

PAC Facilities

This section describes PAC facilities for customizing the PAC system and for expediting your work with PAC entities and objects. It also summarizes the PAC reporting system and administrator functions.

- Customizing PAC
 - Applications
 - Incorporation
 - Predict Case Support
 - Object Versions
 - Cross-Reference Data
 - Support for Dynamic Source Variables
 - Object Lists
 - Reporting System
 - Administrator Functions
 - Utilities
-

Customizing PAC

Application Program Interfaces

A PAC / PAA application program interface (API) lets you call a PAC function from a user-written program without logging on to PAC and invoking the function from within PAC / PAA. APIs provide flexibility in accessing PAC information and interfacing with your existing systems.

For example, you can create a maintenance request or migration event without entering the PAC / PAA system. When you enter information on a front-end screen, the information is communicated to PAC / PAA.

The maintenance-request API lets you interface an external problem-tracking system with PAC. You can validate maintenance requests against the external system or extract data from it.

Another API lets you retrieve statistics for an object, display source code for an object version, and edit the source code from the work area.

PAA delivers APIs specifically for reporting use. These APIs enable the user to design and customize user reports based on PAA (Production) information.

PAC includes APIs for working with the following entities:

- Application
- Status
- Application status link

- Migration path
- Migration event
- Object list for an event
- Event authorization
- Audit report for an event
- Maintenance request
- Object version (displaying information, loading Natural source code)
- Automatic compilation of foreign objects
- PAA entity reporting

PAC Express

PAC Express is a delivered EXAMPLE application using the APIs and demonstrates the use of such APIs. It tries to simplify the presentation of PAC to users and reduces the number of actions required to execute a PAC function. PAC Express integrates the PAC APIs to provide simplified screens from which you can perform PAC functions without entering the PAC system.

PAC Express can reduce by half the number of actions required to execute some functions, such as adding, authorizing, and submitting migration events. The modifiable source code is provided so that you can tailor PAC Express precisely to the requirements at your site.

User Exits

At various points in the system, PAC provides user exits that pass control to a user-written Natural program. When activated by the administrator, these exits allow you to do the following:

- Add site-dependent code into PAC processing for auditing, security verification, or additional rules,
- Add functionality to certain PAC processing,
- Override default change-management procedures.

For example, if the user exit PACEX010 is active, it is invoked during the validation of an object list for a migration. The PAC administrator can view each entry in the list and disallow the migration of specific objects.

More than 30 user exits are available in PAC and PAA.

Applymods

System applymods change certain PAC or PAA processing defaults. For example, by default the authorizer of a PAC migration event can override some event parameters during authorization. When Applymod 1 is active, the parameters cannot be overridden during authorization.

Another example is that by default, PAC creates a new version of an object even if the object fails to compile successfully during a migration from a development or maintenance status. When Applymod 14 is active, PAC rejects an object with compilation errors; no new version is created.

The authorizer of an event can activate or deactivate applymods by selecting from a list. The PAC administrator establishes the default Applymod settings and can prevent specified settings from being overridden by the event authorizer.

There are more than 30 system applymods for customizing PAC and PAA.

Applications

To define a new application, copy the following elements from an existing application:

- the base application definition (name, level, and processing defaults),
- the base application definition and its status links,
- the base application definition, its status links, and their associated migration paths.

This facility simplifies the process of defining each application and its life-cycle to PAC. Once the first application is defined, others can be defined quickly by copying and modifying the first definition.

Incorporation

Incorporation allows you to bring objects already in production into PAC without recompiling them. This facility is useful when implementing PAC at a site with a large number of application objects in production. An incorporated application establishes the base level for subsequent development, testing, implementation, and maintenance in the PAC-controlled environment.

To incorporate an application, you first define an incorporation status and link the application to the status, specifying the location of the application objects in the application status link. When you migrate objects from the incorporation status to the Control status, PAC assigns them version numbers. Objects can be incorporated directly from a Natural library or using the Include mode in the migration event, you can incorporate objects from a work file.

Predict Case Support

PAC supports Predict Case applications. PAC verifies each object against the Predict Case library definition. Once an application is under PAC control, PAC assigns version numbers and builds cross-reference information about the relationships among the Natural and Predict Case objects.

Object Versions

Aligning Versions

Alignment is a Control-to-Control migration, which recompiles application objects so that they all reference the same versions of subordinate objects. When it recompiles the objects, it assigns them new version numbers.

By default, alignment uses the most recent version of each object. However, you can override the default and instruct PAC to use a different version by specifying the version number or the status in which the preferred version exists.

Aligning an application has the following benefits:

- It simplifies subsequent maintenance.
- It helps you keep track of object versions and dependencies.
- It prevents anomalies (which can occur, for example, when a program that uses copy code is not recompiled after the copy code is changed).

Using the Generate List function, you can also align the objects of an application in two statuses so that the same object versions exist in each status. For more information, refer to the section [Generating an Object List](#).

Archiving Obsolete Versions

The PAC archiving facility unloads inactive object versions to an external storage device and purges the physical object from the PAC-controlled environment, leaving only a control record. The unload is performed in batch. Object versions can be restored if necessary.

PAC includes automatic selection of object versions for archiving. The automatic selection is based on the following site-specified criteria:

- The number of object versions in the application
- The age of the object version
- The status in which the object version exists

A user exit can be used to override default criteria for selected objects.

You can use a reporting routine to write the results to a work file for subsequent editing or processing.

Using a batch command, you can dynamically create multiple migration events, including object lists, to archive a range of applications.

Cross-Reference Data

PAC supports Predict Xref (cross-reference) data for objects. The Xref data is stored in the Predict file specified in the application status link. Xref data is only available for Natural and Predict objects and not for foreign objects.

Xref data is useful in nearly all phases of the application life-cycle. It can assist in finding errors and inconsistencies in objects, analyzing the impact of proposed changes, and identifying existing resources that can be used in an application.

The following is a partial list of Xref information that can be retrieved for an object in a specified status:

- External objects invoked by the object
- Methods used to invoke Natural objects (for example, CALLNAT)
- Entry points or functions that can be invoked by other objects
- Copy code used to compile the object
- Maps and help routines used by the object
- Variables and data areas used by the object
- Files and fields used by the object and the type of file/field use (for example, reading or updating)
- Natural error messages used by the object
- Processing rules used in a map
- Work files used by the object

For more information, refer to the Predict Reference Documentation.

PAC automatically cross-references object versions. Thus, you can view the specific version of an object that is used by another object version. PAC cross-reference data for object versions is called object usage data.

Support for Dynamic Source Variables

The source code for some programs is compiled at execution time using the current values of dynamic source variables. These programs must be RUN; therefore, the source code must be migrated to environments where the programs are executed.

PAC and PAA support applications using dynamic source variables by allowing you to migrate Natural source code to a test or production environment. All functionality of cataloged objects (for example, scheduling and backout) is available for source code in production.

Object Lists

Expanding an Object List

The Expand function helps automate the creation of object lists. PAC "expands" objects in an object list by automatically adding specified sets of related objects to the list. Expand can produce the following sets:

- all objects needed to compile a specified object (for example, data areas, copy code, views/DDMs, and maps),
- all objects referenced by a specified object during execution (for example, programs, subroutines, and maps),
- all objects that use or reference the specified object.

Before an object list can be expanded, it must contain at least one object. An object list can be expanded manually or automatically:

When you invoke the Expand function from within the object-list editor, PAC expands each object in the object list.

You can use a selection list to add one or more objects to an object list and instruct PAC to expand the object(s) at the same time.

When defining a migration path or authorizing a migration event, you can instruct PAC to expand the object list automatically during the migration. In the authorization, you can specify whether to expand all the objects in the list or only selected objects.

Even without knowing the version numbers, you can control the versions of objects that PAC expansion adds to an object list. You can instruct PAC to use specific versions of objects, versions in a specified status, or the latest versions in the PAC-controlled environment.

Generating an Object List

The PAC Generate List facility generates an object list automatically from one of the following sources:

- PAC generate, objects selected from origin status, based on name and type
- a Predict set,
- a Predict Case set,

- the list of objects in a maintenance request,
- user-defined archive selection criteria (for archive events),
- input from a work file (batch only).

You can also use Generate List to align the application objects in two statuses. PAC compares the objects in the origin and destination statuses; it includes in the object list only the objects in the origin status that are not in the destination status or that have a different date-time stamp.

Reporting System

You can display and print reports of any PAC entity. Some of the information that can be displayed in PAC is listed below:

- Applications:
 - the application definition (including processing options, general defaults, and default libraries),
 - statuses to which a specified application is linked,
 - applications that have not been linked to any status.
- Statuses:
 - applications linked to a specified status,
 - statuses to which no applications have been linked.
- Application status links:
 - the Natural library and file assignment for the application at the status,
 - the Predict Case library and file assignment (if applicable),
 - the file translation table and step library assigned to the link,
 - defined destination statuses (migration paths) from the status,
 - object versions in the application library at the status,
 - Predict Xref data for the application at the status.
- Migration paths:
 - the configuration of migration paths for a specified application,
 - the authorizers and job for a specified migration path.
- Migration events:
 - the origin and destination statuses (migration path),
 - the Schedule Date,
 - the maintenance request (if applicable),

- the object list,
- authorization information,
- job information,
- the audit report of the processing of the event.
- Object versions:
 - control and directory information,
 - source code,
 - object versions used by the object at compile time or run time,
 - object versions that use the object at compile time or run time,
 - the status where the object version resides,
 - an object history (all PAC versions of each object for an application, including the statuses where they currently reside).
- File translation tables,
- Jobs (you can display the JCL/JCS),
- Maintenance requests,
- Change control logs.

For information about PAA reports, refer to *Managing and Auditing the Production Environment*.

PAC reports for applications, jobs, object versions, statuses, migration events, and maintenance requests can be customized for your site.

Administrator Functions

PAC provides administrators with facilities to perform the following control and maintenance functions:

- Add, modify, and delete user profiles, including authorization levels.
- Establish defaults for the PAC system, profiles, applymods, descriptions, applications, and Predict generation
- Activate and deactivate user exits.
- Release locked data.
- Reset migration events so that they can be rerun.
- Maintain tables of codes for maintenance requests.
- Foreign table maintenance.
- Maintain security definitions for views.
- Archive event maintenance.

- Purge change control logs, audit histories, and obsolete object versions. You can purge object versions with or without first archiving them.

Utilities

Compare Utility

The PAC Compare utility allows you to compare the following:

- The Natural directories for two objects or two versions of an object,
- Lists of saved (source) and cataloged (executable) objects in two libraries or in a library or PAC status,
- the source code for two objects or two versions of an object,
- the contents of two Natural libraries,
- foreign objects under the control of PAC,
- PAA controlled objects.
- In addition to the Compare, the results can be written away to a workfile, printed, saved to a Natural text member or manipulated in several different ways.

You can compare two PAC object versions, two non-PAC objects (objects in Natural libraries outside the PAC-controlled environment), or a PAC object with a non-PAC object. Compared objects can reside in different libraries.

To identify an object version for comparison, you can specify either the version number or the status in which it resides.

Migration Utilities

The PAC utilities MIGRATE LOAD (MIGLOAD) and MIGRATE UNLOAD (MIGUNLD) perform the same functions as the Natural utilities NATLOAD and NATUNLD but also create and use PAC control information. If you want to use only one set of utilities in a mixed environment, you can use MIGLOAD and MIGUNLD for non-PAC objects as well.

With the MIGUNLD utility, you can unload objects from a set created using Predict Xref data, objects from Natural libraries, and Natural views/DDMs from a Predict file. For each unloaded object, the MIGUNLD report lists the object name and type, version, destination library, origin database and file numbers, and other information.

With the MIGLOAD utility, you can load Natural saved objects, cataloged objects, views, and user error messages. A MIGLOAD report lists the name, type, and version of each object loaded; its destination library, database, and file; whether it replaced an existing object with the same name in the destination library; and other information.

You can execute MIGUNLD and MIGLOAD in batch or online, using menu selections or direct commands.

Scan Utilities

PAC includes scan utilities:

With SCANPATH, you can scan all migration paths for a specified job name. This utility includes options to replace the job name selectively or globally and to modify migration-path defaults.

With SCANOBJ, you can scan a PAC application's Natural objects for a character string. You can specify a range of objects or object types, a version number, a range of version numbers, or all versions in a particular status. The utility supports absolute scans.