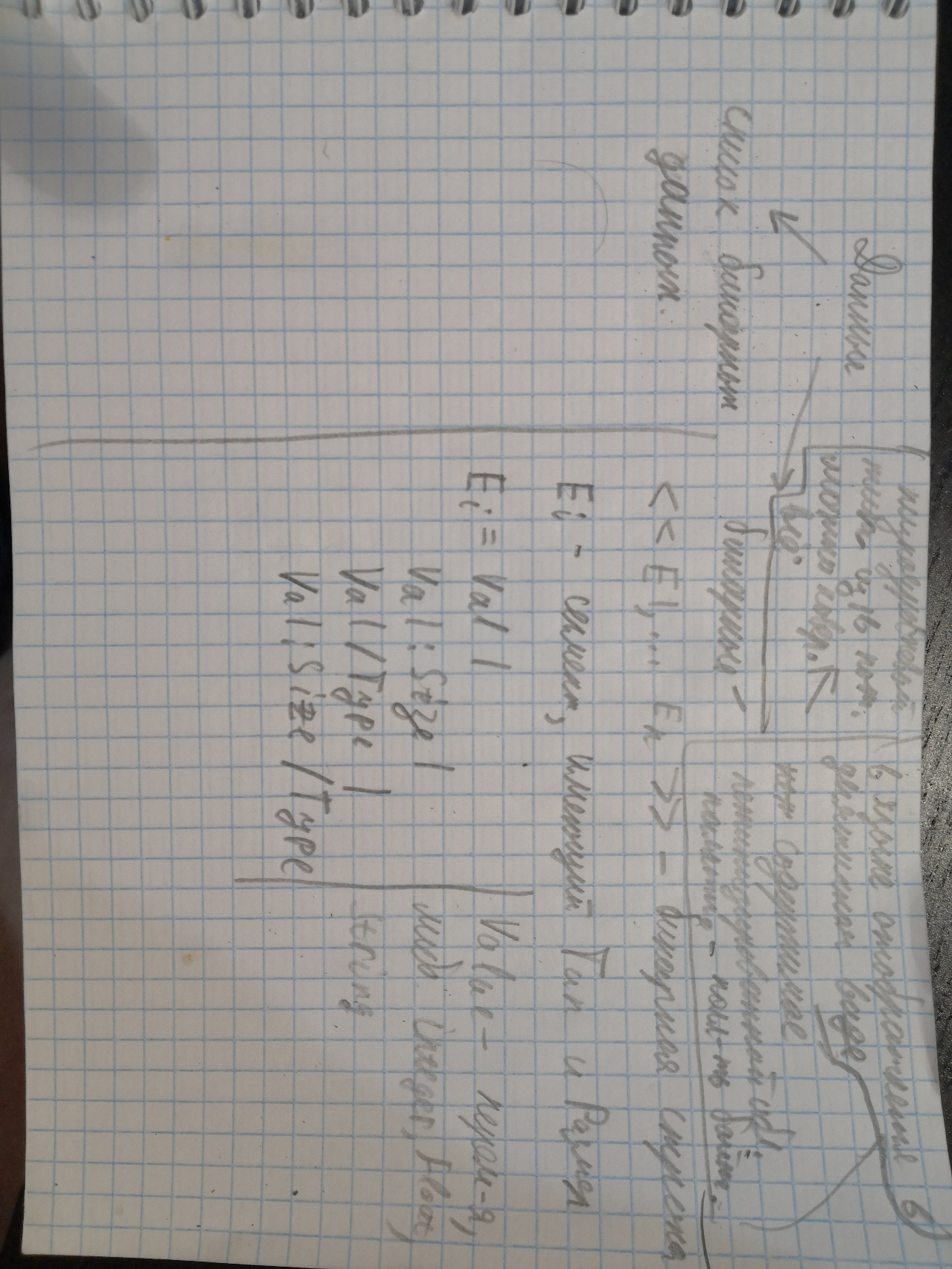
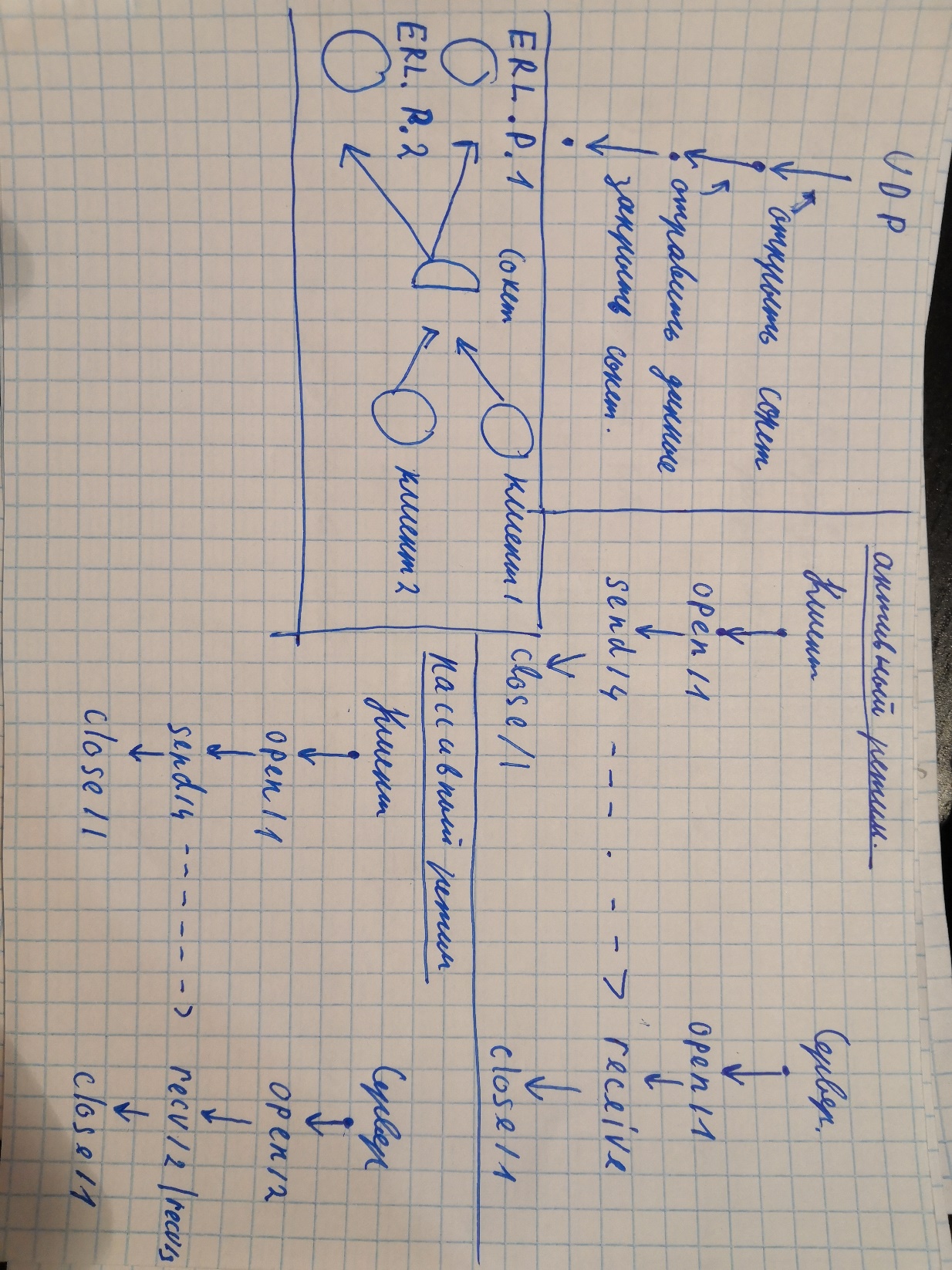
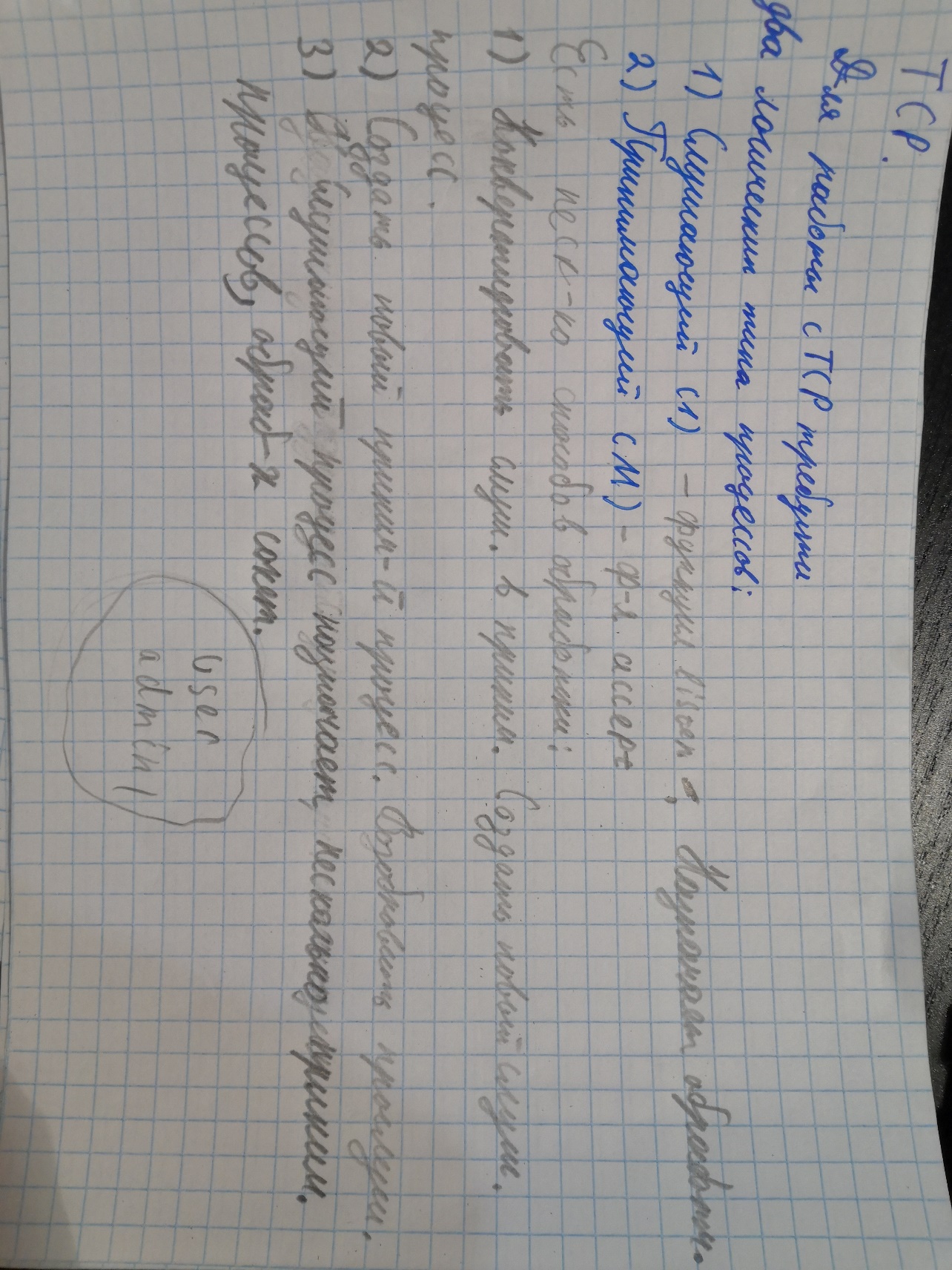
# UDP

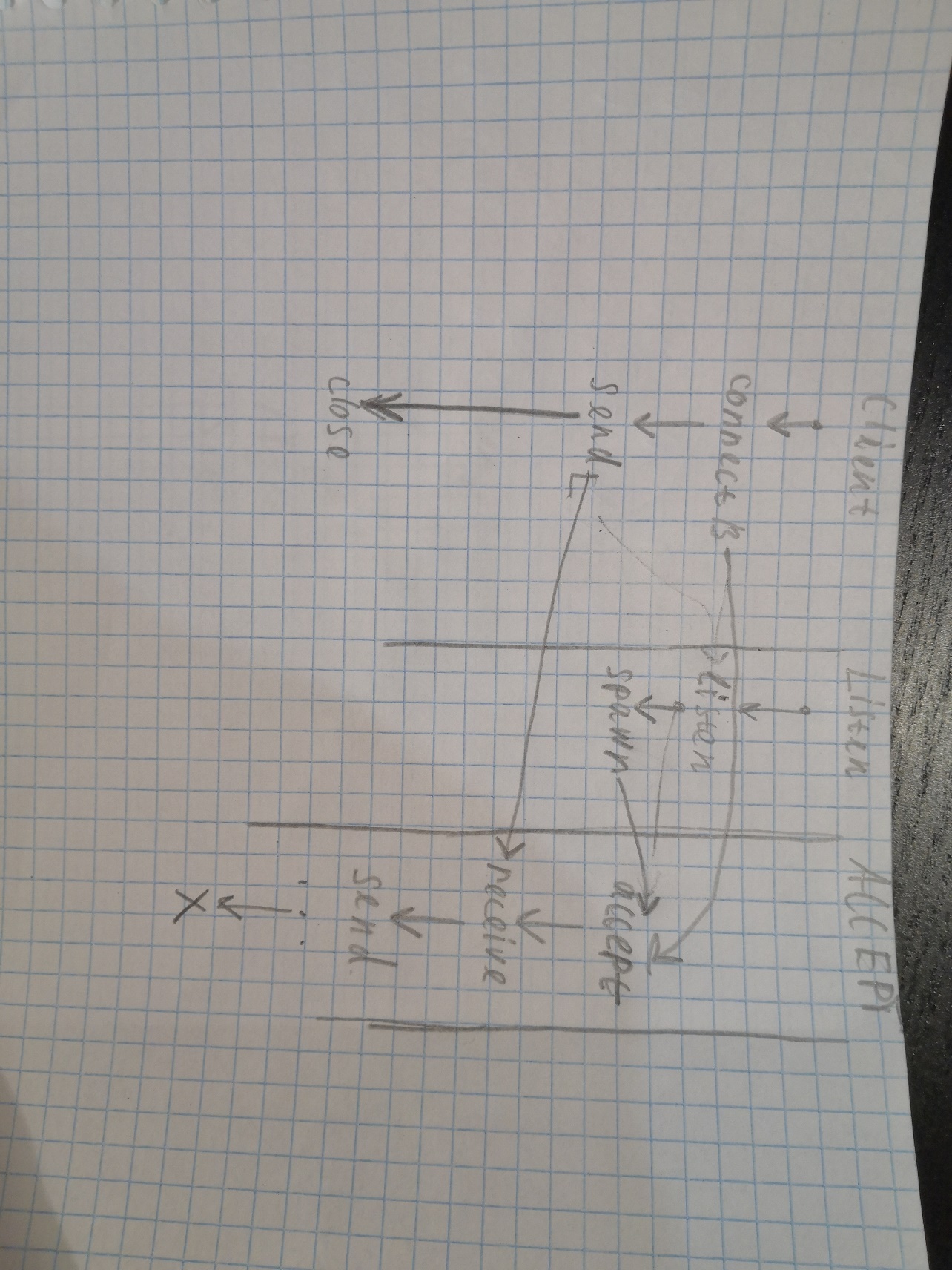




|  |  |  |  |
| --- | --- | --- | --- |
| **Вход функции** | **Выход функции** | **Что делает** | **Types** |
| **open(Port)** | **{ok, Socket} | {error, Reason}** | Открытие udp-сокета. Полученный на выходе Socket используется для отправки дейтаграмм из Port |  |
| **open(Port, Opts)** | **{ok, Socket} | {error, Reason}** | Открытие udp-сокета с дополнительными опциями |  |
| **send(Socket, Host, Port, Packet)** | **ok | {error, Reason}** | Отправка Packet серверу |  |
| **close(Socket)** | **ok** | Закрыть сокет |  |
| **recv(Socket, Length)** | **{ok, RecvData} | {error, Reason}** | Получение данных в пассивном режиме | **RecvData = {Address, Port, Packet} |** |
| **recv(Socket, Length, Timeout)** | **{ok, RecvData} | {error, Reason}** |  | **{Address, Port, AncData, Packet}** |
|  |  |  | **Packet = string() | binary()** |

# TCP





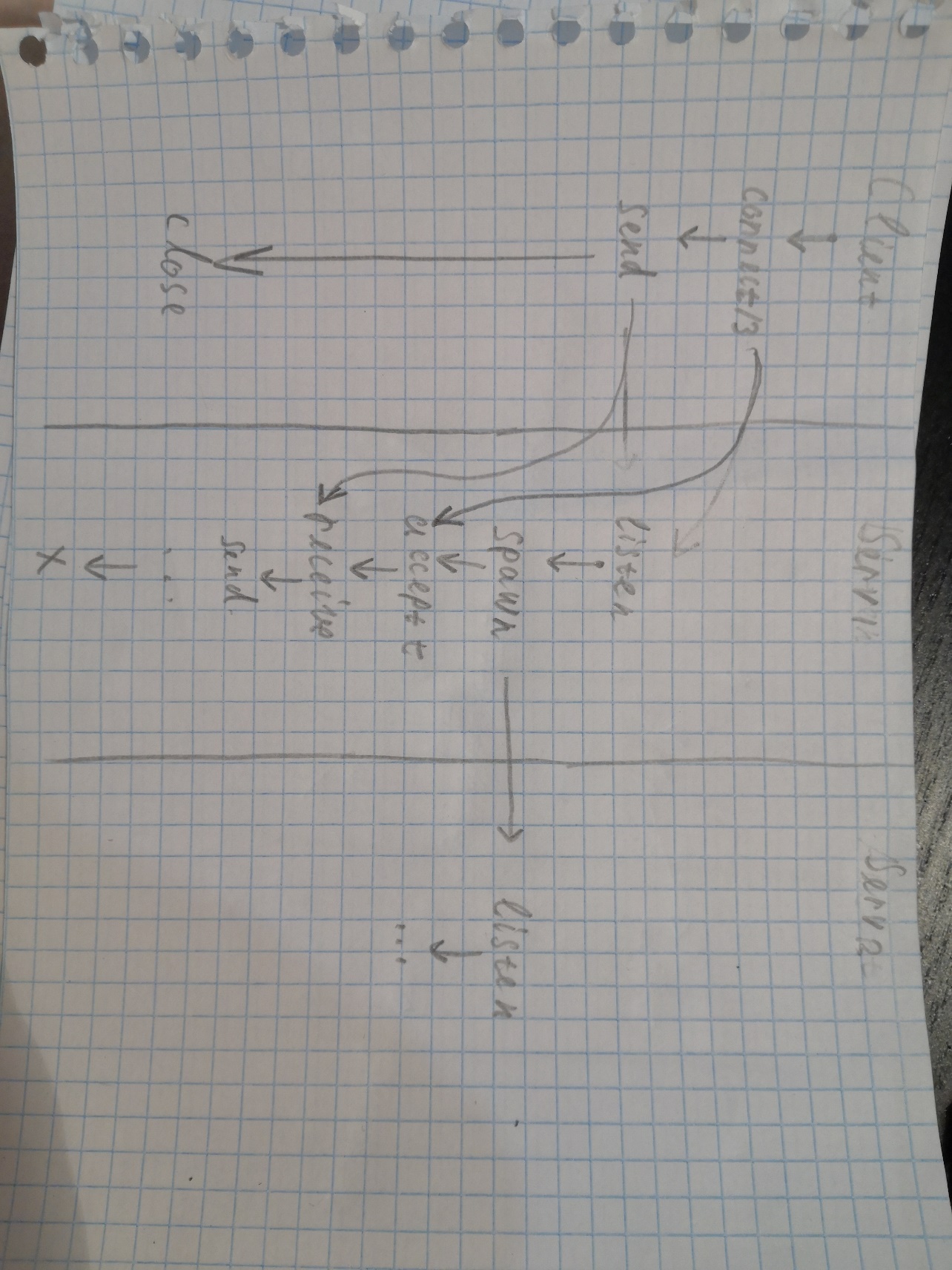


Table 1 — пример сервера, использующего несколько принимающих процессов для одного сокета

|  |  |  |  |
| --- | --- | --- | --- |
| **Вход функции** | **Выход функции** | **Что делает** | **Types** |
| **connect(SockAddr, Opts, Timeout)** | **{ok, Socket} | {error, Reason}** | Отправляет запрос на соединение с сервером | **inet:socket\_address() | inet:hostname()** |
|  |  |  | socket\_address = {127,0,0,1} |
|  |  |  | hostname = atom | string |
| **listen(Port, Options)** | **{ok, ListenSocket} | {error, Reason}** | Sets up a socket to listen on port Port on the local host. The returned socket ListenSocket should be used in calls to accept/1,2 to accept incoming connection requests. |  |
| **accept(ListenSocket)** | **{ok, Socket} | {error, Reason}** | Принимает входящий запрос на подключение от клиента. На вход подаётся ListenSocket, возвращаемый функцией listen. |  |
| **send(Socket, Packet)** | **ok | {error, Reason}** | Sends a packet on a socket. |  |

|  |
| --- |
| start(Num,LPort) ->  case gen\_tcp:listen(LPort,[{active, false},{packet,2}]) of  {ok, ListenSock} ->  start\_servers(Num,ListenSock),  {ok, Port} = inet:port(ListenSock),  Port;  {error,Reason} ->  {error,Reason}  end.  start\_servers(0,\_) ->  ok;  start\_servers(Num,LS) ->  spawn(?MODULE,server,[LS]),  start\_servers(Num-1,LS).  server(LS) ->  case gen\_tcp:accept(LS) of  {ok,S} ->  loop(S),  server(LS);  Other ->  io:format("accept returned ~w - goodbye!~n",[Other]),  ok  end.  loop(S) ->  inet:setopts(S,[{active,once}]),  receive  {tcp,S,Data} ->  Answer = process(Data), *% Not implemented in this example*  gen\_tcp:send(S,Answer),  loop(S);  {tcp\_closed,S} ->  io:format("Socket ~w closed [~w]~n",[S,self()]),  ok  end. |