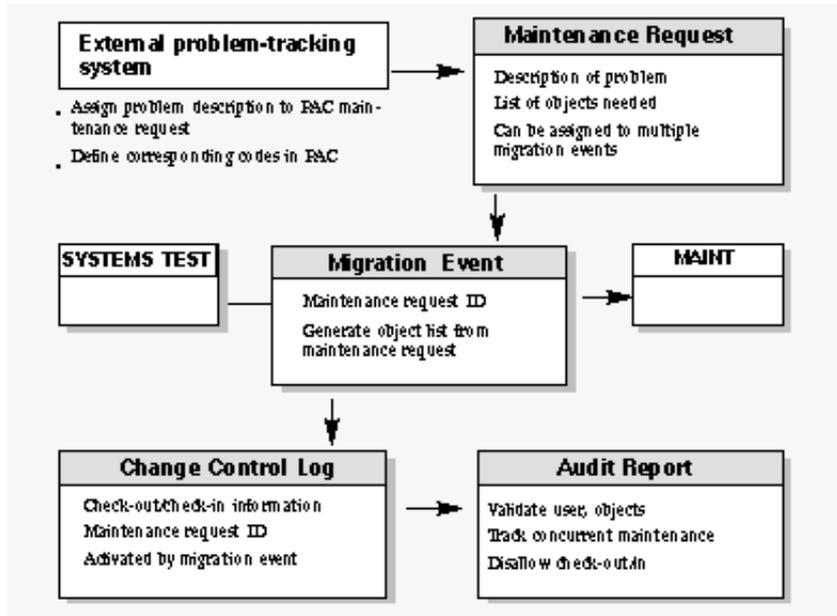


# Maintaining Natural Objects in PAC

PAC provides facilities for checking and maintaining the integrity of application object versions, including maintenance requests, check-out/check-in processes and change control logs. User exits and Application Program Interfaces (APIs) allow you to customize these facilities to meet your site requirements.

The diagram below shows the relationships among PAC migrations to and from maintenance, change control logs and audits, and maintenance requests.



This chapter covers the following topics:

- Maintenance Requests
- Change Control Logs

## Maintenance Requests



When maintenance is required for an application, a maintenance request can be created to document problems and maintenance activities. It allows you to identify to PAC certain problems, change enhancements, or topical information associated with a list of programming objects that are to be used to solve a problem or implement a relevant function. As a user, you can maintain the information returned by the maintenance request.

In addition to describing the problem and related activities, a maintenance request can list all objects required to resolve the problem; PAC can then generate an object list automatically from the list in the maintenance request.

Activities performed for a given maintenance request, as well as all objects participating in the associated migrations, are linked and can be tracked.

This maintenance request may span applications, a capability that is important when generating migration events.

When you assign a maintenance request to an event migrating an object to or from maintenance, PAC automatically assigns the request to a change control log. You can select and view change control logs related to the maintenance request.

As related problems emerge, they can be added to a request. Related problems can be grouped in a single maintenance request which can be assigned to multiple migrations. You can view all migration events related to a problem or assign them to a group migration event.

To document complex problems and activities, you can also group requests, each of which may be assigned to multiple migrations.

Once a maintenance request is defined to PAC, you may invoke additional user-defined maintenance request facilities using the "Extended User Info" facility of the maintenance requests Additional Options.

You can define and assign site-specific codes to indicate the state of a maintenance request (for example, open or closed) and the action to be taken in response. You can also prioritize requests. Using user exits, you can establish your own rules and standards. Application Program Interfaces (APIs) enable you to call PAC directly from outside PAC to perform maintenance request activities.

If a maintenance request is defined for the migration event, then the objects are also added to the maintenance request Related Object Versions list.

## **Interfacing with External Problem-Tracking Systems**

Although not a problem-tracking system itself, PAC does provide some basic problem-tracking facilities and may be used to extend the facilities of existing external problem-tracking systems.

You can integrate PAC with an external problem-tracking system through the Maintenance Requests sub-function.

1. Define the PAC maintenance request ID to match the problem ID in the external problem-tracking system.
2. Define the maintenance request so that the identifier of the problem in the external problem-tracking system is synonymous with the maintenance request in PAC.
3. Assign the problem description in the external system as an attribute to the PAC maintenance request.
4. Define the codes for PAC problem states, priorities, and actions to correspond with codes in the external system.
5. Record the description of a problem in a problem-tracking system and use Application Program Interfaces (APIs) to reflect or copy these external problem descriptions into PAC.
6. Use provided user exits to derive the maintenance request data from, or keep the maintenance request data in synchronization with, the data in the external problem-tracking system.

## **Assigning Maintenance Requests to Migration Events**

Once a maintenance request is defined, you may assign it to a migration event to enhance check-out/check-in tracking and allow you to review all events, created or completed, for a particular request.

The following example illustrates how a maintenance request is used to enhance tracking of object maintenance activities:

**Seq Action**

- 
1. User A discovers a problem in production.
  2. User A identifies the object causing the problem.
  3. User A opens a maintenance request (RQ 00001) for the problem.
  4. Programmer B creates a migration event (production to maintenance) ORD-PM-0001 with a maintenance request of RQ 00001.
  5. Programmer B starts making modifications to resolve the problem. Subsequently, User A discovers the need to maintain other objects and creates another migration event (Control to maintenance) ORD-CM-0007 with the same maintenance request of RQ 00001.
  6. When the updates are completed, Programmer B generates an event (maintenance to Control) ORD-MC-0011 to migrate the new objects into PAC.
  7. Maintenance request RQ 00001 is again specified.
  8. Programmer B now creates an event (Control to integration-test) ORD-CTI-0036 with maintenance request RQ 00001 to test the new changes.
  9. At this point, the project leader may access maintenance request RQ\_00001 to review its status; and then may review all events completed as part of the change. The list of events will be as follows: ORD-CM-0007 ORD-CTI-0036 ORD-MC-0011 ORD-PM-0001

**Table Maintenance**

When maintenance requests are defined to PAC, Status tables and Action tables are used to determine the sequence in which maintenance requests are retrieved for selective processing.

These tables must be defined before you can enter maintenance requests. Status and Action information codes are required input when you add maintenance requests to PAC, and the values provided are validated against the Status and Action tables defined at your site.

Tables are defined by the PAC administrator who uses the table maintenance facility to set up table entries and perform maintenance functions for maintenance requests. Tables should be defined so that they extend the integration of PAC maintenance requests with the external problem-tracking system currently used at your site.

Depending on the table type (Status or Action), the PAC administrator specifies the possible codes along with descriptive text for each as table entries.

**Status Table Examples****Code Status Text**


---

OPE	Open
INI	Initial, new problem
CLO	Closed
INF	Inform user
WER	Waiting for additional error information
INC	Incomplete information given
QUE	Question

**Note:**

The only Status code that cannot be deleted is CLO (Closed). A maintenance request with the Status of Closed can no longer be modified.

**Action Table Examples****Code    Action Text**

---

SRC    Source change  
VER    Next version  
DOC    Documentation change  
MSC    Miscellaneous  
HDW    Hardware problem  
SFT    Software problem

**Maintenance Request Reporting Facilities**

The PAC administrator also establishes a mechanism to prioritize the sequence in which the list of maintenance requests is retrieved. Table entries must have an associated sequence number. This number along with the maintenance request priority determines the sequence in which maintenance requests are displayed with the Selection option.

To facilitate reporting, PAC uses a grouping facility. The grouping hierarchy is as follows:

Status    Status of the maintenance request;  
Priority    Priority assigned to the problem;  
Action    The action or reason code to further define the status;  
Request    The name of the maintenance request. This may be a numeric and/or alphabetic value.

Also, the list of maintenance requests may be retrieved in alphabetical order.

**User Exits**

You may use PAC user exits for interfacing with external problem-tracking systems, rules, security, extended processing, and customization.

## Exit Description

---

- 19 User Exit 19 maintenance request Data Manipulation; invoked during the processing (adding, modifying, displaying) of the maintenance request data. PAC uses this user exit to pass the data and/or allow the user to update the PAC information.
- 20 User Exit 4 maintenance request User Exit 20 maintenance request Invoking Maintenance Requests; invoked whenever any function (except Purge, which may be handled with User Exit 4) for maintenance requests is invoked. User Exit 20 can be used to meet security requirements and validate and/or override the maintenance request ID. (The maintenance request ID may be updated using this exit).
- 21 User Exit 21 maintenance request Linking to Events; allows you to validate the use of maintenance requests specified with migration events. For example, the exit may be set up to ensure that a maintenance request is always specified for a migration event where the origin or destination status is a maintenance status type. This is important for tracking the actual state of the maintenance request and the progression of objects through the defined application life-cycle.

Refer to the PAC Reference documentation for detailed information about PAC user exits.

## Application Program Interface (APIs)

The Maintenance Request API (APINMREQ) allows users to add, modify, purge or display (retrieve) information about a particular maintenance request.

A list of current APIs is provided in the PAC Reference documentation.

## Change Control Logs



The PAC check-out/check-in facility is activated automatically by a migration to or from a maintenance status. The facility tracks and documents the actions taken to maintain Natural objects under the control of PAC.

### Note:

Check-out/check-in facilities are not currently available for foreign objects.

When an event migrating an object to a maintenance status is processed, PAC creates a change control log for the check-out, which records the

- application name;
- object name and version;
- user ID of the user checking it out;
- terminal ID or batch ID;
- date and time of the check-out;
- library and user system file to which the object is migrated;
- maintenance request ID (if applicable); and
- optional notes by the user.

When the object version is checked back in from the maintenance status, PAC updates the log with the new version number created as a result of the maintenance.

In conjunction with a maintenance request, the check-out/check-in process may be used to maintain a list of all objects required to satisfy a specific maintenance request.

Although users may check out the same version of an object, each user has a unique copy of that object.

PAC audits check-out and check-in activities. A user exit lets you view the audit information, validate the user and object, and disallow the check-out or check-in. The user exit also lets you track and control concurrent maintenance activities. When a user checks an object out, you can list other users who currently have the same object checked out. When a user checks an object back in, you can list other users who still have the object checked out, or who checked the object in after this user checked it out.

## Check-Out Processing

When you set up the initial migration event to check an object out for maintenance, you may optionally specify an associated maintenance request ID.

The processing of the migration event results in the following:

- The specified version of each object is migrated to the location of the maintenance status.
- A change control log is created for each object.
- An object status entry is created for each object indicating that it has been migrated to a maintenance status.

If PAC detects that any object being checked out is already checked out to a maintenance status or that its most recent migration was to a development status, PAC writes a warning about the concurrent maintenance to the audit report. If User Exit 7 has been implemented, it may be used to permit or disallow the concurrent check-out of the object in question. The exit may also be used to write additional user information to the audit report.

## Check-in Processing

After maintenance has been performed on an object, the object must reenter PAC before the change can be recorded. A migration event must be set up to return the object to the PAC controlled environment; the migration event should specify

- the maintenance origin status where the object is being maintained;
- the destination status, which must be Control, or a test or production status type; and
- the maintenance request ID (recommended but optional).

As a result of the migration, the object is moved or copied from the maintenance status into the destination status and implicitly to Control status if the destination status is other than Control. The link of the object version checked out to the maintenance status is purged.

During the check-in process, PAC locates the corresponding change control log (check-out record), updates it with the check-in information, and closes it. If there is no corresponding open change control log for an object (for example, a new object was created as the result of the maintenance process), a change control log is created and will contain only the check-in information.

## Move Option

The Move option results in the object being compiled and assigned a new version number. The new version of the object is migrated to the destination status.

By default, the Move option is in effect for all migration paths where the origin status is a maintenance status type. This means that when objects are migrated from maintenance back into PAC (that is, checked in), both the saved and cataloged objects are physically purged from the maintenance library.

If Applymod 24 is active, only the saved (source) object is purged from the maintenance library. The cataloged object is not purged. This helps to protect the integrity of the environment for any subsequent testing of other maintenance changes.

## Copy Option

If the Copy option is used, the object is recompiled in PAC and assigned a new version number. The new version of the object must then be copied to the maintenance library.

A new change control log recording the check-out of the new version of the object is created, and the new version of the object is placed in the maintenance status from which the previous version of the object was migrated. The new change control log is created to maintain consistency between what is in PAC and what is in the maintenance environment.

The following example illustrates the steps used when the Copy option is selected for migrating objects from maintenance:

1. APGMP1.0003 is checked out to the status MAINTENANCE (status type M).
  - An object-status link is created for APGMP1.0003-MAINTENANCE.
  - A change control log is created for APGMP1.0003 and the check-out information is recorded in the log.
2. APGMP1.0003 is modified and saved; the saved date-time stamp information is also updated.
3. APGMP1.0003 is checked in from the status MAINTENANCE and assigned a new version of 0007 and recompiled in PAC.
  - The object-status link for APGMP1.0003-MAINTENANCE is purged.
  - The change control log that corresponds to APGMP1.0003 is located and updated with the check-in information for APGMP1.0007.

Thus, the change control log documents that APGMP1 was checked out to maintenance as Version 3, modified, then checked back into PAC as Version 7. Version 3 is the maintenance version of Version 7.

4. To maintain consistency between what is in PAC and what is in the maintenance environment:
  - An object-status link for APGMP1.0007-MAINTENANCE is created.
  - A new change control log with associated check-out information is created for APGMP1.0007.
  - APGMP1.0007 is copied to the maintenance status specified in the migration path.

## Canceling Maintenance

If an object version has been checked out to maintenance, and it is later decided that the object requires no modification, a migration event may be set up to cancel the maintenance for the object. The migration event specifies

- the maintenance origin status where the object is being maintained;
- the destination status, which must be a retire status type; and
- the maintenance request ID (recommended but optional).

As a result of the migration, PAC locates the corresponding change control log with the check-out information; updates it to indicate that the maintenance was cancelled; and purges the object from the maintenance status location.

## Selecting Objects With User Exit 7

User Exit 7 is invoked during the check-out/in of objects to/from a maintenance status type. As each object is processed, you can allow or disallow the migration of that object. A nonzero response code returned from the exit instructs PAC to reject a specific object for processing.

### Note:

Since PAC currently does not support message switching facilities, it is recommended that users integrate their own message switching facilities. For example, use Connect from User Exit 7 in the check-out/check-in process of PAC so that all users are correctly informed about check-out/check-in activities.

Refer to the PAC Reference documentation for more information about User Exit 7.

When PAC User Exit 7) is active during check-out of an object, you may

- validate the user and the object being checked out;
- display information about the object being checked out;
- obtain a list of other users who also may have previously checked out the same object (concurrent maintenance); and
- disallow the check-out of the object.

When a same or different version of an object is checked out by another user, User Exit 7 may be used to notify that user that another version of the object may be undergoing modification in another library. It is the user's responsibility to check the validity of the notification.

If PAC User Exit 7 is active during check-in of an object, you may

- validate the user and the object being checked in;
- display information (for example, terminal, library, maintenance request) about the object being checked in;
- obtain a list of all users who
  - have the same object still checked out;
  - checked in the same object subsequent to the check-out date of the object.
- disallow the check-in of the object.