Laboratory Goals / Objectives

The purpose of this lab is to investigate the discovery service by designing and implementing a directory-based solution. At the end of the lab, an understanding of the discovery service should be demonstrated, by providing a working simulation of the implementation.

Overview

This laboratory work is dedicated to extending the student knowledge and skills on discovery services in general and directory-based ones in particular. To achieve this goal, the student is required to take decisions regarding a convention for service definition, to create the set of protocols for server-directory-client interactions and for considering and implementing (if possible) a service provisioning mechanism.

For distributed systems, users and their applications need to discover and use services or resources not available on their computers/mobiles. This operation is particularly important for mobile users or in dynamic networks where new services are offered and other services cease to exist (stop running). Other reasons for discovering remote service is load balancing and the need to access unique resources that are shared.

Users are interested in finding *appropriate service*, according to the user definition: (low) cost, (easy) location accessibility, trustworthy, secure access etc. The appropriateness of a service can be determined by examining attributes stored with the service offer.

A directory can be used to store the service offers made by servers, as it provides more information than the name of the service. The lookup process may start with a name and produce the service reference (net address) and attributes, or, alternatively, start with a set of attributes and find out the service.

To implement a directory-based discovery service, several decisions have to be considered.

1. The convention used for creating the service description depends on the communication protocol stack adopted by the server and the client. For example, beside the name of the service, it can include the service URL, such as in RMI, or it can have more information such as in the case of UPnP (service type, service ID, description file, etc – see lecture notes 6). As mentioned before, adding attributes to the name record allows the client to select the service that meets certain requirements.

2. The next important decision after adopting the service description rules is the set of protocols that will be used by the server to discover the directory service, to make its service offer public and, later, to interact with clients. The same for the client which has to discover the directory service, lookup the service and use the reference to address its request to the server.

3. Finally, any discovery service has to provide means for service provisioning information. For example, Jini uses leases for the service record and also for the client/server sessions. This is good practice. Another aspect here is that a service can be used exclusively (accepts only one client at a time), or non-exclusively (for each client request, a new thread is created). If one chooses an exclusive use of the service, the service may create and manage a flag that will indicate if it is busy or not.

Lab Work

There are several assumptions that need to be considered. The most important is that the directory, the server and the client will run on the same computer, and not on different computers. Therefore, in this case, the discovery is about the availability of the service (running or not, busy or not). The client looks up services which are running and not busy. The attributes of the service are important and are an alternative to the name for the look up operation. The servers and clients don't need to discover the directory in this case. Directory discovery is replaced in this simplification by simple invocation of the registration/lookup method of the directory server.

Tasks to Do

1. **Analysis**. First, draw the general, distributed, directory-server-client model, explaining the role played by each of them. Choose and present the convention for service definition (e.g., name, IP address, port number, etc). Then, consider all the actions associated with their interaction protocols, such as discovery of the directory service, registration of the service offer in the directory, lookup by the client, etc.

2. **Design**. Outline the protocols in pseudo-code. This will be your design.

3. **Programming**. Adapt the protocols above for one computer environment and then provide an implementation of the directory, server and client (in Java or Python).

4. **Testing**. Test your implementation by simulating the execution of the protocols: the server registers its service offer in the directory, the client looks-up the directory, either by service name or by service attributes, the client issues the request to the server, the server runs and returns the result. You may implement two separate servers, one being the calculator developed in the previous lab.

Submission

Return your results to Tasks 1-4 described above, including the pseudo-code, the **code** and **screenshots**, in a **pdf** file on moodle, by the deadline. At the end of the first week you should be able to show and explain the analysis and the design sections tasks 1 and 2.

Add comments to your code. Place your name and student number at the top of each class file. Place comments at the method level; use one-line comments inside method bodies to describe more complex statements if you feel they are not obvious.

Questions

The following questions are to be filled in individually by each student and the pdf file retuned through Moodle before the deadline.

1. What is a discovery service?

2. Provide the convention used for service definition and explain your choice.

3. Provide the code for the directory server.

4. Provide the code for the server.

5.Provide the code for the client.

6. What service provisioning mechanism did you consider?

7. Your additional comments on implementing this service.