Simple Distributed File Management

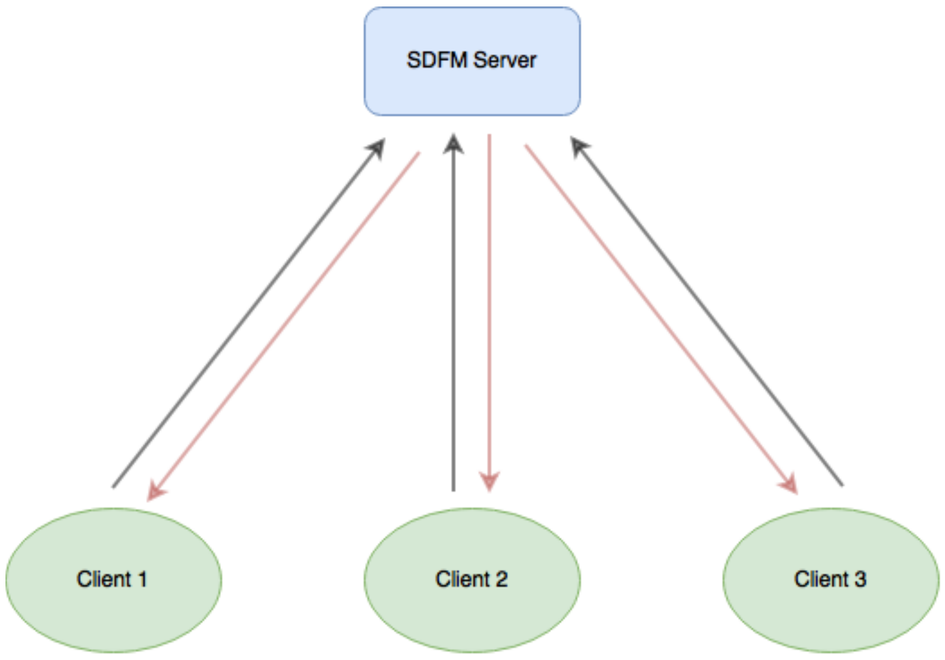

Figure 01 : Network Architecture of SDFM

As shown in Figure 1, **SDFM** contains two main entities: one **Server** and a set of **Clients.** the figure shows three **Clients**. Client-1, Client-2 and Client-3. Each Client stores a set of files in a directory called u_files. The Server maintains a **table**, shown in **Table 1** as an example, that contains six pieces of information.

1. File name
2. Client address
3. Client Port
4. Upload time
5. Last download time
6. Total downloads

The Client address can be the IP address (in dotted decimal notation), or its DNS name, of Client that stores the file. The Client Port is the TCP port number at which the client can be contacted to download the file. The Upload time is the time when the Client made the file available for sharing. The Last download time is the latest time the file was downloaded. The Total download gives the number of times the file has been downloaded.

Table 1: An example of Server table

File Name	Client Address	Client Port	Upload time	Last download time	Total Downloads
File1	Client-1	51000	Fri Jan 12 14:27:52 1996	Fri Jan 12 14:30:52 1996	5
File2	Client-2	50002	Sat Jan 13 11:20:42 1996		0
File3	Client-3	51098	Fri Jan 12 12:17:22 1996	Fri Jan 12 13:10:27 1996	3
File4	Client-4	45000	Fri Jan 12 14:37:55 1996	Fri Jan 12 14:47:15 1996	4

All communications between Clients are through Server. Specifically, assume Client-1 wants to download File 2that is located at Client-2 at Port 50002. Client-1 must send its request to Server that will then ask Client-2 to send File2 to Server. Receiving File2, Server then forwards it to Client-1.

Figure 1 shows one possible communication protocol to implement SDFM. As shown, each Client and Server pair is connected by two parallel TCP connections: Control Line and Data Line. Client uses Control Line to send a user command to Server; Server uses Control Line to send the result of each command to Client. Server uses Data Line to ask Client to send one of its files to Server. Client uses Data Line to send the requested file to Server. Note that Control Line is closed only when Client connection ends, while Data Line is closed each time a file transfer is completed.

For Control Line, Server listens (passive mode) at a well-known port, which in this case is one of your assigned port numbers. Each Client initiates a connection (active mode) to Server via the Control Line.

For Data Line each Client listens (passive mode) at an ephemeral port D that is created by the operating systems. Server uses port number L−1 to initiate a connection to its intended Client via port D.

A. Requirements

1. Each child Server-S is waiting for its Client-C"s valid commands at the Control Line and provides a service according to the command. If the command is valid, Server-S executes the command. Otherwise, Server-S sends an error message to Client-C.

   SDFM recognizes the following user commands (not case sensitive).

   a) upload file

      Client-C creates a listen socket at an ephemeral port D, and sends the “upload file” command together with D to Server-S via Control Line. Notice that at this stage, Client-C at port D becomes the “server” for file that is stored in its directory u_files. Receiving the command, Server-S creates a new entry in the Server table.

In the new entry, File name is set to file, Client address is set to the IP address of Client-C, Client Port is set to D, Upload time is set to the actual time when it receives the request, Last download time is set to null, and Total download is set to 0. Server-S then sends the new entry information to Client-C via Control Line. Client-C shows the new entry on its screen (e.g., for C = 1):

```
Upload File1
Location: Client-1, 51000
Upload time: Fri Jan 12 14:27:52 1996
Last download time:
Total download: 0
```

b) get table

Client-C sends the command to Server-S via the Control Line. Receiving the command, Server-S sends the Server table to Client-C, also via the Control Line. Note that the table is initially empty. Receiving the table, Client-C shows it on its screen. You can decide the format of the printed table on the screen;
e.g,

```
File1
Location: Client-1, 51000
Upload time: Fri Jan 12 14:27:52 1996
Last download time: Fri Jan 12 14:30:52 1996
Total download: 5

File2
Location: Client-2, 50002
Upload time: Sat Jan 13 11:20:42 1996
Last download time: None
Total download: 0

File3
Location: Client-3, 51098
Upload time: Fri Jan 12 12:17:22 1996
Last download time: Fri Jan 12 13:10:27 1996
Total download: 3
```

c) delete file

Client-C sends the command to Server-S via the Control Line. Receiving the command, Server-S deletes the entry for file from the Server table. Then, Server-S sends the deleted entry information to Client-C via Control line. Client-C shows the deleted entry on its screen

```
Deleted File1
Location: Client-1, 51000
Upload time: Fri Jan 12 14:27:52 1996
Last download time: Fri Jan 12 14:30:52 1996
Total download: 5
```

d) download file

Client-C sends the command to Server-S via the Control Line. Receiving the command, Server-S initiates a connection to the owner (“file server”) for file, say Client-2, at Port D. Server-S creates a Data line to Client-2. Then, Server-S asks Client-2 to send file2 via Data line. Receiving the request, Client-2 sends file2 to Server-S via Data line, and closes Data line. Receiving file2, Server-S closes Data line, increases Total download of file2 by one, updates Last download time with the current time, and sends file2 to Client-C. Server-S then prints a message on its screen, e.g.,

> File2 from Client-2 has been successfully transmitted to Client-C on Fri Jan 12 14:30:52 1996

Receiving the file2, Client-C stores it in a directory called d_files, and print a message on its screen stating that the operation has been completed, e.g.,

> File2 has been successfully downloaded from Client-2 on Fri Jan 12 14:30:54 1996

Note that you are allowed to create and implement your own protocol and/or communication model to implement command “download file”, without using Data Line. However, your report must include detailed discussion of the protocol, including any effect of your protocol to the other commands.

e) quit

Client-C wants to terminate the connection with Server-S, i.e., closing Control Line. Receiving the command, Server-S closes Control Line to Client-C. Further, Server-S deletes all Client-C"s entries from the Server table. Client-C closes its Control Line and prints a message on its screen and closes its sockets, e.g.,

> Client-C has terminated.

2. The concurrent Server is able to accept up to max_clients simultaneous Clients. If the child Server-S does not receive any command (from its Client-C) within a max_wait_time, Server-S sends a „good bye" message, closes Control Line and terminates. The max_clients, and max_wait_time must be arguments passed to Server when Server is started.