User/Maintenance Documentation

CORBA Services

This document covers the small CORBA Services application, which helps to setup and run the i-scream server in a CORBA environment.

Revision History

29/03/01	Initial creation				
		Committed by:	tdb1	Verified by:	ajm4
				Date:	29/03/01
		Committed by:		Verified by:	
				Date:	
		Committed by:		Verified by:	
				Date:	
		Committed by:		Verified by:	
				Date:	
		Committed by:		Verified by:	
				Date:	

Introduction

The whole of the i-scream server architecture is built around the CORBA system. Running CORBA applications isn't complicated, but they all bind together using a Naming Service. This service is usually setup independently of the CORBA application, in this case the i-scream server, but will require a webserver to place reference files on.

The CORBA Services program aims to wrap this up neatly into one program, making it easy for a first time CORBA user to get going with the i-scream server.

What is CORBA Services?

CORBA Services contains two sections. Firstly it manages configuring and running the CORBA naming service (in this case, JacORB's naming service) along with any other services you may which to run. It does this in an extremely simple manner; you just list those you wish to run and they're started up for you.

Secondly CORBA Services manages serving of the CORBA initial references, a job which would usually require a separate webserver. CORBA Services automatically starts up a basic webserver which will serve the required references to any requesting applications. It has some basic security measures which should restrict the list of files it makes available.

All that is therefore required to start this up is a quick configuration and then start. The default configuration may be fine in most cases, which makes the task even quicker. One can then proceed to configure and start the i-scream server.

Installing and using CORBA Services

CORBA Services is written in Java, and should therefore work on any platform that has Java 1.2+ support. It is entirely self contained, and should therefore run as is when downloaded.

Downloading and installing

The first stage is to get hold of the last build of the CORBA Services program from the i-scream website. It may be downloaded from the *Builds* section found here: -

http://www.i-scream.org.uk/builds/

Once downloaded, the archive should be extracted to a location of choice, maintaining the directory structure. The `lib` directory contains the required JacORB libraries, and the `web` directory is the "web visible" folder. Configuration, which we will look at next, is kept in the `etc` directory.

CORBA Services is now ready to be configured and run.

Configuring

The first stage is to setup the services that need to be run. This is held in the services configuration file, `etc/services.conf`, and as default contains the following line;

```
jacorb.naming.NameServer ./web/NS_Ref
```

This line tells CORBA Services to start the JacORB name server, with the parameter given. In this case, it tells the name server to place it's reference file in the `web` directory as `NS_Ref`. Further services may be added in a similar manner, please consult the external JacORB documentation[1] for details on what they are and what parameters they need. Please note that any files, such as references, that need to be available to CORBA applications must be placed in the `web` directory.

The next step is to setup the security features. This is a very basic affair, and just requires editing the list of allow URLs in the configuration file `etc/okUrls.conf`. This file as default contains the following line;

```
/NS_Ref
```

This permits only requests that look like this to be served;

```
http://corba.services.host.name:port/NS_Ref.
```

If further services are added, their references files should also be added to this list. If an URL is requested that is not in this list, CORBA Services will attempt to return a 403 (Access Denied) error to the caller. However, if the file is listed but simply cannot be found, a 404 (Not Found) error will be sent instead.

It is feasible to have any text based files served from this directory, even including HTML. However the following should be noted.

- Binary files will almost definitely not work
- Sub-directories of the web directory have not been properly tested, although they should work as expected.
- The built-in webserver is unlikely to conform to any standards laid out for a webserver to conform to, and may not respond correctly to some requests. It has only been tested in a very basic manner, although more thoroughly with CORBA applications.

[1] http://www.jacorb.org/

- It is not recommended this webserver be used for anything other than serving CORBA references, you do so at your own risk.

Configuration of CORBA Services is dynamic. This means you can change the configuration on the fly, and it will take immediate effect. There is therefore no need to shutdown and restart the program to activate a change.

The final stage is to get the CORBA Services program up and running.

Running

It is recommended that CORBA Services be run on a terminal, or maybe under a program such as the unix screen. This allows viewing of any debugging output.

Starting up CORBA Services is just a case of running the required run script. However, if this fails to work, the following command can be executed.

```
java -jar iscream-corbaservices.jar
```

If this too fails to work, it could be a java problem. This guide provides no further help in troubleshooting CORBA Services.

When the program starts up something similar to the following should be shown;

```
-----------------------------------------
---  i-scream CORBA Services Manager  ---
---    © 2001 The i-scream Project    ---
---    (http://www.i-scream.org.uk)   ---
-----------------------------------------
---          Starting System          ---
---       Reading Configuration       ---
---      Starting CORBA Services      ---
jacorb.naming.NameServer ./web/NS_Ref
---      Starting Mini WebServer      ---
Host:           killigrew.ukc.ac.uk
Port:           8080
URL List File:  ./etc/okUrls.conf
Web Directory:  ./web
---   i-scream CORBA Services Ready   ---
```

The output here should correspond with what we have already seen in the configuration section. The first few lines are comments, followed on line 9 by the list of JacORB services we are actually starting. The next and last section list details of the webserver, including the machine name and port on which it is running. The last two lines should correspond with that in your configuration.

CORBA Services is now running, and ready to serve requests. This can be verified with your web browser by looking at a reference file URL.

At times you may see the following message in the console;

```
ServerRequest: Error replying to request!
```

This message simply means there was a problem communicating with the client program, whatever that may be. This is not a problem for CORBA Services, it is designed to deal with these errors and carry on serving.

CORBA Services is a multithreaded application in that it can serve many requests at once. It hasn't, however, been tested under a very high load.

Looking at how CORBA Services works

Overview

CORBA Services is broken into two distinct areas. Firstly there is the section that starts up the services, giving them the appropriate parameters. Then there is the webserver section, which serves the references dumped out by the services started in the first section.

A high amount of reflection is used in the first section, which may appear confusing to someone not experienced with it.

Analysis

This is not a substitute for looking at the code and reading the javadoc comments. It is meant as a supplementary guide as to how the program hangs together.

Starting Services

This section is performed only once at startup, and as such is built into the main method in the CorbaServices class. This section of the program deals with reading in the list of services to start, and then starting them up. This is not a trivial process, as can be seen by looking at the code.

The program starts off by reading in the services from the configuration file. It then parses these into arrays of classes to load and associated parameters.

The next step is to loop round starting up all of these classes. This is done by constructing the class, then constructing an object containing the arguments for the class. Next we get a hook on the main method of the class, which we will use to start it with.

With these bits of information in hand, we create a new ServiceThread object giving it the required objects we created before. Note that ServiceThread is an inner class of the main CorbaServices class. Finally we start the ServiceThread up, which has the effect of starting the JacORB service in question with the given parameters, just as you would with any main method at the command line.

Then, we start up the MiniWebServer which begins serving requests.

Serving Requests

This section is pretty straightforward. A request for a file is received and we send the contents of the file back. All that's extra to that idea is a list of allowed URLs. Lets look in a bit more detail at what happens.

As requests arrive a Handler thread is spawned off to deal with them. This inner class is responsible for dealing with the client for it's entire connection to the server.

The Handler thread strips down the request to get the actual URL. Once it has this it can run a check to see if the URL is permitted in the list of allowed URLs. If it is permitted it reads in the file and sends the contents back to the client. If for any reason it can't read the file, it returns the standard 404 (Not Found) error message. If the URL is not allowed, the client gets refused with a 403 (Access Denied) error message.

The connection to the client is then complete, and the thread ends.

It should be noted that the checking of URLs happens on the fly, and as such the configuration file can be changed with immediate effect. This means the CORBA Services program need not be restarted after a configuration change.

The MiniWebServer class is designed to be semi-robust. It does not guarantee to complete every service request, but it does make a good effort to continue providing it's service. It does

this by always looping round and restarting when something goes wrong, rather than just bailing out. This may mean the odd failed request, but the service should remain active.

From a statistical point of view, this program has not yet crashed or failed to do as expected when in use under our testing environment. This does not make it error free, but certainly says something about its stability.

Further information

Further information is available in our other documentation; the latest versions of which may be found online at the project website. Thank you for using i-scream products.

http://www.i-scream.org.uk