Resource Sharing among devices in a network

Sneha Madle* and Sanyam Mehta*

Department of Computer Science & Information Systems,

BITS PILANI K K Birla Goa Campus

E-mail: f20140705@goa.bits-pilani.ac.in; f20140526@goa.bits-pilani.ac.in

Abstract

Advancements in computing are leading to a promising future for collaborative computing, where globally-scattered distributed resources (i.e., files, software, etc.) belonging to different organizations or individuals (i.e., entities) are collectively used in a cooperative manner to provide services. We propose a platform which allows individuals to share the resources of their devices among themselves and to effectively utilize their computing power.

Introduction

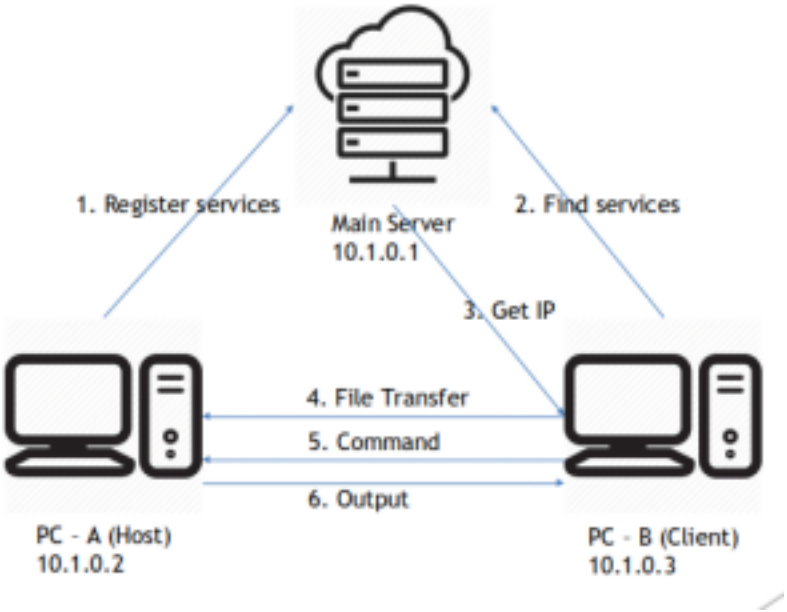
In computing, a shared resource, or network share, is a computer resource made available from one host to other hosts on a computer network. It is a device or piece of information on a computer that can be remotely accessed from another computer, typically via a local area network or an enterprise intranet, transparently as if it were a resource in the local machine. Network sharing is made possible by inter-process communication over the network.

Outline

The project aims to provide local resource sharing platform by connecting to an Ad Hoc network. Users can transfer multiple files to the system hosting the required service and then work on those transferred files using that service. Multiple users can connect and use the resources at the same time on any of the connected system. The platform is developed using Java to facilitate easy integration of the platform on systems running on any OS supporting Java JDK.

Working

In a network, one of the devices will act as a main server which maintains a database of all the services offered by the devices connected.



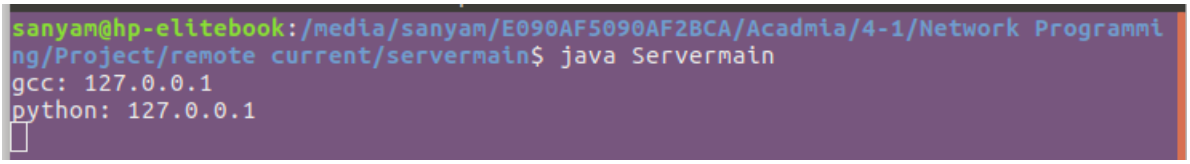
Scheme 1: Flow of data in the platform

1. When a system wants to share any service, it will connect to the main server and register all the services it is providing.

Scheme 2: Connecting to Main Server

Scheme 3: Registering services - gcc and python

The main server records all the services being hosted at a given time.



Scheme 4: List of registered services along with corresponding IP addresses

2. When one of the user wants to use a resource available, they connect to the main server to select the resource they need if it is hosted. Once they select a service hosted, the main server sends one of the IP addresses of systems hosting the particular service, to the user. The main server balances the no. of particular service requests to a host by implementing a round-robin algorithm.

```
sanyam@hp-elitebook:/media/sanyam/E090AF5090AF2BCA/Acadmia/4-1/Network Programmi
ng/Project/remote current/client$ java Client
Usage: java Client <host ip>
Now connecting to MainServer at 127.0.0.1:10000
Press 0: For gcc | 1: For python |
0
```

Scheme 5: Selecting the required service

3. Once the user is connected to the system hosting the required service, they will transfer all the files required to that system.

```
sanyam@hp-elitebook:/media/sanyam/E090AF5090AF2BCA/Acadmia/4-1/Network Programmi
ng/Project/remote current/client$ java Client
Usage: java Client <host ip>
Now connecting to MainServer at 127.0.0.1:10000
Press 0: For gcc | 1: For python |
0
Now connecting to 127.0.0.1:9999
filename ("exit" to complete file transfer): fib2.cpp
filename ("exit" to complete file transfer): exit
command (q to quit):
```

Scheme 6: Transferring files.

4. The next step is to give command prompt instructions to work on those files.

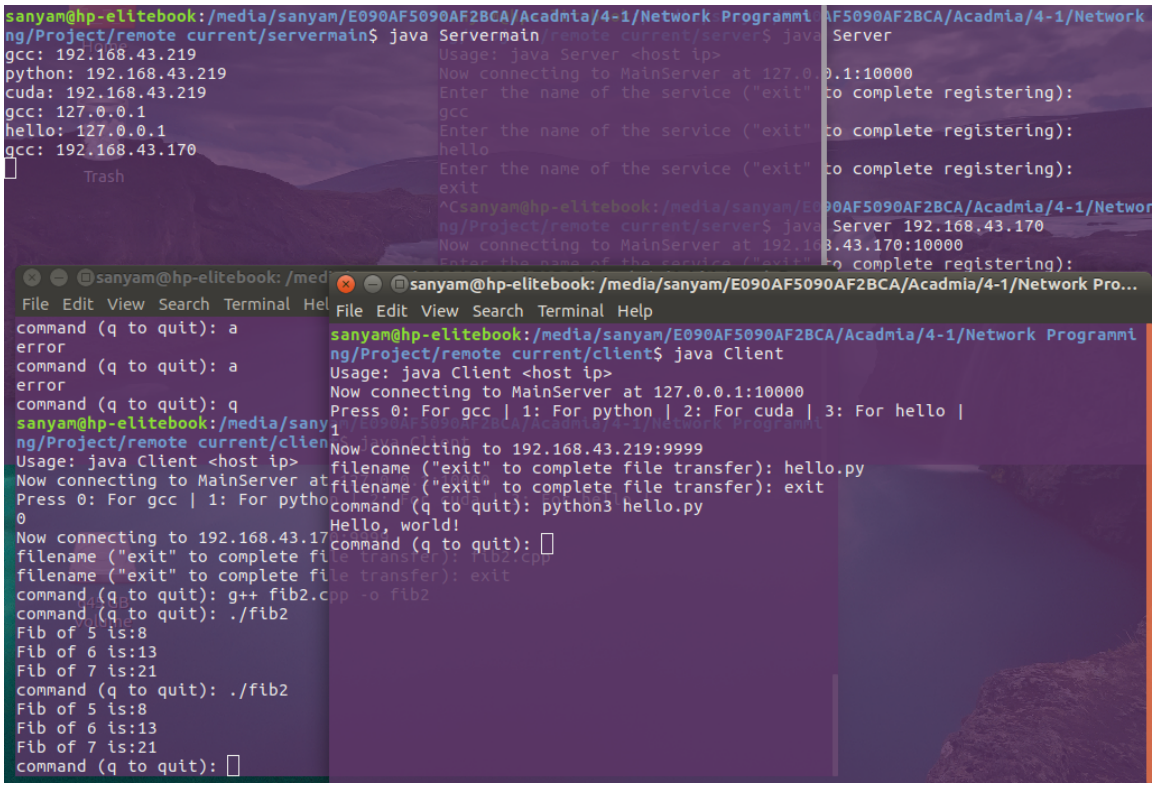
```
sanyam@hp-elitebook:/media/sanyam/E090AF5090AF2BCA/Acadmia/4-1/Network Programmi
ng/Project/remote current/client$ java Client
Usage: java Client <host ip>
Now connecting to MainServer at 127.0.0.1:10000
Press 0: For gcc | 1: For python |
0
Now connecting to 127.0.0.1:9999
filename ("exit" to complete file transfer): fib2.cpp
filename ("exit" to complete file transfer): exit
command (q to quit): g++ fib2.cpp -o fib2
command (q to quit): ./fib2
```

Scheme 7: Giving commands to compile and execute the file on the system connected.

```
sanyam@hp-elitebook:/media/sanyam/E090AF5090AF2BCA/Acadmia/4-1/Network Programmi
ng/Project/remote current/client$ java Client
Usage: java Client <host ip>
Now connecting to MainServer at 127.0.0.1:10000
Press 0: For gcc | 1: For python |
0
Now connecting to 127.0.0.1:9999
filename ("exit" to complete file transfer): fib2.cpp
filename ("exit" to complete file transfer): exit
command (q to quit): g++ fib2.cpp -o fib2
command (q to quit): ./fib2
Fib of 5 is:8
Fib of 6 is:13
Fib of 7 is:21
command (q to quit):
```

Scheme 8: Getting the output.

An example with three systems hosting multiple services



Scheme 9: Top left - Main Server; Top right - one of the three hosts; Bottom - two users.

Acknowledgement

We would like to express out sincere gratitude to our project instructor, Prof. Mahadev A. Gawas (Dept. of Computer Science & Information Systems, BITS Pilani K.K. Birla Goa Campus) whose contribution in stimulating suggestions and encouragement helped us to complete this project.