 1

Trabalho
Chat UDP

PSI 2653
Meios Eletrônicos Interativos I

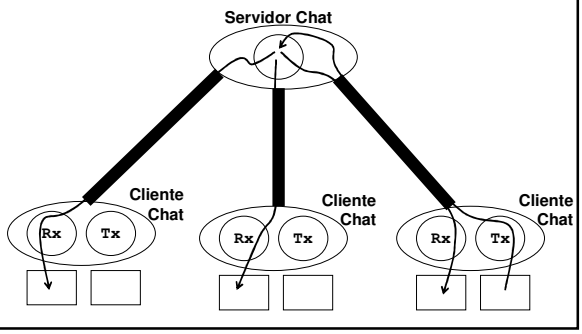
 2

Chat UDP

- **Objetivo:**
  - Desenvolvimento de um programa chat UDP (cliente e servidor)
- **Composição do grupo**
  - 2 pessoas
- **Formato do trabalho**
  - Papel A4, folhas grampeadas (não encadernar!!)
  - Página de rosto informando:
    - Nome da disciplina, título do trabalho e nome dos autores
- **Entrega:**
  - Data entrega: **2 de junho**
  - Entrega do trabalho escrito durante a aula
  - Execução do programa durante a aula
  - Serão descontados 2 pontos da nota para cada dia de aula em atraso

 3

Chat UDP



 4

Chat UDP

- **Servidor CHAT UDP**
  - Deve aguardar requisições na porta 10.000
  - Deve permitir sessões de chat com até 3 usuários (3 clientes chat) simultaneamente
  - Deve apresentar a mensagem "Número de usuários excedido" quando exceder a capacidade de 3 usuários de chat (3 clientes chat)
  - Deve guardar o "socketaddress" do cliente quando receber mensagem de pedido de login (USER)
  - Deve guardar o nome do usuário
  - Deve verificar, para cada mensagem recebida, o "socketaddress" de origem.
  - Deve, a cada 30s, encaminhar mensagem de teste a cada cliente com a finalidade de verificar se ainda está ativo. Caso duas mensagens de teste não sejam respondidas, deve realizar seu "logout"

 5

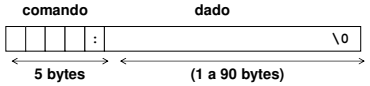
Chat UDP

- **Cliente CHAT UDP**
  - Deve enviar datagramas UDP para a porta 10.000 do servidor chat
  - Deve possuir dois threads:
    - Thread transmissor:
      - Obtém a mensagem do usuário e a transmite ao servidor
    - Thread receptor:
      - Aguarda mensagens do servidor e apresenta no terminal
  - A tela de recepção deve apresentar as mensagens para o usuário da seguinte forma:
    - Maria > Olá a todos
    - Ricardo> Olá Maria
    - Jose > Olá Maria, seja bem vinda.

 6

Chat UDP

- **Formato das mensagens**
  - A mensagem utilizada na comunicação entre o cliente e servidor devem ser codificadas em ASCII e devem possuir duas partes:
    - Comando: tamanho de 5 caracteres
    - Dado: tamanho variável, de 1 a 90 bytes (incluindo caractere '\0')

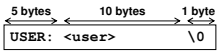


 7

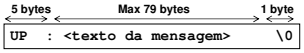
Chat UDP

- **Cliente: Mensagens do cliente para servidor:**

Login



Envio de mensagem



Pedido de encerramento

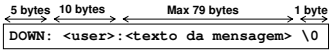
Resposta de teste

 8

Chat UDP

- **Servidor: Mensagens do servidor para cliente**

Mensagem para cliente



Teste se cliente

Resposta de login: ok

Resposta de login: ocupado

Resposta a encerramento

 9

Chat UDP

- **Servidor: Resposta aos comandos do cliente**
  - Resposta à mensagem USER de login:
    - Com disponibilidade: OKOK
    - Sem disponibilidade: BUSY
  - Resposta à mensagem UP:
    - Sem resposta
  - Resposta à mensagem EXIT:
    - BYE

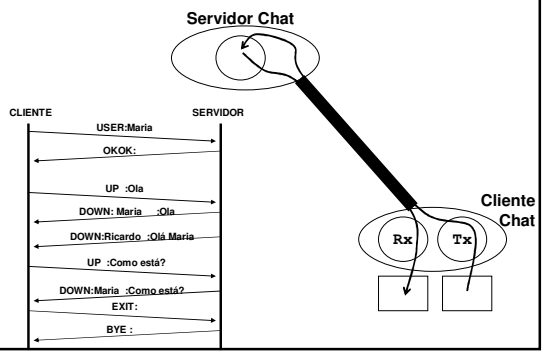
 10

Chat UDP

- **Cliente: Resposta aos comandos do servidor**
  - Resposta à mensagem DOWN:
    - Sem resposta
  - Resposta à mensagem TEST:
    - OKOK

 11

Chat UDP



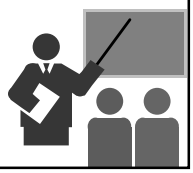
 12

Chat UDP

- **Ambiente e linguagem**
  - Ambiente Linux
  - Linguagem C
  - Biblioteca pthreads
  - Interface sockets

 13

Dicas



 14

Dicas

- **Desenvolva seu projeto em etapas:**
  - 1ª etapa: Cliente e servidor ECHO
    - Servidor UDP: servidor ECHO
    - Cliente UDP: cliente ECHO com dois threads (transmissor e receptor)
  - 1ª etapa: Implementação do protocolo
    - Servidor UDP: atendendo somente um cliente
    - Cliente UDP: com dois threads (transmissor e receptor)
  - 2ª etapa: Final
    - Servidor UDP: atendendo até 3 clientes.

 15

Dicas

- **Manter tabelas de controle no servidor:**
  - Manter uma tabela de controle com uma entrada para cada cliente.
  - Para cada cliente armazenar, no mínimo:
    - "username"
    - "socket address" do cliente

 16

Dicas

- **Janelas**
  - Em um chat, conforme são digitadas as mensagens, são também recebidas outras mensagens, de forma concorrente.
  - Nesta situação, caso seja utilizada somente uma janela para apresentação das mensagens transmitidas (digitadas) e das mensagens recebidas, tais mensagens poderão ficar intercaladas, tornando muito confuso para o usuário.
  - Assim, devem ser utilizadas duas janelas, uma para digitar as mensagens a serem enviadas e uma outra na qual são apresentadas as mensagens recebidas dos usuários.

 17

Dicas

- **Dicas para utilização de duas janelas:**
  - Comando para identificação do terminal corrente: "tty"
  - Trecho de código para enviar mensagens de texto para outro terminal:

```
char   terminalname[80];
FILE * terminal;

...
printf("Entre com o nome do terminal auxliar ao chat: ");
scanf("%s",terminalname);
terminal = fopen(terminalname,"a+");
if (terminal == NULL)
    {
    perror("Abertura do terminal");
    exit(1);
    }
....
fprintf(terminal, "teste de terminal \n");
....
```

 18

Dicas

- **Funções para desenvolvimento**
  - Utilizar fgets() ao invés de scanf()
    - #include <stdio.h>
    - char *fgets (char *string, int size, FILE *stream);
    - Evita problemas de overflow do buffer, pois gets() permite definir o tamanho do buffer.
    - A função fgets() lê caracteres até encontrar newline ou chegar ao tamanho do buffer. O newline é acrescentado à string. O caracter '\0' é acrescentado ao final.
  - Exemplo:

```
#include <stdio.h>
char   buffer[80];
fgets(buffer,80,stdin);
buffer[strlen(buffer)-1]='\0';  // retira \n
```