CSC337401 - Adv.Distribut.Programming Paradigms

Homework 1 - Design Document

Author:
Mehdi Alami Idrissi

Instructor:
Dr. Omar Iraqi

September 30, 2023

1 Introduction

The ‘xFx App‘ is an extension of the ‘Fx App‘, designed to offer advanced functionalities tailored to enhance user experience and optimize network resources. The new features introduced in the ‘xFx App‘ allow clients to retrieve a list of shareable files, resume interrupted downloads, and avoid redundant downloads of previously fetched files.

2 xFx Protocol

The `xFx Protocol` facilitates file operations between a client and server. This document outlines the supported operations and their corresponding communication structures.

2.1 Establishing a Connection

To initiate any action, the client must first establish a connection with the server. Once connected, the client sends a *header* to inform the server about the intended operation.

2.2 Operations

2.2.1 List Files

When the client wishes to view all shareable files on the server:

- **Client Header:**

  ```
  list_files\n
  ```

- **Server Response:**
  The server replies with the names of all shareable files in the Server-Share directory:

  ```
  [file1 name]\n
  [file2 name]\n
  ...
  ```

2.2.2 Download

To download a file, the client sends the following:

- **Client Header:**

```
download [file name] [MD5 hash (optional)] [offset (optional)]\n
```

- **Server Responses:**
  - If the file is not found:

    ```
    NOT FOUND\n
    ```

  - If the file is found and the MD5 hash matches (no changes since the last check):

    ```
    FILE_UNCHANGED\n
    ```

  - If the file is found but either there's no MD5 hash or the hash differs:

    ```
    OK [file size]\n
    ```

    This is followed by the bytes of the file, starting from the optional offset (if provided).

2.2.3 Upload

To upload a file:

- **Client Header:**

```
upload [file name] [MD5 hash]\n
```

- **Server Responses:**
  - If the uploaded file's hash matches an existing file on the server:

    ```
    FILE_UNCHANGED\n
    ```

  - If the file doesn't match or doesn't exist on the server:

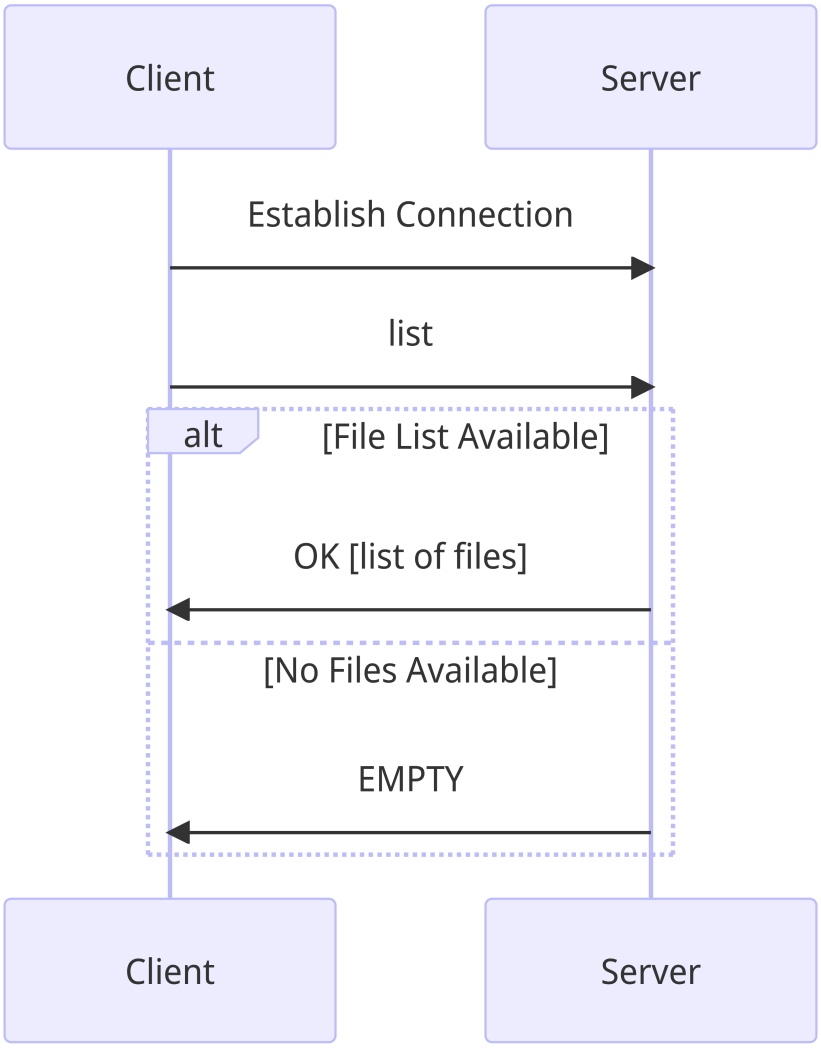
    ```
    PROCEED_WITH_UPLOAD\n
    ```

    After receiving `PROCEED_WITH_UPLOAD`, the client sends the bytes of the file. The server successfully receives and saves the file and logs the event. No confirmation is sent to the client.

3 Flow Diagrams

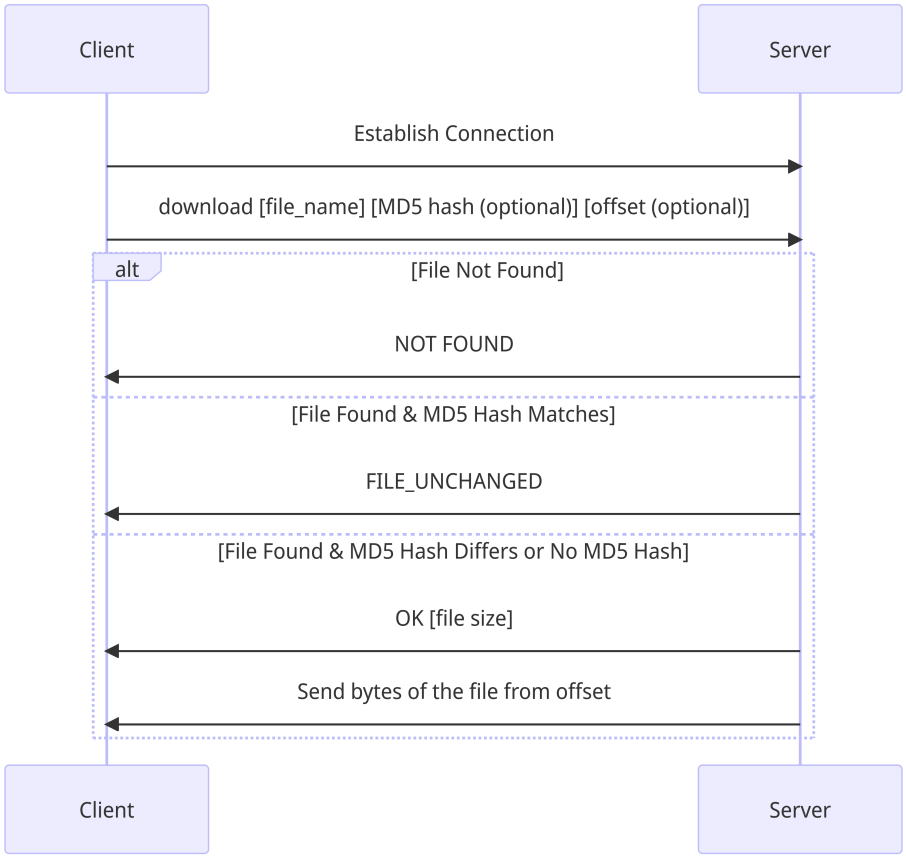
3.1 List Files Command

The command `sudo python3 Client.py l` is used to list all the shareable files on the server. The flow diagram for this command would illustrate the client sending a request to the server and the server responding with a list of available files.



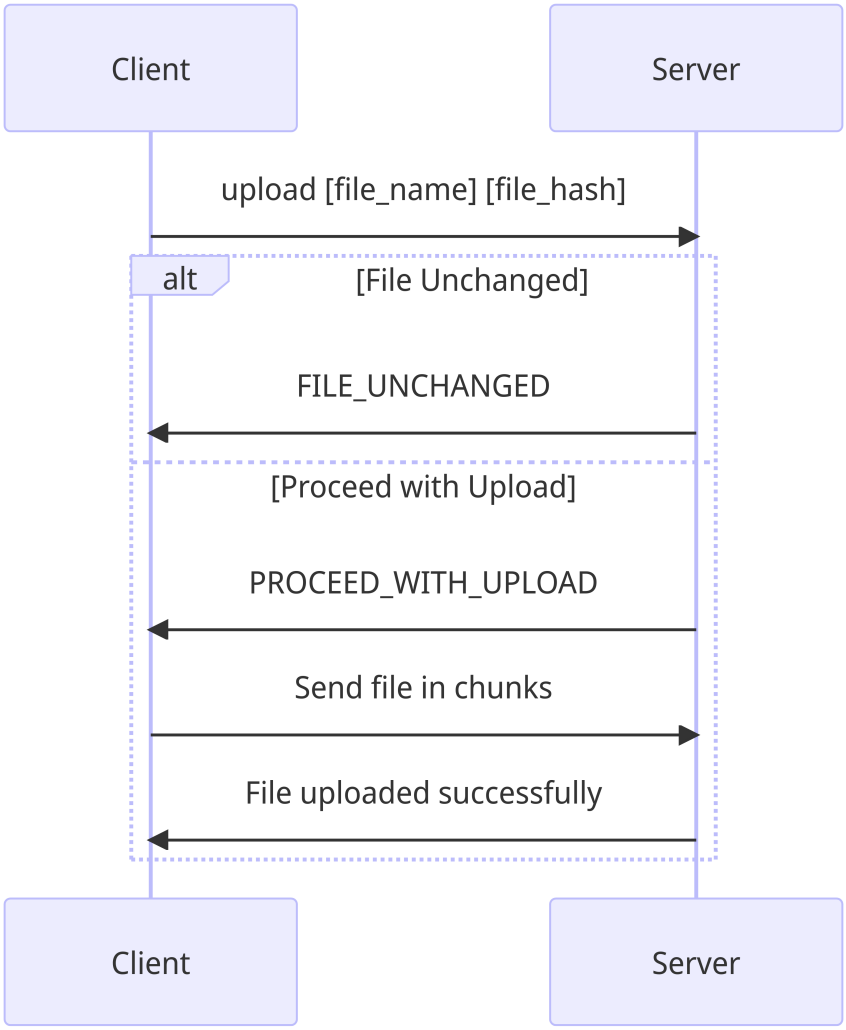
3.2 Download Command

The command `sudo python3 Client.py d [file_name]` initiates the download process for a specified file. The flow diagram for this command would depict the client sending a download request with the specified file name, and the server responding either with the file data or an error message if the file is not found.



3.3 Upload Command

The command `sudo python3 Client.py u [file_name]` is used to upload a specified file to the server. The flow diagram for this command would show the client sending the file data to the server, and the server acknowledging the receipt of the file or indicating an error if the upload fails.


Client
Server
upload [file_name] [file_hash]
alt
[File Unchanged]
FILE_UNCHANGED
[Proceed with Upload]
PROCEED_WITH_UPLOAD
Send file in chunks
File uploaded successfully
Client
Server