Network Management using RPC

Peeyush Gupa

CpeE - 550

peeyush@iastate.edu

System Level Design:

1. **Overview**: In the Remote System Monitoring application, clients can request statistics from distant servers using the Network Management Application, a distributed system. The program keeps track of CPU use, user logins, and other statistics on a host and sends this information to clients across a network. With various servers processing requests from numerous clients, the system is built to be fault-tolerant.

2. **Architecture**: The application has two main components Client and Server. The Client allows users to request different parameters like date, time, CPU usage, memory usage, etc., of the remote machine. And Server fetches these parameters by using system files and C libraries to get the values requested by the user. The communication occurs over an RPC (Remote Procedure Call).

3. **Components**:

    a. Client: The client will send the request to the server for the data requested by the user, receive the server's response, and display it to users. The functions supported by clients-

        i. Date and Time - Sends requests for fetching Date and Time values from the remote machine. This function will take in a parameter to get the format you want to give the value (whether you want Date, Time, or both).

        ii. CPU Usage - Sends requests for fetching the CPU usage of the remote machine. It will receive the usage of CPU in double.

        iii. Memory Usage- Sends requests to fetch the remote machine's RAM and page data information. This client function calls two different server functions. One to fetch the RAM data and the other to fetch the Page Data.

iv. Load Usage- Sends a request to fetch load averages for 1min, 5min, and 15mins. The response is received in the form of a string with load averages for 1min, 5min, and 15mins.

v. User List - Sends a request to show the user list in the Linux system. It fetches the list of users. The response is received in the form of a string.

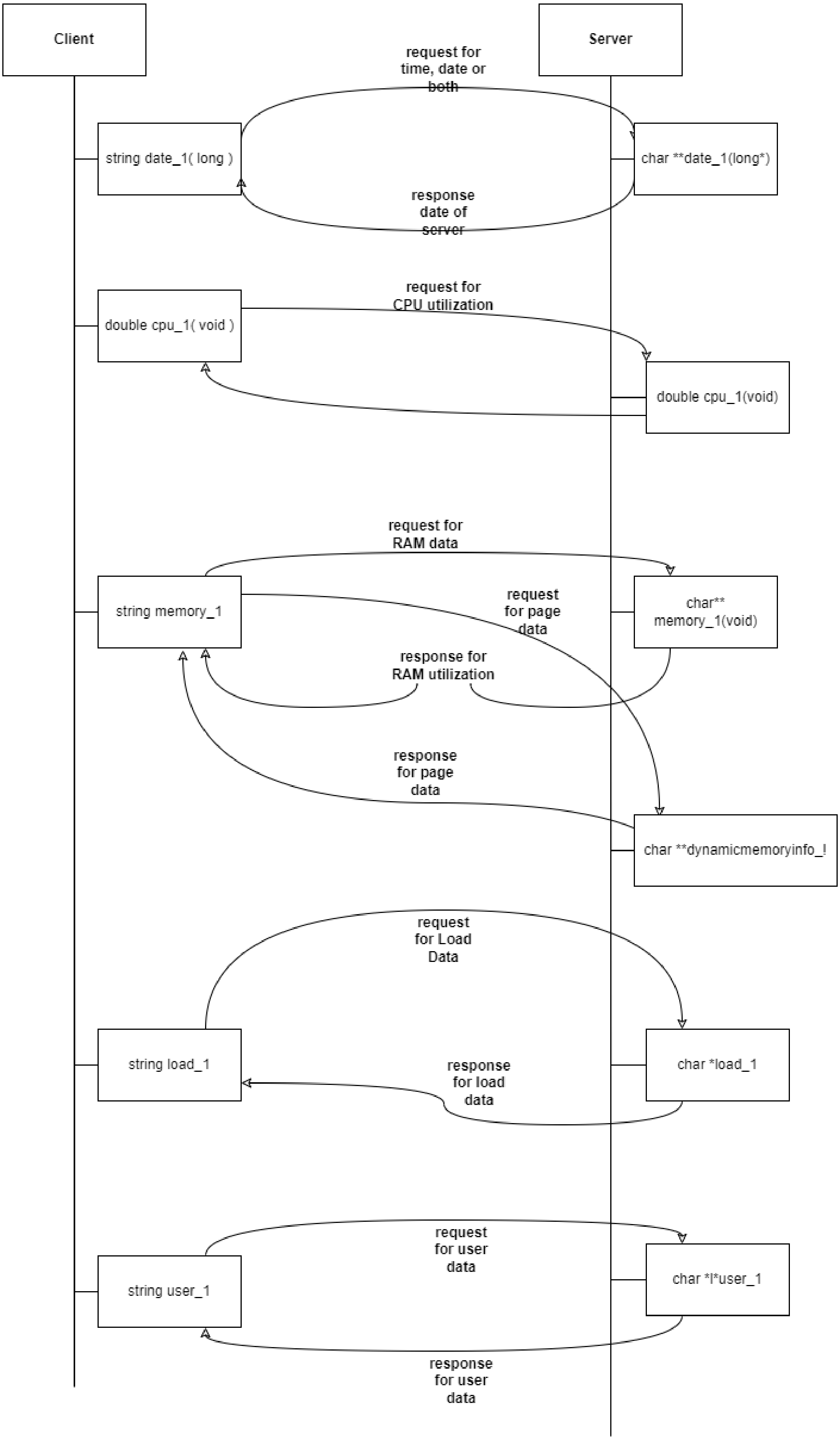
b. Server: The server will receive the request from the client, calculate the required values, and sends the response back to the client.

i. Date and Time- Fetches the request for date and time and responds with the date and time value. It uses the `strftime` function to fetch date and time values. Based on the input parameter received (1,2 or 3), the date, the time, or both will be sent in response.

ii. CPU Usage- Will use the `‘/proc/stat’` file and sends the CPU utilization value to the client.

iii. Memory Usage- It provides details for RAM, each page size, and the total pages available. The RAM details are obtained from the `sysinfo` function. And for page details, the values are obtained from the `mallinfo` function present in `malloc.h` library for c.

iv. Load Usage- It provides the load on the system for 1min, 5 mins and 15mins. It gets the response from the `getloadavg()` function and returns the value to the client.

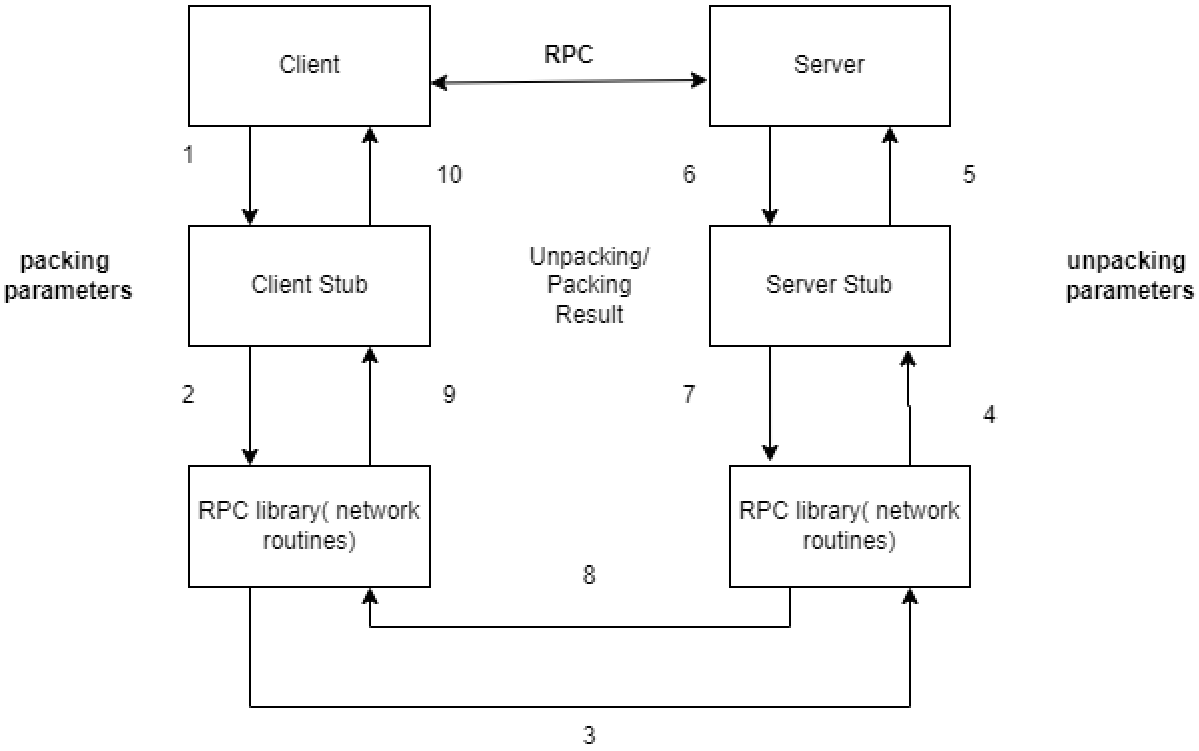

Client
Server
request for
time, date or
both
string date_1(long)
char **date_1(long*)
response
date of
server
request for
CPU utilization
double cpu_1(void)
double cpu_1(void)
request for
RAM data
string memory_1
request
for page
data
char**
memory_1(void)
response for
RAM utilization
response
for page
data
char **dynamicmemoryinfo_!
request
for Load
Data
string load_1
char *load_1
response
for load
data
request
for user
data
string user_1
char *l*user_1
response
for user
data

How functions will look like.

The RPC Structure:

The client application can use the Remote Procedure Call (RPC) technology to call functions or processes that are executing on a distant server as if they were operating on the client system. In distributed systems, RPC is frequently used to enable remote access to services.

A client must send a request to the server via the network in order for the server to receive it, process it, and then provide the client a response in order for a remote procedure call to be made. Both the client and the server must implement specific software elements known as "stubs" in order to achieve this.



Source- Class study material (Inter-Process Communication: RPC + **1st Course Project (CP 1)**: Remote System Monitoring using RPC)

A stub is a piece of code that acts as a proxy for the actual function or procedure running on the server. There are two types of stubs: client-side stubs and server-side stubs.

- Client-Side Stubs - The software elements in charge of making a request to the server and getting a response are known as client-side stubs. The client-side stub creates a request message with the parameters of the remote procedure and delivers it over the network to the server whenever the client application calls a remote operation.
  The client-side stub likewise awaits the server's answer, and when it comes, it unpacks the message and sends the outcome back to the client application. The client program is protected from the difficulties of network connection by the client-side stub, which takes care of all the specifics of sending and receiving messages over the network.
- Server-Side Stubs - The software components known as server-side stubs are those that receive the request from the client-side stub, use the server's real process or function, and then transmit the server's response back to the client-side stub. The actual method or function on the server is called when the server-side stub unpacked a request message it has received via the network.
  The client-side stub receives a response message from the server-side stub across the network when the procedure or function has finished running and contains its results. Unpacking the request message, calling the relevant function, and delivering the response message back to the client are all handled by the server-side stub.

In summary, client-side and server-side stubs are essential components of RPC systems. The client-side stub handles the communication between the client application and the server, while the server-side stub handles the communication between the server and the client. By using stubs, RPC systems allow remote procedure calls to be made as if they were running on the local machine while abstracting away the details of network communication.