

UNIVERSITY OF SCIENCE AND TECHNOLOGY OF HANOI  
DEPARTMENT OF INFORMATION AND COMMUNICATIONS TECHNOLOGY



---

DISTRIBUTED SYTEM - PRACTICAL 2

# **RPC FILE TRANSFER**

---

By  
Truong Hoang Khanh Linh - 22BI13254

# 1 Design

The RPC service is built using Python's xmlrpc module, XML-RPC. The service enable remote communication between a client and a server over HTTP. It provides service for file transfer operations.

- The client can send or download a file to the server.
- The sesrver listens for incoming client requests from the client and processes them.

Below is a architecture diagram for XML-RPC:

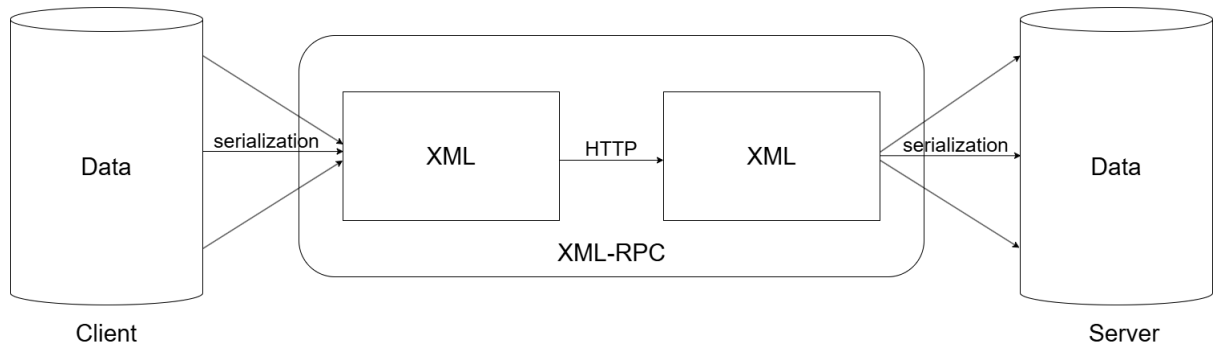


Figure 1: XML-RPC Architecture

## 2 System organization

The system follows a client-server architecture, with them communicating over XML-RPC.

- The server

The server hosts the file transfer service and listens on port 888. Its provides RPC functions to receive the file and return the requested file from the client. Received files are managed locally.

- The client

The client connects to the server through the server URL link. It offers two functions for file transfer service: to read a local file and send it to the server, to request a file from the server and save it locally.

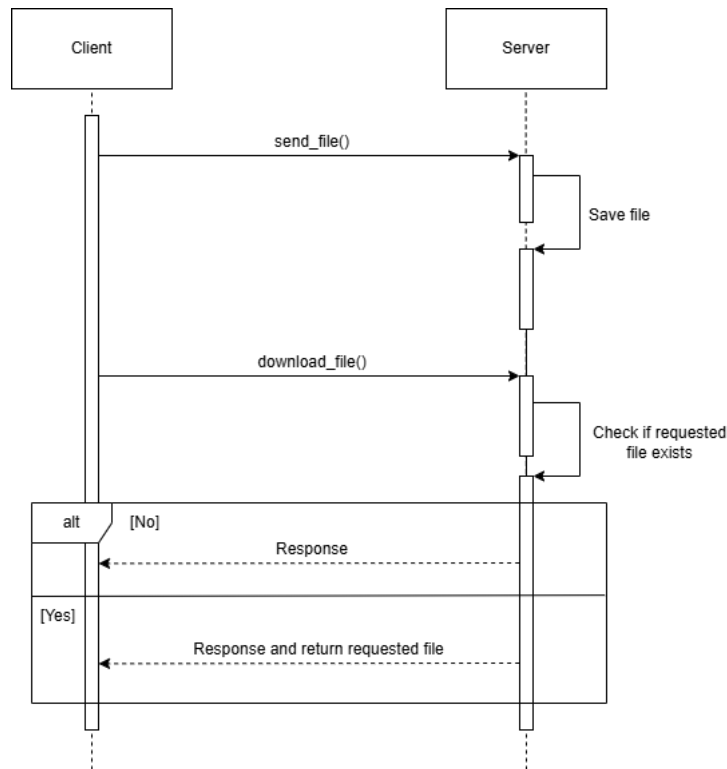


Figure 2: Work Flow of the system

## 3 Implementation

The system is implemented in Python using RPC. The client and the server transfer files over XML-RPC.

### 3.1 The client

To send a file to the server, the client runs the `download_file()` function, passing the filename and the file content in binary mode. It wraps the content using `xmlrpc.client.Binary()` to ensure the transmission of binary data over the XML-RPC protocol.

To download a file from the server, the client requests a file by calling `download_file()` with the desired filename. If the download is successful, the downloaded file is saved locally with a specified name.

```

import xmlrpc.client

def send_file(server_url, filename):
    with open(filename, 'rb') as file:
        file_content = file.read()

    proxy = xmlrpc.client.ServerProxy(server_url)    # Create an XML
                                                    -RPC server proxy

    # Send the file to the server
    response = proxy.send_file(filename, xmlrpc.client.Binary(
        file_content))
    print(response)

def download_file(server_url, request_filename):
    proxy = xmlrpc.client.ServerProxy(server_url)

    # Request the file to download from the server
    file_content, message = proxy.download_file(request_filename)

    # Save the downloaded file
    if file_content.data:
        with open('downloaded_file', 'wb') as file:
            file.write(file_content.data)
    print(message)

if __name__ == "__main__":
    server_url = "http://192.168.86.128:888"

    download_file(server_url, "file_to_download.txt")
    send_file(server_url, "file_to_send.txt")

```

## 3.2 The server

The server receives the file transferred from the client with `send_file()`, saves it locally as `\received_file`, and returns a success message to the client.

After receiving the request to download file from the client, the server first checks for the file's existence. If the file exists, it reads the requested file content in binary mode, wraps the content using `xmlrpc.client.Binary()` and then sends it back to the client along with a success message. If the file does not exist, it will return an message indicating the error.

Below is the code for the server:

```

import xmlrpc.server
import os

def send_file(filename, file_content):
    # Save the received file
    with open('received_file', 'wb') as file:
        file.write(file_content.data)
    print(f"File received and saved as 'received_file'.")
    return "File sent successfully."

def download_file(filename):
    # Check if the requested file exists
    if not os.path.exists(filename):
        return "File does not exist."

    with open(filename, 'rb') as file:
        file_content = file.read()
    print(f"Requested file sent successfully.")
    return xmlrpc.client.Binary(file_content), "File downloaded and saved as 'downloaded_file'."

if __name__ == "__main__":
    server = xmlrpc.server.SimpleXMLRPCServer(("192.168.86.128", 888))

    # Register the functions with the XML-RPC server
    server.register_function(send_file, "send_file")
    server.register_function(download_file, "download_file")

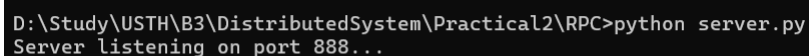
    print("Server listening on port 888...")

    # Start the server
    server.serve_forever()

```

## 4 Testing

First, run **server.py** to set up the server, let it listen to connections on port 888.



```

D:\Study\USTH\B3\DistributedSystem\Practical2\RPC>python server.py
Server listening on port 888...

```

After that, run **client.py** to connect the client to the server. Once the connection is established, the client will transfer file to the server and download file from the server at the same time.

Files are now transferred successfully as shown in the images below.

```
C:\Users\user\Documents>python client.py
File downloaded and saved as 'downloaded_file'.
File sent successfully.

C:\Users\user\Documents>type downloaded_file
[Content of file to download]
C:\Users\user\Documents>_
```

```
D:\Study\USTH\B3\DistributedSystem\Practical2\RPC>python server.py
Server listening on port 888...
Requested file sent successfully.
192.168.86.128 - - [20/Dec/2024 02:01:20] "POST /RPC2 HTTP/1.1" 200 -
File received and saved as 'received_file'.
192.168.86.128 - - [20/Dec/2024 02:01:20] "POST /RPC2 HTTP/1.1" 200 -
|
```

```
D:\Study\USTH\B3\DistributedSystem\Practical2\RPC>type received_file
[Content of file to send]
D:\Study\USTH\B3\DistributedSystem\Practical2\RPC>|
```