



Trace: - [trab1.2](#)

cursos:icsr30:trab1.2

## Trabalho com conexão TCP - Parte 2.

HTTP/TCP - Servidor Web Multithreads

### Table of Contents

- Trabalho com conexão TCP - Parte 2.
  - HTTP/TCP - Servidor Web Multithreads
- O Protocolo de Controle de Transmissão - TCP
  - Fluxo do trabalho:
  - O trabalho deve:

### O Protocolo de Controle de Transmissão - TCP

Neste trabalho iremos continuar explorando a implementação de uma aplicação rodando sobre *TCP* através da programação com *sockets*. Este trabalho tem a finalidade de trazer o conhecimento de programação e funcionamento básico do protocolo *TCP*, principalmente demonstrando os serviços que o *TCP* fornece para a camada de aplicação. Baseado no primeiro trabalho, mas agora transformando o anterior em um **Servidor HTTP simplificado**.

#### Fluxo do trabalho:

- Procurar um código "Hello word" usando servidor TCP multi thread e seu cliente.
  - Este trabalho pode ser realizado em qualquer linguagem de programação, a escolha do aluno, mas lembre-se: não pode ser usado bibliotecas que manipulem o TCP, e sim usar o TCP diretamente através da criação e manipulação dos sockets.
- No servidor TCP (deve executar antes do cliente)
  - Escolher um porta para receber as conexões (maior que 1024)
  - Aceitar a conexão do cliente
  - Criar uma thread com a conexão do cliente (para cada cliente). Na thread:
    - Receber dados recebidos pelo cliente
    - Tratar esses dados (requisição HTTP)
      - Ex.: GET /pagina.html HTTP/1.0
- No Cliente TCP (deve executar depois do servidor)
  - Usar o Browser de sua preferência
  - Colocar o endereço da máquina e porta escolhida para o servidor
    - URL : @ip do servidor:(Porta servidor)/pagina.html
  - O Browser deve apresentar o arquivo requisitado na URL
    - O Browser deve mostrar ao menos arquivos HTML + JPEG
    - O Browser deve interpretar ERROS.
      - Ex.: Resposta com 404.

#### O trabalho deve:

- Usar Sockets TCP Multi-thread
  - Servidor
- No Servidor (Nesta Fase não é necessário implementar o cliente, pois será usado um Browser como cliente.)
  - Receber requisições do Cliente
  - Tratar corretamente as requisições HTMP e fazer o esperado.
- O Browser deve funcionar apresentando o arquivo requisitado na URL
  - O Browser deve mostrar ao menos arquivos HTML + JPEG
  - O Browser deve interpretar ERROS.
    - Ex.: Resposta com 404.



Lembre-se: O vídeo deve ser uma demonstração prática, complementada por explicações claras sobre o funcionamento e as decisões de projeto.



O vídeo **deve** ser anexado como **link** nos comentários da entrega dentro da atividade do classroom. **NÃO** deve ser enviado o arquivo do vídeo!

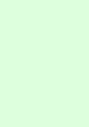


Veja o exemplo de uma página HTML simples:

```
<HTML>
<HEAD>
  <TITLE>Título da página</TITLE>
</HEAD>

<BODY>
  Conteúdo da página
</BODY>
</HTML>
```

### Requisições HTTP



Veja o exemplo de uma requisição HTTP simples:

```
GET /pagina.html HTTP/1.0
Host: www.UTFPR.edu.br
Accept: text/plain; text/html
Accept-Language: en-gb
Connection: Keep-Alive
Host: localhost
Referer: http://localhost/ch8/SendDetails.htm
User-Agent: Mozilla/4.0 (compatible; MSIE 4.01; Windows 98)
Content-Length: 33
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
```



Resposta HTTP:

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/4.0
Date: Mon, 3 Jan 2016 17:13:34 GMT
Content-Type: text/html
Last-Modified: Mon, 11 Jan 2016 17:24:42 GMT
Content-Length: 112
```

```
<html>
<head>
<title>Exemplo de resposta HTTP </title>
</head>
<body>
Acesso não autorizado!
</body>
</html>
```



- O aluno deve usar as chamadas TCP e não pode usar bibliotecas que mascarem o trabalho.



- Para agilizar a verificação de integridade são utilizadas somas de verificação (checksums) ou resumos criptográficos como o MD5 e SHA.