Registration

If a class is to be made accessible to DCOM clients, it is necessary to add some information about the class to the system registry. DCOM clients will mostly address a class with a meaningful name, the so-called programmatic identifier (ProgID) as in the following example:

```
CREATE OBJECT #O1 OF CLASS "Employee"
```

For a Natural class, the class name defined in the `DEFINE CLASS` statement is written into the registry as a ProgID.

System registry entries map this ProgID to the globally unique ID (GUID) of the class, allowing DCOM to uniquely locate all information about the class. Further information that is stored in the registry includes the path and name of the responsible DCOM server, the path and name of the type library, and interface information.

This chapter covers the following topics:

- Registration with Natural
- Automatic Registration
- Manual Registration
- Registration Files and Type Library
- Client Registration
- Registration Hints

Registration with Natural

Natural classes can be registered (or unregistered) manually with the system command `REGISTER` (or `UNREGISTER`), automatically after the class is stowed (or deleted), or by running the .reg files, which are generated every time a class is registered.

In order to register classes, you must have the rights to modify the system registry and your system environment must be able to use COM.

It is usually not advisable to change the Natural entries in the system registry directly in the registry editor because this can lead to inconsistent registry entries.

A class is always registered for the server ID under which Natural was started.

Automatic Registration

If the profile parameter `AUTOREGISTER` is set to "ON", a Natural class is automatically registered when it is stowed (cataloged), and unregistered automatically when it is deleted. This means that the user can test the class directly after stowing it.

Automatic registration uses the activation policy setting defined in the `WITH ACTIVATION POLICY` clause of the `DEFINE CLASS` statement of the class. If this clause is not specified, the setting from the profile parameter `ACTPOLICY` is used.

If automatic registration is set and a class is stowed (cataloged), the class is unregistered before it is stowed and registered after the stow has finished so that all old registry entries are removed.

Manual Registration

The following topics are covered below:

- The REGISTER Command
- The UNREGISTER Command

The REGISTER Command

The system command `REGISTER` is used to register Natural classes. They are registered for the server ID under which Natural was started.

```
REGISTER { class-module-name }  [ { library-name } [ { ES } ] ]
         {                   }  [ {              } [ { IM } ] ]
         { *                 }  [ { *            } [ { EM } ] ]
```

class-module-name

This defines which class or classes are to be registered by specifying the appropriate Natural object module name.

library-name

This defines which library or libraries are to be searched for the class or classes.

ES, IM or EM

This defines the activation policy, which is registered for the class or classes.

You can set one of the following parameters:

Parameter	Description
`ES`	Sets activation policy "ExternalSingle".
`IM`	Sets activation policy "InternalMultiple".
`EM`	Sets activation policy "ExternalMultiple".

The following table shows which classes will be registered for all possible class/library combinations:

Class Module Name Specification	**Library Name Specification**		
library-name	*	-	
class-module-name	class with class module name *class-module-name* of library *library-name*	all classes with the class module name *class-module-name* which are found in the current step libraries	class with class module name *class-module-name*
*	all classes which are found in the library *library-name* are registered	all classes which are found in the current step libraries are registered	all classes of the current logon library are registered

If this parameter is not specified in the `REGISTER` command or the `DEFINE CLASS` statement, the default activation policy defined in the parameter file is used.

The UNREGISTER Command

The system command `UNREGISTER` is used to unregister Natural classes.

```
UNREGISTER { class-module-name }   [ { library-name } [server-id] ]
           { *                 }   [ { *            }             ]
```

class-module-name

This defines which class or classes are to be unregistered by specifying the appropriate Natural object module name.

library-name

This defines the library or libraries which are to be searched for the class or classes.

server-ID

This defines the server ID of the class or classes.

The following table shows which classes will be unregistered for all possible class/library/server ID combinations:

Class Name Specification	**Library Name /Server ID Combination**				
	--	***library-name -***	***library-name server-ID***	**** -***	****server-ID***
class-module-name	class with *class-module-name* in the current logon library if it is registered for the current server ID	class with *class-module-name* of library *library-name* if it is registered for the current server ID	class with *class-module-name* of library *library-name* if it is registered for the server *server-ID*	all classes with *class-module-name* found in the current step libraries if they are registered for the current server ID	all classes with the name *class-module-name* found in the current step libraries which are registered for the server *server-ID*
*	all classes of the current logon library which are registered for the current server ID	all classes found in the library *library-name* which are registered for the current server ID	all classes found in the library *library-name* which are registered for the server *server-ID*	all classes found in the current step libraries which are registered for the current server ID	all classes found in the current step libraries which are registered for the server *server-ID*

A `REGISTER` or `UNREGISTER` system command will return an error message if *class-module-name* or *class-module-name* and *library-name* are specified but either the class or library is not found. If only an asterisk (*) is given in the `REGISTER` or `UNREGISTER` system command, no error message is returned if no class has been registered or unregistered.

If a class without class GUIDs or interface GUIDs is specified in the `REGISTER` system command, an error message will be returned. Such a class can only be used in the local Natural session.

Note:
Under Natural Security, this command can only be called for a single library. This means the library name has either to be omitted or a specific library has to be used. It is not possible to use an asterisk (*).

Registration Files and Type Library

Registration files (".reg" files) enter information in the system registry when they are executed.

Natural will automatically create registration files for the server and the client side when a class is registered.

The server ".reg" file contains the same information that was entered in the system registry and the client ".reg" file contains all information, which is generated for the client side. When a class is unregistered, the .reg files will be deleted. If a ".reg" file is not to be deleted with the unregistration, the file has to be renamed before unregistering the class because Natural deletes only files with the default ".reg" file names.

The ".reg" files will be named *classmodule_name_S.reg* (for the server) and *classmodule_name_C.reg* (for the client) and, to activate a different version, *classmodule_name_V.reg*.

A type library is created automatically when a class is registered, and it is deleted when a class is unregistered. A reference to the type library is also entered in the registry.

The default type library name is *classmodule_name.tlb*. A new name will be generated if a type library with this name exists already.

The registration files and the type library are stored in the Natural *etc* directory as follows:

```
$NATDIR/$NATVERS/etc/serverid/classname/v<version-number>
```

Example

The files for version one of a class `MY.TEST.CLASS` registered for the server ID "SERVER01" are located as follows:

```
$NATDIR/$NATVERS/etc/SERVER01/MY.TEST.CLASS/v1
```

Client Registration

Natural does not enter the registration information for the clients automatically in the system registry, but creates a registration file for the client. The client registration file contains an entry (`RemoteServerName`) that tells DCOM on which machine the DCOM server class can be found. This entry is not filled from Natural. It can be entered in either of two ways:

1. The `RemoteServerName` can be entered in the registration files. In this case the line

   ```
   "RemoteServerName"=
   ```

   has to be changed to

   ```
   "RemoteServerName"="server_machine_name"
   ```

   After this, the registration file has to be executed on the client machine.

2. The registration file is executed first, and then the `RemoteServerName` is changed using the `DCOMCNFG` tool or the Registry Editor (see the section *DCOM Configuration on Windows*).

Registration Hints

The following points should be taken into account when registering and unregistering classes:

- The class GUID should never be changed for an existing class: Natural displays an error message if a class that is already found in the registry is registered again with another GUID. The old class must first be unregistered in this case.

- The same class should never be registered for more than one server ID: there is a one-to-one relationship between the server ID and the AppID, and a class has only one AppID defined, which means that a registration for a second server ID overwrites the AppID. Furthermore, if the class is unregistered for one server ID, all entries of the class are removed without checking whether it is registered for a second server.

- Except for client registration, you should always use the Natural system commands `REGISTER` and `UNREGISTER` to change registry entries for a class because they remove redundant registry entries.

  For example, if a client class has been registered for "server1" and a server registration file with a registration of the same class for "server2" is run, the AppID key of the class is changed and all references to the old AppID key are lost. So this old AppID key can never be deleted. When a class is registered with the system command `REGISTER`, a check is made to see whether the AppID has been changed, and the old AppID is removed if no other class needs it.

- If Natural is not available on the client machine and registry entries for a Natural class are to be removed from the system registry, you should do this with the registry editor. If Natural is available on the client machine, it is easier to register the class first with the Natural system command `REGISTER` and unregister it afterwards with the system command `UNREGISTER`.

- The registration information for a class is taken from the catalogued class object, so that it is not possible to register or unregister a class that is only available in source format.

- If you want to register classes during a Natural session, the session must be started with the parameters `PARM` and `COMSERVERID` only as shown below. This is because only these two parameters are stored in the registry key "LocalServer32". If a class is tested with other parameter settings, there is no guarantee that it will run later when it is started from a DCOM client.

```
NATURAL.EXE PARM=COMPARM COMSERVERID=SERVER1
```

- Usually only users with administrator rights can change the system registry. So if you receive an error when trying to register a class, check to see whether you have the rights required to change the registry.

- When a Natural class is registered, some additional information is entered in the registry that is only needed by Natural (not by DCOM). The information which is stored in the additional registry keys is the server ID (see section *NaturalX Servers*), the activation policy (see section *Activation Policies*) and the location (Natural class module name and library of class) of the class. This information is necessary, for example, if all classes of a specified server ID are to be unregistered or to make the served classes available when Natural is started.

- There is a one-to-one relationship between the server ID and the AppID (under *HKEY_CLASSES_ROOT/AppID*) of a class. When a class is registered for a new server ID, a new GUID - the AppID, is generated and assigned to this server ID. The AppID is used by DCOM to group the DCOM classes. Security settings and (for client registrations) the remote machine name are defined for an AppID, i.e. all classes, which belong to one AppID, have the same security settings (see the sections *Configuration Overview* and *Security with NaturalX*).