the components necessary to deploy your application (the Dispatch Service, Printing Service, and the Environment Manager).

Chapter 8: Java Components

This chapter describes how to install and configure the Java components for CA Plex. For more information, see the Java Platform section of the online help.

Java Development Requirements

CA Plex supports the generation of Java applications on both the client and server. Java applications can be developed and deployed in the following configurations:

- Java client to RPG (System i) server
- Java client to Java server
- Java client or server to C# .NET server
- C# .NET Server to Java Server
- Windows client to Java server

CA Plex generates 100 percent pure Java code that can be run under any Java Virtual Machine on any platform. If you have any existing Java applications generated by CA Plex, run-time backwards compatibility enables new versions of the CA Plex Java run time to be deployed without regenerating existing applications.

CA Plex Java clients are primarily tested on Windows. Java client applets are tested using the Sun Java plug-in. Note that CA Plex Java clients require the Sun Swing classes to support GUI components.

The following table summarizes the installation requirements for CA Plex Java development. Deployment requirements are documented in the Java Platform section of the online help.

Component	Description
Java client	■ CA Plex Java components ■ Sun Java SE JDK 6.0. Other versions of the JDK work with CA Plex, but they have not been fully tested. Freely downloadable. ■ The JAVA_HOME environment variable must be defined.
Java server	The same as Java client, with the addition of: ■ Third-party DBMS and JDBC driver (for runtime data access) ■ ODBC driver (for database builds, not required at run time) For System i Java server development: ■ PLEXJVA610 library ■ IBM Client Access (or equivalent) ■ IBM System i Toolbox for Java

How to Install Java Components

The CA Plex Java components are installed automatically when you install CA Plex. They include the following:

- ObJava directory
- Apache ANT
- The ob610dsp.bat file in the ObJava directory

To install and configure the Java components, do the following:

1. Download and install the required version of the Sun Java SE Development Kit (JDK).
2. Set the JAVA_HOME environment variable.
3. Configure the server for run-time.
4. Transfer Java server functions (JAR file) to the server.
5. Start the Java Dispatcher.

Define the JAVA_HOME environment variable

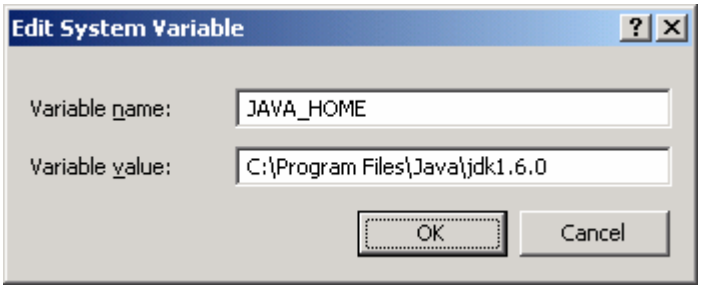
The JAVA_HOME environment variable determines the Java compiler (Javac.exe) that will be used for Plex Java builds. Plex does not automatically set this environment variable which provides you with the flexibility to switch between different versions of the compiler. This setting is not required to run generated applications.

To define the JAVA_HOME environment variable

1. Double-click the System icon in the Windows Control Panel
2. On the Advanced tab, click the Environment Variables button.

   You can define JAVA_HOME either as a user variable or as a system variable.
3. Click the appropriate New button, and enter the Variable name as JAVA_HOME and the Variable value as the full path of the folder where the JDK is installed. For JDK 1.6, the full path will be *C:\Program Files\Java\jdk1.6.0*.

   Note that JAVA_HOME should NOT point to the Bin sub-folder of the JDK installation folder. For example:



4. Click OK.

You have defined the JAVA_Home environment variable.

How to Configure the Server for Run-time

To configure the server for run-time, do the following:

1. Copy the ObJava directory from the CA Plex directory on your client machine to the remote server.

   Note: You do not require some sub-folders of Objava run-time applications. For example, you do not need to copy the apache-ant and src sub-folders.
2. After building your Java application's server functions on a client, transfer the resulting JAR file to the server.

3. Administer the Java Dispatcher for CA Plex by one of the following methods:
   - Using a web server that processes a shipped servlet
   - Using a batch file that runs the Java VM interpreter within the session

   Using the batch file is the simplest option. The use of a web server is more appropriate for production applications.

For more information about instructions on starting the Dispatcher, see Using a Batch File to Start the Java Dispatcher in the online help.

For more information about web servers, see Setting Up a Web Server in the online help.

As an alternative to the Java Dispatcher, you can use the EJB runtime proxy to manage communications in distributed Java applications. For more information, see the online help.

Set Up Java on the System i

To run CA Plex Java functions on the System i, you must install the PLEXJVA610 library and the CA Plex Java components onto your System i.

These instructions assume you are using the System i Toolbox for Java. The use of IBM Client Access (or equivalent) is also recommended for accessing the Integrated File System (IFS) on the System i.

Note: If you are developing Java client to RPG server applications, you can ignore this section. To connect to RPG server functions on a System i, Java clients use the System i TCP/IP dispatcher that is included in the PLEX610 product library. The installation process is the same as for Windows clients and is described in the chapter "System i Components."

PLEXJVA610 Library

The PLEXJVA610 library is supplied as a save file in the AS400 Libraries folder on the CA Plex CD-ROM. This must be transferred to the System i and restored in the same way as the other CA Plex System i libraries. For more information, see Transferring the Product Libraries from CD to System i and Restoring the Product Libraries from Save Files (*SAVF) earlier in this guide.

How to Copy the Objava Directory to the System i

To copy the Objava directory to the System i, perform the following steps:

1. Install the CA Plex Java components onto a Windows machine, as described in Install Java Components section in this chapter.
2. Create a directory at the top-level of the System i IFS. In these instructions, we assume the directory name is Plex.
3. Copy the ObJava directory and its contents from your Windows machine to the System i, to create a Plex/ObJava subdirectory.

How to Configure the Java Dispatcher on the System i

The following instructions assume you are using the JDBC driver from the System i Toolbox for Java. The Dispatcher must be configured with the necessary JDBC settings to ensure that the System i Toolbox for Java classes are available on the class path.

To configure the Java Dispatcher, perform the following steps:

1. Create a directory at the top-level of the IFS. For example, /PlexJava.
2. Change the settings in the ob600srv.properties file. Descriptions of these settings can be found in the CA Plex help. A typical example follows with the necessary modifications in bold. Each entry should be on a single line.

```
Environment.Default.Driver=
    com.ibm.as400.access.AS400JDBCDriver
Environment.Default.DataSource=
    jdbc:as400://MACHINE/LIBRARY
Environment.Default.User=USERID
Environment.Default.Password=PASSWORD
```

Java Classes Optimization

You must optimize the system .class files for them to perform well on the System i. This only needs to be done once.

You use the CRTJVAPGM command on a single file, an entire JAR file, or an entire directory (using wildcards). There are four levels of optimization you can choose. Unless you have a reason to do otherwise, use full optimization (level 40). The command creates an optimized hidden file for each .class file that is loaded by the System i Java Virtual Machine. For more information about this command, see the System i online help.

Run CRTJVAPGM as a batch process on the following files:

- obrun.jar
- JT400.jar (you can copy this file from the default location to the Plex directory you created earlier to make the path shorter)

Start the Java Dispatcher on the System i

You can use the YSTRJVADSP command to start the Dispatcher. You can also use the RUNJVA command to start the service.

To start the Java Dispatcher using the YSTRJVADSP command

1. Add PLEXJVA610 to your library list.
2. Use the YSTRJVADSP command to start the Dispatcher.

   Note: This command includes parameters for the port number and locations of the run-time and properties files. The path to JT400.jar file should be referenced in the Path To Additional Class parameter.

For more information about this command, see YSTRJVADSP in the Index of the CA Plex online help.

A second method of starting the service is by using the RUNJVA command.

To start the service using the RUNJVA command

1. Enter **RUNJVA** on a System i command line and press F4.
2. For the Class parameter, type in the name of the run-time class that requires service:

   ObRun.ObComms.ObService—to start the Java Dispatcher

   Note: The class names are case-sensitive.
3. For the Parameters parameter, type in the port number and the path to the properties file, for example, 1998 and /Plex/Objava.
4. At the Classpath prompt, type in the full path to the CA Plex Java run time, for example, /Objava/lib/obrun.jar.

Chapter 9: CA 2E Data Migration

This chapter describes how to install the CA 2E Data Migration product.

The CA 2E Data Migration product supports the migration of relational data modeling meta-data from CA 2E to CA Plex. It does not support the migration of action diagram code and panel designs. For complete CA 2E to CA Plex migration support, see the CA partner solutions from ADC Austin (www.adcaustin.com).

What Is Shipped to You

The CA 2E Data Migration product includes the following migration libraries in the AS400 Libraries directory on the CA Plex product CD-ROM:

- YOMF
- YOMFVENG
- YOMFDB

CA 2E Data Migration Requirements

For PC and System i prerequisites, see the following:

- PC Development Environment
- Minimum System i Development Requirements

This section describes only the prerequisites that are specific to CA 2E Data Migration.

Memory

The minimum memory requirement for CA 2E Data Migration is 1 GB RAM.

Since Data Migration can populate large CA Plex models quickly, consider increasing the amount of RAM on the PC used for development, if you are importing a large CA 2E model.

Minimum System i Prerequisites

The requirements include the following:

- i5/OS
- A PC-to-System i communications software package (for more information about specific requirements, see the chapter "Upgrading from Earlier Releases of CA Plex" in the *Release Notes*.)
- 20 MB of disk space for the CA 2E Data Migration product

How to Restore the Product Libraries

Use the following instructions to restore the CA Plex objects from the product CD to the System i.

1. Transfer the migration files, YOMF.savf, YOMFVENG.savf, and YOMFDB.savf, from the CD to your System i. For more information, see Transfer the Product Libraries from CD to System i.
2. If you are using the product CD, follow the steps in Restoring the Product Libraries from Save Files (*SAVF).

   Restore the files, YOMF.savf, YOMFVENG.savf, and YOMFDB.savf.
3. If you are upgrading an existing release of CA Plex, note that the product library names for Data Migration have changed. Though it is not necessary for this installation procedure, you may want to delete the old libraries using the following command strings:

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

   Note: If you have not completed the steps in PC-to-System i Communications Software, enter the following command strings:

   ```
   CLRLIB LIB(PLEX610)
   CLRLIB LIB(YTUTORIAL)
   CLRLIB LIB(YTUTREFER)
   ```
4. Restore the CA Plex run-time libraries:

   If you are using V3R6 or later, enter the following:

   ```
   RSTLIB SAVLIB(PLEX610) DEV(device) ENDOPT(*LEAVE) RSTLIB(PLEX610)
   FRCOBJCVN(*YES *RQD)
   ```
5. Restore the Tutorial library by entering the following:

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

6. Restore the Tutorial Reference library by entering the following:

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

7. Add PLEX610 to the top of your library list.
8. Restore the Migration main product library as follows:

   ```
   RSTLIB SAVLIB(YOMF) DEV(device) ENDOPT(*LEAVE) RSTLIB(PLEX610) FRCOBJCVN(*YES
   *RQD)
   ```

9. Restore the Migration user interface library as follows:

   ```
   RSTLIB SAVLIB(YOMFVENG) DEV(device) ENDOPT(*LEAVE) RSTLIB(PLEX610)
   FRCOBJCVN(*YES *RQD)
   ```

10. Restore the Migration Mapping database as follows:

    ```
    RSTLIB SAVLIB(YOMFDB) DEV(device) ENDOPT(*LEAVE) RSTLIB(PLEX610)
    FRCOBJCVN(*YES *RQD)
    ```

Parallel CA 2E Installations

If you are working with parallel CA 2E installations, you must adapt the YIMPS2OBSY/QBATCH job description library list to fit your environment. Adapt your job description to reference the correct libraries if you have done the following:

- Placed CA 2E objects in additional libraries
- Consolidated CA 2E objects into fewer libraries
- Renamed your CA 2E libraries

Chapter 10: Application Integrator

This chapter describes how to install Application Integrator. *Application Integrator* enables reverse engineering information from several sources into CA Plex.

There are several data sources that you can import into the Application Integrator. Each one is a separately licensed product.

- Schema/ODBC—Enables the integration of data schemas from any ODBC data source. This replaces the previous ODBC Import Utility.
- Schema/400—Enables the integration of a DDS-defined data schema from the System i. This replaces the previous DB2/400 Import Utility.
- Application Integrator/400—Enables the integration of i5/OS programs (hand-coded or generated by other tools).
- Application Integrator/2E—Enables the integration of CA 2E functions direct from an CA 2E model.

For more information about using Application Integrator, see the online help.

What Is Shipped to You

The CA Plex product CD includes the following:

- The Application Integrator tool
- The APPINT System i library, installed as a save file in the Application Integrator directory

System Requirements

The requirements specific to Application Integrator are as follows:

- PC Development Environment. Large imports may require more than the recommended minimum memory.

  Note: For more information about PC Development Environment, see the System Requirements section in the readme.

- Minimum System i Development Requirements, which include the following:

  - i5/OS

    Note: For more information about supported version, see the readme.

  - For System i connections, a TCP/IP connection

    Note: For more information about Minimum System i Development Requirements, see the chapter "System i Components."

Install the Application Integrator

You can install the Application Integrator while installing the base product or after the base product installation is complete.

To install the Application Integrator tool when you install the CA Plex base product

1. Click Custom when asked to specify the type of setup to install.
2. Select Application Integrator in addition to what is already selected.

To install the Application Integrator tool after you install the CA Plex base product

1. Click Start, Settings, and Control Panel.
2. Double-click the Add/Remove Programs icon.
3. In the list of products, select CA Plex, and click Change/Remove.
4. On the InstallShield Wizard window, select Modify, and click Next.
5. Select the Application Integrator check box, and click Next to start the installation.

Set Up the ODBC Data Source

At the end of the installation process, you may see a warning message indicating that no directory has been specified for ODBC file DSNs on your system. If a warning message appears, you will have to complete the set up manually.

To set up the ODBC Data Source

1. Locate the file Appint.dsn in the Application Integrator directory.
2. Copy this file to the default directory on your system where ODBC file DSNs are stored. This is typically C:\Program Files\Common Files\ODBC\Data Sources but can be changed using the ODBC Administrator.

Note: If you did not see a warning message, these steps are not necessary.

Prepare to Import from the System i

If you are importing from a System i-based data source, you must first install the CA Plex product library, PLEX610, and the Application Integrator product library, APPINT on the System i. If you are only using the Schema/ODBC or Schema/Biz modules you can ignore this section.

To restore the CA Plex objects from the product CD to the System i

1. Transfer the save files from the CD to your System i. Follow the steps in the Transferring the Product Libraries from CD to System i section in the chapter "System i Components."

   The appint.savf file is installed in the Application Integrator directory. The plex.savf is available in the AS400 Libraries folder on the product CD.
2. Follow the steps in Restoring the Product Libraries from Save Files (*SAVF) in the chapter "System i Components" if you are using the product CD.
3. To restore the APPINT library, enter the following:

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

Multiple Versions of CA Plex

If you are running multiple versions of CA Plex, edit the APPINT/APPINT job description to use the correct names for the PLEX and APPINT libraries.

Chapter 11: Frequently Asked Questions

This section contains some frequently asked questions and their answers.

Questions and Answers

Q: When should I back up my group model?

A: You should perform regular backups of your group models on a daily basis. A group model is the central design repository. It is extremely important. If you accidentally destroy a local model, all the work you have done since the last extract from the group model is lost. If the group model is accidentally destroyed, all the work for that group model is lost. It may be possible to recover all or some of the data from a local model (using the XML Import/Export feature) but this is not guaranteed in all cases. For more information, see Backups in the online help.

Q: How often should I update?

A: We recommend that you update your changes to the group model at least once a week, but preferably more often. Frequent updating in a workgroup environment makes your changes available to other developers. More importantly, after changes are updated to the group model, they become part of the central repository and are more secure. Note that extended periods between updates affect the time it takes to do an update; the more work that CA Plex has to do to get through the differences between your local model and the group model, the longer it takes.

Important! Do not wait several weeks between updates.

Q: Why does my function, field, table, or view have two implementation names?

A: You and someone else updated the group model after having generated and built functions and panels or tables or views. Your implementation and file name triples and objects—and those of the other person—were added to the group model without one set replacing the other. If this happens, you must delete the duplicate names.

Note that a group model conflict occurred when the second person tried to update the group model. You can choose not to update at this point. To avoid such conflicts in advance, ensure you frequently update changes, especially after making changes that impact other developers in your workgroup. For more information, see Avoiding and Resolving Conflicts in the online help.

Q: I tried to generate an object but I got an error stating that I must specify an implementation language. What went wrong?

A: If you inherited the object from a pattern library (such as OBASE) make sure that the configuration of your local model is correct. To do this, choose Configuration from the File menu and, in the Model/Library box, select each model in turn. For each one, make sure that the Variant option is not set to Base. Instead, select the appropriate variant for your current work.

Q: What is the difference between levels and versions in group models?

A: A version is a collection of levels in a specified sequence. Assigning work to various levels means that you can keep track of general fixes and customer specific fixes simultaneously. For example, in the application FunkyApp, after Release 1.0 two fixes were made for customer A and customer B, followed by a general release of fixes at Release 1.1. This resulted in four levels: Release 1.0, Customer A fixes, Customer B fixes, and Release 1.1. The following table shows the version and level combinations possible. For more information, see Working with Versions and Levels in the online help.

Version	Levels Include	Description
Release 1.0	Release 1.0	The first generic release of FunkyApp 1.0 for all customers
Customer A 1.0	Release 1.0 Customer A fixes	A FunkyApp 1.0 release specific for customer A
Customer B 1.0	Release 1.0 Customer B fixes	A FunkyApp 1.0 release specific for customer B
Release 1.1	Release 1.0 Release 1.1	Generic FunkyApp 1.1 release for all customers except A and B
Customer A 1.1	Release 1.0 Customer A fixes Release 1.1	A FunkyApp 1.1 release specific for customer A
Customer B 1.1	Release 1.0 Customer B fixes Release 1.1	A FunkyApp 1.1, release specific for customer B

Q: I cannot use constants or literal values in action diagrams. Why?

A: It might be faster if you could enter literals anywhere in the code, as you can in most programming languages, so why does CA Plex not permit this? The reason is maintainability. Too many programs in too many languages have literals that are coded into the programs with no explanation as to the significance of the letter or numeric value. As an enterprise development tool, CA Plex was designed to enable applications to be easily maintained by someone other than the original developer. Hence, you are forced to label all literals all the time to make your programs easier to understand.

Q: What are all these +++ meta-operations in library functions?

A: Meta code is used in pattern libraries to interrogate the triples in the model and provide the expected functionality. For example, a field with a triple MyField default MyDefault does not provide any default values without supporting meta code in the action diagram to set that default value. To learn more about meta code, Meta-operations, click the Locate button to see related topics in the Contents tab in the online help. Work on customizing supplied library code by copying and altering the code of others.

Q: When should I abstract my code into a pattern?

A: Sometimes a pattern is obvious, but the extra effort to make a great, inheritable, customizable pattern is not worth the effort because you may only implement it twice. On the other hand, something that obviously should be a pattern may be difficult to abstract. Often, the answer is that after you implement it once—certainly when learning about abstracting something for the first time—it is then clearer and simpler to work out whether you should abstract it and how you can turn it into a pattern. To know when to abstract becomes easier with experience.

Q: I cannot see my field names in the action diagram debugger. Why?

A: In the C++ build options, select the Force Build Of Selected Objects; then rebuild your functions. If this does not work, delete the entire Obj subdirectory, and then rebuild. For more information, see Generating and Building for Action Diagram Debug in the online help.

Q: How can I simplify parameter mapping?

A: An important technique is the management of field domains. Field domains control how CA Plex automatically maps parameters. When you perform mapping manually, field domains also restrict the list of available fields, which makes it easier to find the one you want. For more information, see Field Domains in the online help.

Another technique is to use default mapping in the Parameter Mapping dialog. Default mapping lets the Action Diagrammer automatically make its best guess when mapping parameters between function calls, saving you from mapping each parameter individually. For more information, see Setting up Default Mapping in the online help.

Q: What is the difference between pre, post, and edit points? Which should I use?

A: Edit points are left over from previous versions of CA Plex. Collection points (the collective name for pre and post points) are an improved place to put your code. The difference is that edit points overwrite each other at the different levels down the inheritance tree (you have to be extremely careful with the nesting), whereas collection points are cooperative and just collect code together in sequence. You can always add code to the bottom of a collection point in any action diagram, regardless of whether it has been used before. Edit points are restricted because if they have been used, you can no longer use the edit point further down the inheritance tree. Therefore, we recommend that you do not use edit points.

So which should be used: post or pre points? Sometimes it is obvious. You want to put code before or after another piece of code in the edit, pre, or post point so you know where to put it. What if it does not matter if the code goes in the pre or post point? Then the code always goes into the post point. If someone inherits from your function and wants to add code, if your code is in a post point they can add code before it (in the pre point) or after it (remember that you can always add code to the end of a collection point, regardless if it has been used higher in the inheritance hierarchy). If you add your code habitually in the pre point, people can only add code after (never before) your code if they are further down the inheritance hierarchy. In conclusion, always use post points unless there is a reason not to. For more information, see Edit Points and Collection Points in the online help.

Q: Which post point do I use?

A: This is perhaps the most difficult issue within CA Plex for the novice. we recommend that you do the following:

1. Make an educated guess.
2. Expand the code before and after the post point.
3. Consider whether this is the correct post point.
4. Test it and see.
5. If it does not work, try Step 3 again; then try Steps 1 to 5 again.

Knowing which post point to use is the key to writing CA Plex functions quickly. Look at other functions and determine why the local modifications were added at the particular points that the developer chose. Remember that the class and pattern libraries are based on the use of edit and collection points, so they can be examined for clues. Sometimes it helps to open action diagrams from the libraries, for example examining UISTYLE/EditDetailGrid shows you how to integrate UISTYLE/Detail and UISTYLE/Grid, and how to use UISTYLE/UIBasicShell.

Examining other pieces of code, particularly the libraries, helps to increase your total knowledge of which post point to use and when to use it. This increases your ability as an CA Plex developer.

Q: Why do I have to keep on dragging my line to the edit box at the top of the window?

A: You do not have to do this. You can press the Insert key on the keyboard to enable inline editing. Note that for multiline comments, you must press Ctrl+Enter to enter the line into the action diagram, but for a normal single line of code, just press Enter when you have finished editing.

Q: How do I delete a field or a variable from a function?

A: You can do this in the Model Editor by deleting triples. You cannot just press the Delete key on the Variable Palette. Go to the top line of the function or the Variable Palette (they both show the function name) and press F12. This calls a Model Editor focused on the function. Any fields and variables that may be deleted are listed in the triples. Note that inherited fields and variables are not listed. To delete an inherited field you must delete the triples from the function that specified them higher in the inheritance hierarchy. You cannot delete the fields and variables of library functions. Remember to press F5 (Refresh) to redisplay the variable palette to show that the fields and variables have been deleted.

Q: How can I document and add comments to my code?

A: Use the Comment and Seq constructs frequently. Take AllFusion of the CA Plex long object names to produce meaningful names. Carefully consider the naming of subroutines and edit points and use block narratives where appropriate.

Q: How can I indicate that a field is supposed to be a dual (call by reference) parameter?

A: Change the triple in the Model Editor from MyFunction input MyField to MyFunction dual MyField. You cannot do this from the Action Diagrammer.

Q: I changed the size of a Windows panel in the Panel Designer but at run time the size did not change. Why?

A: The Save Placement property of the panel is probably set to Yes and the previous size is probably still saved in the application .ini file. To see the change, delete the entry in the .ini file (or rebuild it using the Create Exe command) or set Save Placement to No.

Q: My button on the panel does not do anything, even though the code is there. Why?

A: The physical event in the panel has most likely not been mapped to the CA Plex logical event. A *physical panel event* is the result of somebody clicking, dragging, pressing, double-clicking, or selecting something on the panel. For example, pressing a button or selecting a menu item or entering text are all regarded as physical events, because they occur visibly on your computer screen.

The other events in CA Plex are *logical events*. Logical events only happen within CA Plex. They are not visible to you. Instead CA Plex lets you tie a physical event (clicking) to a logical event. The action diagram, and in particular the events handler, can respond only to logical events.

If you right-click the button (when it is selected) in the Panel Designer and choose Event Mappings, you can see the Event Mappings dialog. Click Pressed in the list box on the left (Physical event) and check that the list box on the right (Logical event) has the corresponding event selected in blue (for example, MyButtonPressed). If it is not selected in blue, select it before clicking OK to close the dialog. Switch back to the action diagram—pressing F11 switches you quickly between a function and its panel—and press F5 to refresh. Find the event code (an Event: Event MyButtonPressed construct in the Events Handler construct), which should now be executed when you press the button.

If you want to check which physical events are mapped to which logical events, look in the Events folder in the Panel Palette in the Panel Designer. If a logical event has one or more physical events assigned to it, it can be expanded and the physical event or events are displayed.

Q: Why does the MDI parent not respond to common events triggered in the MDI children?

A: The Menu ID properties of the menu items are most likely not set. Open the MenuShell panel of MyMDI in the Panel Designer, and expand the Menu bar folder in the Panel Palette. Find the menu items that are intended to be duplicated across all of the children but responded to by the parent, and ensure that they have a MenuId property of a number. Do not use the numbers 1 to 500, 54016, or 54107.

Q: I have Menu IDs, but my MDI child panel does not trigger MDI panel events. Why?

A: All MDI child panels and the MDI parent panel inherit from the common MenuShell panel, so do not expect the child functions to pick up the Menu ID property numbers you have set until you recompile them. When you recompile the children, the individual panels will know the Menu ID numbers and be able to pass them to the parent. Remember to recompile the MDI parent because it needs to know the Menu ID numbers.

Q: How do I delete a menu item?

A: Press the Delete key. There is no selection to delete a menu item from the right button pop-up context menu.

Q: I cannot see the new values I have entered for my status fields. Why?

A: If you add values to a field in the Model Editor and then go into a panel, these new values are not shown on the field. In the Panel Designer, just select the field and choose Refresh Values from the pop-up menu. The values are refreshed to match the triples in the Model Editor.

Q: Is there a way to see the full text of a message in the Message Log?

A: Yes, right-click the message.

Q: I cannot see the changes I have made reflected in other windows. Why?

A: Click the Refresh button (F5) to see your changes. In some cases, you may need to close both a panel and its related action diagram (answering Yes when prompted to save changes), and reopen them to see changes.

Q: I added a triple in the Model Editor but now I cannot find it. What happened?

A: There are a number of factors that determine which particular triples get displayed in the Model Editor. Make sure that you check each one. For more information, see Understanding the Model Editor Display in the online help. In addition note the following:

- The Model Editor does not display inherited (implicit) triples.
- Triples that you add to an inherited object may not be visible in an unfocused Model Editor window because there is no triple referencing the inherited object. In such cases, focus the Model Editor by dragging-and-dropping the inherited object from the Object Browser.

Q: Why does CA Plex permit me to enter incomplete or illogical data?

A: While CA Plex enforces correct syntax, it does not enforce consistency or completeness until you generate. Thus, the model could not contain the triple *Credit limit length Customer*, but it could contain both *Credit limit length 6* and *Credit limit length 7*. CA Plex does this for the following reasons:

- Sometimes during design, not all of the properties of an object are known. Under these circumstances, a tool that does not let you record anything until everything is known can hurt rather than help.
- In a workgroup environment, it is sometimes required that several different values of a property be recorded simultaneously. So, two team members may disagree on the length of the credit limit field. CA Plex enables both lengths to be recorded and the disagreement to be resolved later.
- In a workgroup environment, it is inefficient or simply impossible to enforce rules interactively. Not all the necessary triples may be present in your local model at a given time.

Q: There is no list of target object types available in the Model Editor. Why?

A: The target object type on the Entry Bar is not capable of input because its value is always determined by the verb that you select.

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.