

# Migrations from Control to Control : Alignment

Alignment is a special migration that forces Natural objects already under the control of PAC to be recompiled in the Control status with the current version or a specific version of all referenced subordinate objects. Alignment is used to ensure that all objects use the same version of a subordinate object at compile time. Alignment occurs when the Control status is specified as both the origin status and the destination status; that is, Control to Control.

Subordinate objects are those objects "used" during the compile of another object. Subordinate objects include maps, copycode, data areas, rules, and views.

By default, PAC uses the current versions of subordinate objects to compile new objects. However, because PAC represents applications in different phases of their life-cycles, it may be necessary to recompile objects already under PAC control with current versions of subordinate objects.

For example, if a global data area (GDA) is changed and migrated into PAC, all objects that use the GDA will need to be recompiled. If you use the alignment facility, there is no need to check the objects out to maintenance first.

At compile time, PAC recatalogs each object in the object list with the most recent version of Control's subordinate object, or with the version specified by the version number or status referenced in the object list (see the information about "rolling"). The newly compiled object versions become the current versions.

If a steplib is needed for the compile, this should be specified on the Control application status link.

Alignment migrations can be run online or in batch; a work file is not required.

The MIGRATE job is used to perform the alignment of object versions.

This chapter covers the following topics:

- Creating the Object List
- 

## Creating the Object List

When creating the object list for an alignment migration event, subordinates that are not to be recompiled must have a "reference" (version or status) attached to their object list entries. If a subordinate has a reference, then rolling occurs:

- the object is not recompiled with the current version of the subordinate objects; and
- a new version is not created for it.

If the subordinate has no reference, then it is recompiled.

The Expand option can be used to help ensure that all necessary objects are included in the object list.

### **Note:**

If status or version references are not specified, the current versions for all objects in the list are used, including any objects added as a result of expand processing.