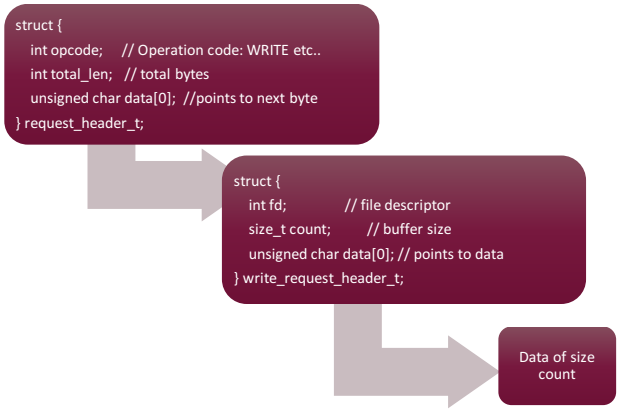
Design Notes

Yuqi Liu yuqil@andrew.cmu.edu

1. Serialization Protocol

I define different structures for data serialization of different function calls. In generally, structure **request_header_t** is transmitted at the first of all data. It dictates how many bytes of data in total and contains opcode which indicates operation type. The data[0] is a pointer that points to the next byte after the structure which is another header structure designated to a certain operation. Finally, it comes with real data that needs to be transmitted. The data structure for Write operation is shown in the left part.



Breath-first-search is used to serialize and deserialize the directory tree in level order.

Details of specific operation header type structure is defined in RPClib.h.

2. Handle Concurrent Clients

To handle concurrent clients, several design decisions I have made is described here.

- Client-server file descriptor conversion.

I perform client-server file descriptor conversion. In the client side, the file descriptor is between 0-1024. But the server side's file descriptor is between 25,000 – 26,024. That is to say, I add 25000 to file descriptor given out by server. In this way, one can easily distinguish between local file descriptor and server's descriptor. If it belongs to local descriptor, local function call is performed.

- Thread vs Process?

I use process at the server side to handle concurrent clients. The reason is mainly because it can avoid client A writing to client B's open file. Because file descriptor is not shared by different processes. So the file descriptor of client A's process cannot be seen by client B's process.