Protocol Description

1. The distinguish between different calls from clients

```
#define OPEN            0
#define WRITE           1
#define CLOSE           2
#define READ            3
#define LSEEK           4
#define STAT            5
#define UNLINK          6
#define GETDIRTREE      7
#define GETDIRENTRIES   8
#define FREEDIRTREE     9
```

Client and Server agrees on the beginning of different calls. This is predefined in the macro definition. So the first thing server gets will be the symbol of what type of call it is.

2. marshall and unmarshall of parameters

Clients and server will agree on the order and format of different parameters put in the message for simple calls. For example, the close call to the server will only send a "[fd]" instead of marshalling in a struct.

If the client is using a char * as a pathname or a buffer for write, then this part will be put at the very end of a message with a 4 byte integer header indicating its length. In this way communication is settled on both sides.

Direntries is the trickiest one. It is marshalled in a recursive way. For example, a directory with files will be marshalled in my protocol like:

[len of directory name]{directory name}[***number of children***]…

Concretely a directory called DIR with fileA and fileB will be marshalled like:

[4]{ DIR }[**2**] [6]{ fileA }[***0***] [6]{ fileB }[***0***]

3. Other considerations during design

I make an assumption that server is usually heavily loaded. So the bandwidth of the server is limited. If a client want to ask for or write to a huge file and transfer 1000000 Bytes at time. My protocol will restrict the transmission to 1024 Bytes per call. By this design, the server won't be heavily loaded.

Besides, client might open files both on the client side and the server side. To distinguish the files between both sides. I add an OFFSET to the file descriptor returned from server. Thus fd less than OFFSET is considered as the files on the local machine; fd larger than OFFSET is considered the files on the remote machines.