

CORBA with SANKHYA Varadhi - A Kickstart Guide

Introduction

The document intends to introduce the various CORBA features and Varadhi functionalities through sample applications that are shipped with SANKHYA Varadhi. It also provides useful information on each of these samples so that the user can get an overview of the powerful features provided by Varadhi.

Varadhi Samples

Varadhi provides several ready to use samples that demonstrate the use of standard CORBA features and Varadhi extensions. The samples contain one or more programs which together provide client/server functionality of a distributed application using Varadhi.

Varadhi samples can be located as follows

On Unix csh

`$VARADHI/samples`

On Windows

`%VARADHI%\samples`

The table below lists a CORBA feature or Varadhi functionality illustrated by each Varadhi sample.

CORBA Feature or Varadhi Functionality	See this Varadhi Sample
A simple Varadhi application	<i>Adder</i>
A single program acting as both client and server	<i>Multiplier</i>
Collocation of objects	<i>Coloc</i>
Usage of ORB::work_pending() and ORB::perform_work() to perform client and server tasks in a single main thread	<i>Pingpong</i>
Usage of sequence of strings	<i>Stringseq</i>
IDL Union Datatype	<i>Union</i>
Sending and Receiving CORBA objects between clients and servers	<i>Hostserver</i>
POA's persistent lifespan policy for creating persistent objects	<i>Persist</i>
CORBA Exception Handling and Varadhi Extension	<i>Exceptions</i>
Usage of Varadhi Naming Server, ns, to locate and invoke operation on a server object	<i>Names</i>
Usage of corbaloc format object url	<i>corbaloc</i>
Usage of ORB::work_pending() and ORB::perform_work()	<i>chat</i>

To download and evaluate SANKHYA Varadhi, go to:
<http://www.sankhya.com/info/products/varadhi/download.html>

After installing Varadhi, refer to \$VARADHI/samples/README for information on building and running the demo programs.

Here is more information on the sample programs:

1. Adder:

This demo program illustrates handling of client request by the server for simple operations (here, addition of two numbers) and return the result (value) to the client.

2. Multiplier:

This is similar to adder demo. Here a single application acts as both client and server. By making use of the service of an adder server, the multiplier server multiplies two numbers using addition.

3. Coloc:

Collocation is a demo program to illustrate collocation of objects. The demo application multiplies two numbers using a collocated adder object.

4. Pingpong:

This demo program illustrates how the main application thread can use the polling mechanism to multiplex between ORB requests and other application specific activities. This demo program uses the following Varadhi functionality:

ORB::work_pending()

ORB::perform_work()

to perform both client and server tasks in a single main thread.

5. StringSeq:

StringSeq is a demo program to illustrate String Sequence usage. The samples directory contains a client/server implementation of String Sequence demo. The server stores a string sequence and client sets and gets values from it.

6. Union:

Union is a demo program to illustrate IDL Union data type usage. This is a client/server implementation of union demo which adds two unions just like the adder demo.

7. Hostserver:

Hostserver is a demo program to illustrate passing of objects between clients and servers. It behaves like a Nameserver. The 'hostserver' program stores the Object Reference of Objects (servers) of "IDL:Host" type, running in different host machines. Clients use this Object reference to invoke operation on the individual 'host' servers.

8. Persist:

Persist is a demo program to illustrate PERSISTENT CORBA Object. The demo program adds two numbers using a Persistent CORBA Object.

9. Exceptions:

Exceptions is a demo program to illustrate exception handling in CORBA. The application adds two numbers using a client/server model and throws/catches CORBA exceptions for wrong arguments. The demo also illustrates the usage of Varadhi Extension to deal with CORBA exceptions on systems with no support for CORBA exception handling.

10. Names:

Names is a demo program that uses Varadhi Naming Server to locate and invoke operation on a server object. This is a client/server implementation of adder demo that uses Varadhi Naming Service instead of stringified IOR files.

11. corbaloc:

Corbaloc is a demo program to demonstrate use of corbaloc format object url. The server creates "Adder" object and registers it. The client uses the corbaloc format Object URL to get the object reference of the "Adder" object using the Object ID "Adder".

12. chat:

Chat is a demo program that uses `ORB::work_pending()` and `ORB::perform_work()` operations. The demo program acts as both client and server. This is a normal interactive chat program.

This is another demo program (like pingpong) that illustrates how the main application thread can use the polling mechanism to multiplex between ORB requests and other application specific activities.

Conclusion

In this application note we have seen how Varadhi samples illustrate the key features available in CORBA.