Instructor:
Dr. James W. Hong

Computer Science
402a/533a
Assignment 3

Due:
9:00 am, Thursday,
November 10, 1994

This assignment involves designing and implementing an ODP Trader-based Resource Management System (TBRMS) on OSF Distributed Computing Environment (DCE). A client application is also required to interact with the TBRMS server to test its functionalities. This assignment is worth 12% of the final mark and can be done individually or groups of no more than three.

The TBRMS is to manage information on various resources in a distributed computing environment. It is to manage information on both hardware and software resources. Examples of hardware resources include network devices (*e.g.*, routers, bridges), computing devices (*e.g.*, mainframes, workstations, X-terminals) and peripheral devices (*e.g.*, printers, plotters, magnetic tape drives). Software resources may include compilers, editors, debuggers, typesetters and a variety of services (*e.g.*, security service, name service, file transfer service, transaction service, and time service).

ISO has begun work on a proposed standard for Open Distributed Processing (ODP). Included in this proposed standard is a mechanism called the Trader which provides a framework for "trading" services in an open distributed computing environment. "Trading" is an ODP term which is defined as the sharing of services between ODP processes. The TBRMS should employ and extend the ODP Trader concept to the management of general resources in a distributed computing environment. For further information on the ODP Trader, read the attached article "Static and Dynamic Trading in Distributed Systems" by C. Popien and the ODP Trader Standard Working Document (which will be available in the cs402 folder in the library).

The minimum set of functions that the TBRMS must provide to clients are as follows:

register – allows a client to register itself with a TBRMS. Note that a client can be an importer and/or an exporter.

deregister – allows a client to deregister itself from a TBRMS.

export – used by an exporter wishing to make a resource available through the TBRMS. The exporting client supplies to the TBRMS the resource properties for a resource.

update – used by an exporter which, after previously exporting a resource, now wishes to update some or all values associated with that resource.

withdraw – used by an exporter which, after previously exporting a resource, now wishes to remove the reference of the resource from the TBRMS.

list – used by an importer to retrieve the details of a particular resource. A client may use the list operation on a variety of resources to select the most appropriate resource to fulfill its resource needs.

search – used by an importer to discover the resources matching a set of resource requirements.

select – used by an importer client to retrieve the interface to a resource.

One of the important design aspects of the TBRMS is information modelling. How should the information on resources (both hardware and software) be represented? How should the information be stored and where? Another important aspect is resource matching. How does the TBRMS find the resources that the clients want? Should we use name matching, type matching, attribute-matching or a combination of those? Answers to these questions must be stated in your design documentation.

A simple client application that tests all the functionalities of the TBRMS must be developed as part of this assignment. User interface to the client application is up to you. That is, you can design a command-driven, menu-driven or graphical user interface (GUI). A user-friendly user interface is encouraged but no extra mark will be awarded for providing a GUI.

A bonus mark (up to 5%) will be awarded for getting the client-server application you developed for your ASN2 to work using the TBRMS. This will involve modifying your ASN2 application slightly to use DCE RPC from Sun RPC but the bulk of the application should not need to be changed. ASN2 may utilize the TBRMS service as follows. The server part of the ASN2 will register and export the service it provides to the TBRMS server and the client part of the ASN2 will register and import the service it needs from the TBRMS server.

The TBRMS server must be able to serve multiple clients simultaneously. This can be achieved by employing **threads**. That is, the server can create a separate thread to handle each of the client's requests.

Resource information can be stored in-memory tables or on flat files maintained by the TBRMS. A bonus mark (up to 10%) will be award to the implementation of TBRMS that uses some type of information repository (such as X.500 Directory, DB2/6000) for storing resource information.

DCE RPC is to be used for communication between the client(s) and the server. Whether to use TCP or UDP for the RPC's underlying transport mechanism is up to you. Several books on DCE will be available on reserve for reference purposes. Some sample DCE programs are available in the `/gaul/s0/course/cs402/DCE_RPC` directory.

For testing, you should have at least two clients interacting with the server running on another host simultaneously. Appropriate error messages are required to be printed for invalid or inappropriate requests from the client. The marking scheme will be similar to the first assignment.

You must hand in the following for this assignment:

- design documentation
- system documentation
- test documentation
- source listing.

Make sure to include my and the TA's username in your .rhosts file (*e.g.*, `mccarthy.csd.uwo.ca jwkhong`, `mccarthy.csd.uwo.ca dstokes`) and state where your program can be found so that we can verify your program (if necessary).

There will be a penalty of 20% of the mark for the assignment turned in less than one day late, and an additional penalty of 10% for each day thereafter.