The omniNames CORBA Naming Service

omniORB 4.3

1 Introduction

omniNames is omniORB's implementation of the OMG's COS Naming Service Specification. It offers a way for a client to turn a human-readable name into an object reference, on which the client can subsequently invoke operations in the normal way. See the OMG specification for full details of the functionality provided by the Naming Service.

The Naming Service stores a set of bindings of names to objects. These bindings can be arranged as an arbitrary directed graph, although they are often arranged in a tree hierarchy. Each node in the graph is a *naming context*. There is a 'root' context at which name lookups usually start. This is the object returned by the call to `CORBA::ORB::resolve_initial_references("NameService")`.

2 Data file

omniNames persists its contents in a data file. The data file takes the form of a 'redo log', to which it writes out an entry every time a change is made to the naming graph. The directory in which this log file is written can be specified with the `-datadir` command line option or the `OMNINAMES_DATADIR` environment variable. When omniNames is restarted it uses the data file so that it can regenerate the naming graph.

Periodically the data file is checkpointed, removing unnecessary entries from the file. The idle time between checkpoints defaults to 15 minutes. That can be overridden by specifying a period in seconds in the `OMNINAMES_ITBC` environment variable.

The name of the data file name normally includes the hostname, in the form `omninames-`*hostname*`.dat`. To suppress inclusion of the hostname, start omniNames with the `-nohostname` option.

Previous versions of omniNames referred to the data file as a 'log file', and the file had the extension '`.log`'. When omniNames starts up, it will accept a data file with the extension '`.log`' and replace it with one with the extension '`.dat`'.

3 Starting omniNames and configuring omniORB

When starting omniNames for the first time, you can either let it start with the default TCP port of 2809, or you can specify the port number on which it should listen. This is written to the data file so that on subsequent invocations it will listen on the same port number and thus can be accessible through the same object reference. When omniNames starts up successfully it writes the stringified object reference for its root context on standard error.

At startup, omniORB reads its configuration from `omniORB.cfg` or from the Windows registry. Amongst other settings, the configuration contains the object reference to the root context of the Naming Service. This object reference is returned by `resolve_initial_references("NameService")`. There are a number of methods of specifying the root naming context—see the omniORB manual for details.

4 Command line parameters

omniNames accepts the following command line parameters.

`-help`	Output usage information.
`-start` *[port]*	Start omniNames for the first time, listening on *port*.
`-always`	In conjunction with `-start`, always start up omniNames, even if its data file already exists.
`-datadir` *directory*	Specifies the directory for the data file, overriding the default.
`-logdir` *directory*	Equivalent to `-datadir`, for compatibility with previous omniNames versions.
`-errlog` *filename*	Causes trace output that would normally be sent to stderr to go to the specified file instead.
`-nohostname`	Normally, omniNames includes the host name in the name of the data file. This option disables that, meaning the data file can be used on a different host, or if the host name changes.
`-install` *[port]*	On Windows, install omniNames service. See below.
`-manual`	On Windows, specify that the service should be configured to require manual start. See below.
`-remove`	On Windows, uninstall omniNames service. See below.

5 Machines with multiple IP addresses

The CORBA naming service is a tree (or graph) of `NamingContext` CORBA objects. For each of those CORBA objects, the object reference contains details about the *endpoint*—i.e. the host address and port—that is used to contact the object.

When the machine running omniNames has multiple IP addresses, omniORB listens for incoming connections on all addresses, but it arbitrarily picks one of the addresses to publish in object references. It might pick the 'wrong' one, meaning that clients connect to the configured root context successfully, but then fail to connect to the sub-contexts. To force omniORB to publish the correct IP address, use the `-ORBendPointPublish` command line parameter:

```
-ORBendPointPublish giop:tcp:address:
```

where *address* can be an IP address or a host name.

Occasionally, it is necessary to limit omniNames to listen on just one of a machine's IP addresses. To do that, use `-ORBendPoint` to specify the address to listen on. In this case, you should also specify a port (e.g. the standard 2809), otherwise an arbitrary ephemeral port will be used for the endpoint:

```
-ORBendPoint giop:tcp:address:2809
```

6 Windows service

omniNames can be run as a Windows service. To install the service, run with the `-install` command line option, with a port to override the default 2809 if necessary. The command line should include any other parameters of relevance, such as the data directory and error log file. It is particularly important to specify an error log file, since the service runs in an environment where stderr goes nowhere.

The service is normally configured to start at system startup time. Specifying `-manual` configures it for manual startup. Regardless of that setting, the service is not automatically started at the time it is installed. It should be started manually with the service control manager if it is to run before the next system restart.

Once installed, the service can be uninstalled by running omniNames with the `-remove` option.