Desenvolvimento de Aplicações Java

Plataforma Corporativa

Sockets

Agosto 2015

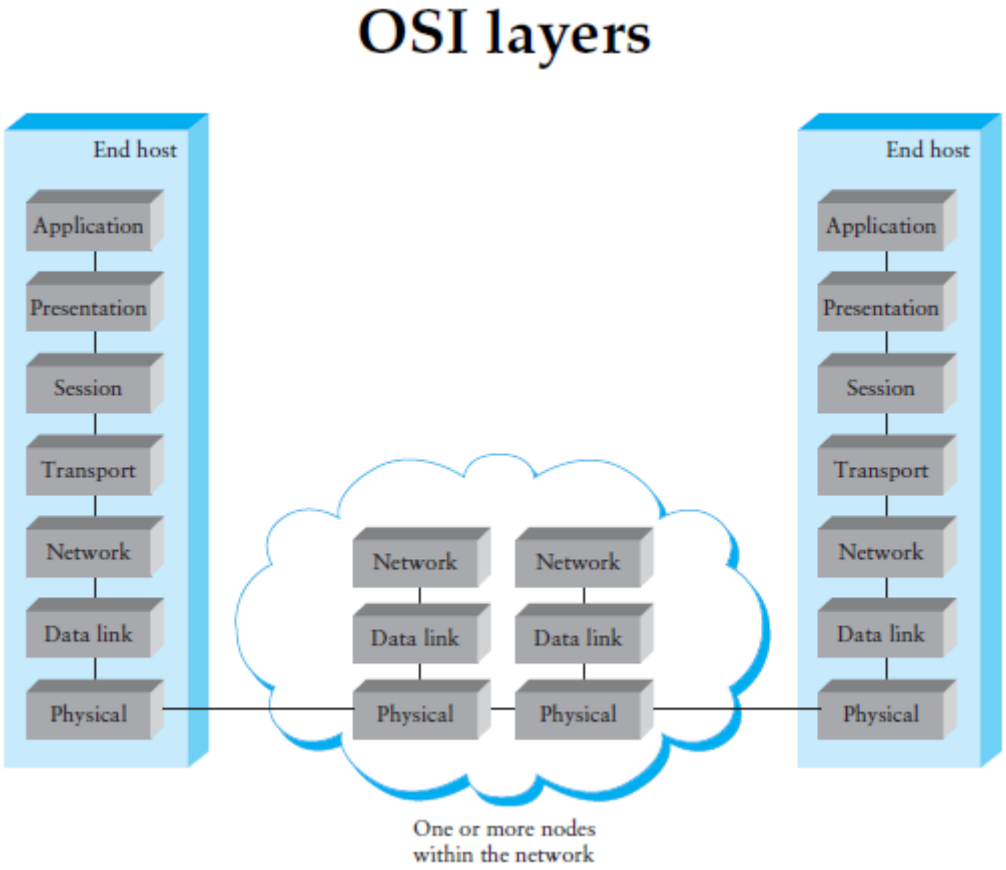
Sumário

1.Introdução

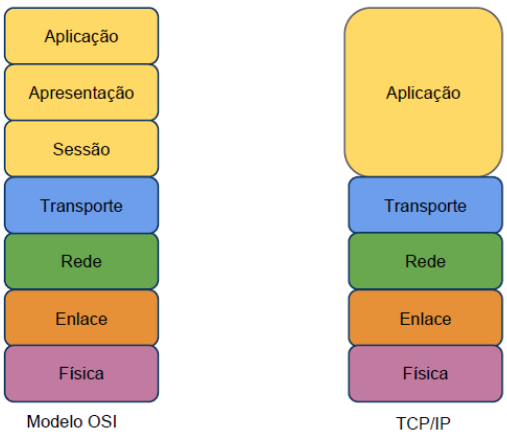
Este documento provê uma breve introdução à conexões socket.

2. Contexto

2.1. Layers



- Physical – sends individual bits
- Data link – sends frames, handles access control to shared media (e.g., coax, twisted pair, etc…)
- Network – delivers packets, using routing
- Transport – demultiplexes, provides reliability & flow control
- Session – can tie together multiple streams (e.g.,audio & video)
- Presentation – crypto, conversion between representations
- Application – what end user gets, e.g., HTTP (web)



2.2. Addressing

• Each node typically has unique address

- (or at least is made to think it does when there is shortage)

• Each layer can have its own addressing

- Link layer: e.g., 48-bit Ethernet address (interface)

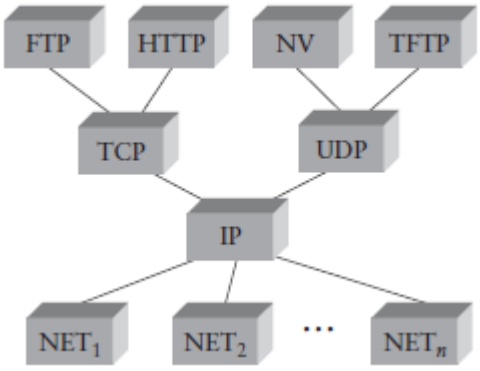
- Network layer: 32-bit IP address (node)

- Transport layer: 16-bit TCP port (service)

• Routing is process of delivering data to destination across multiple link hops

• Special addresses can exist for broadcast/multicast

2.3. Protocol Layers



Many application protocols over TCP & UDP

• IP works over many types of network

• This is “Hourglass” philosophy of Internet

- Idea: If everybody just supports IP, can use many different applications over many different networks

- In practice, some claim narrow waist is now network and transport layers, due to NAT

2.4. Internet protocol

• Most computer nets connected by Internet protocol

- Runs over a variety of physical networks, so can connect Ethernet, Wireless, people behind modem lines, etc.

• Every host has a unique 4-byte IP address

- E.g., www.ietf.org! 132.151.6.21

- Given a node’s IP address, the network knows how to route a packet (lectures 3+4)

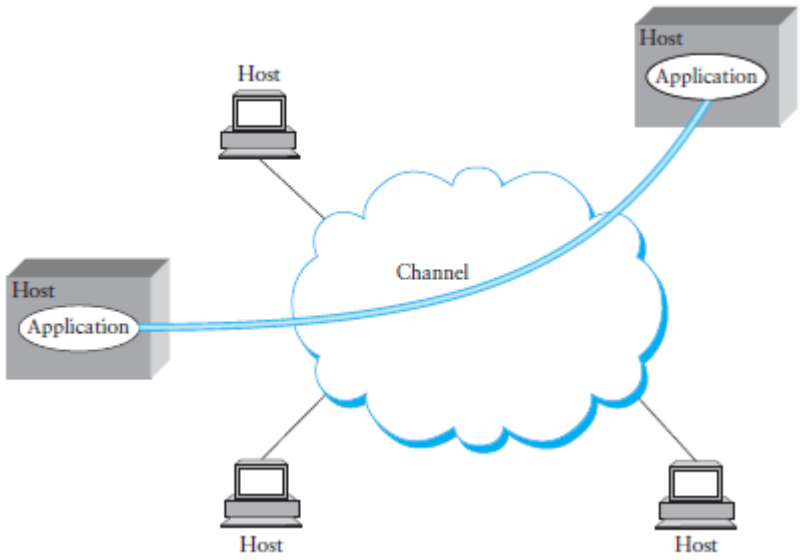
• But how do you build something like the web?

- Need naming (look up www.ietf.org) – DNS

- Need interface for browser & server software

- Need demultiplexing within a host—E.g., which packets are for web server, which for mail server, etc.?

2.5. Inter-process communication



• Want abstraction of inter-process (not just inter-node) communication

• Solution: Encapsulate another protocol within IP

2.6. UDP and TCP

• UDP and TCP most popular protocols on IP

- Both use 16-bit port number as well as 32-bit IP address

- Applications bind a port & receive traffic to that port

• UDP – unreliable datagram protocol

- Exposes packet-switched nature of Internet

- Sent packets may be dropped, reordered, even duplicated (but generally not corrupted)

• TCP – transmission control protocol

- Provides illusion of a reliable “pipe” between to processes on two different machines

- Handles congestion & flow control

2.7. Uses of TCP

• Most applications use TCP

- Easier interface to program to (reliability)

- Automatically avoids congestion

• Servers typically listen on well-known ports

- SSH: 22

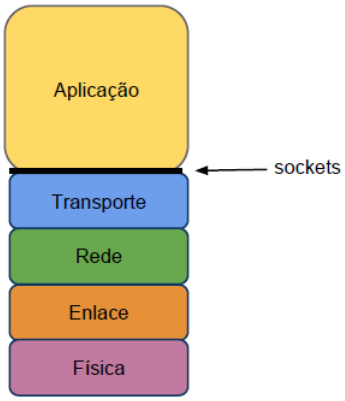
- Email: 25

- Finger: 79

- Web / HTTP: 80

2.8. Sockets

O Socket (Berkley Unix) é um protocolo independente para comunicação entre processos (IPC) .



Os sockets podem ser:

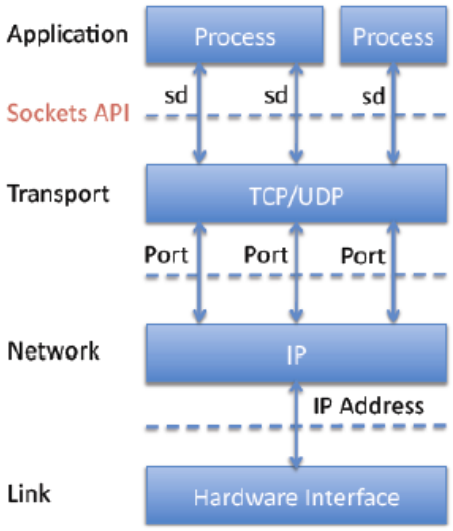
- baseados em conexões ou sem conexão: Existe uma conexão estabelecida antes da comunicação ou cada pacote indica o seu destino?
- baseados em pacotes ou streams: As mensagens têm um tamanho ou são um fluxo contínuo de dados?
- com ou sem garantia de entrega: As mensagens podem ser perdidas, duplicadas, reordenadas ou corrompidas?

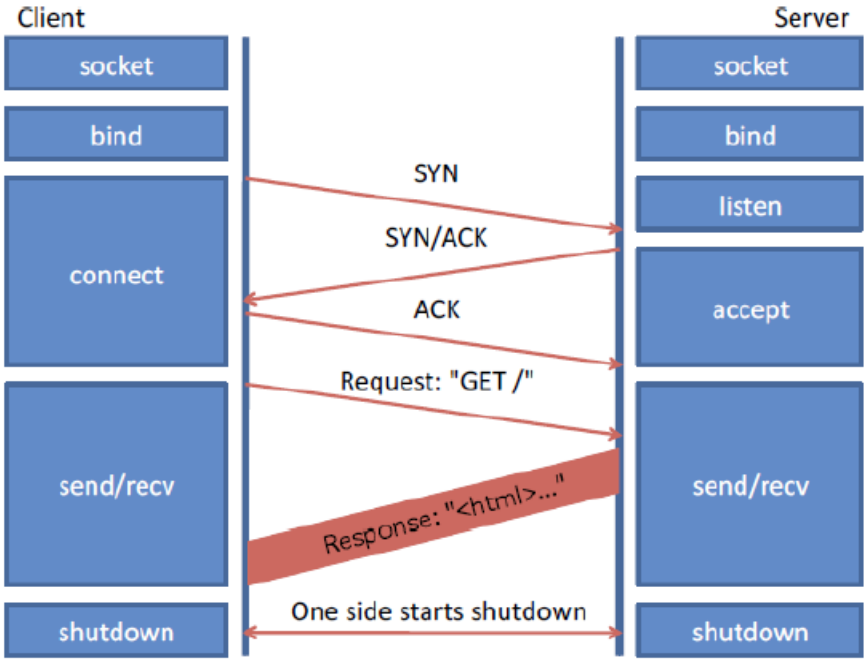
3. Modelo Cliente-servidor

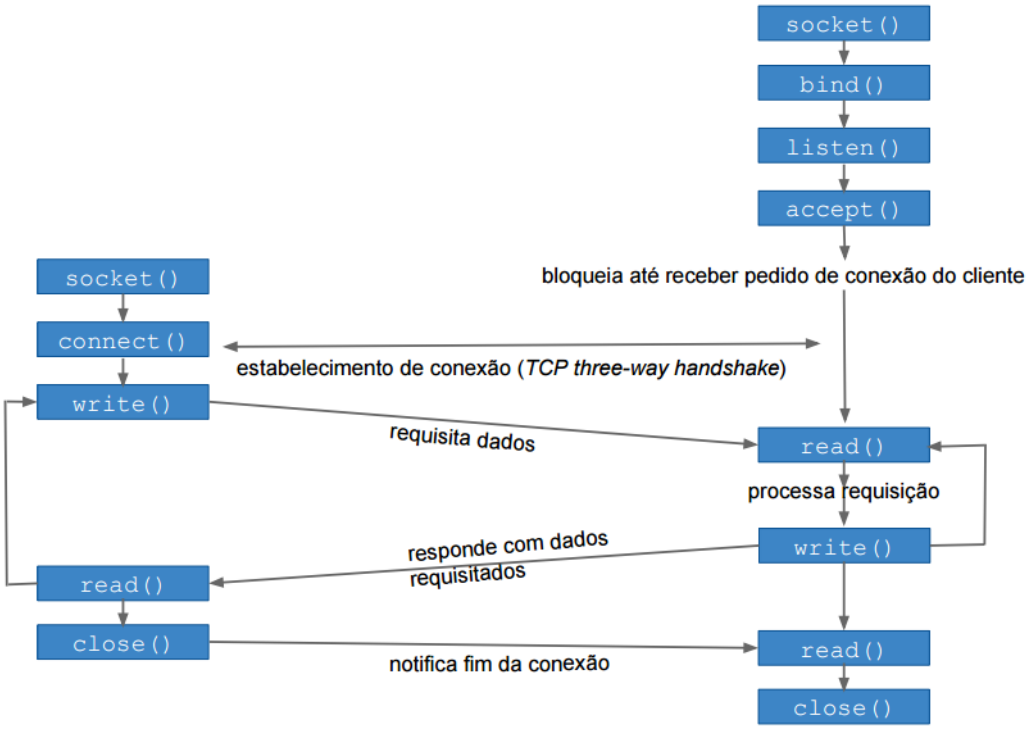
- Rendezvous problem
- Modelo: para qualquer par de aplicações que se comunicam, um dos lados deve iniciar a execução e esperar (indefinidamente) até ser contactado pelo outro lado.
- Uma aplicação que inicia uma comunicação par-a-par é chamada cliente.
- Um servidor é um programa que espera por requisições de um cliente.

TCP/IP

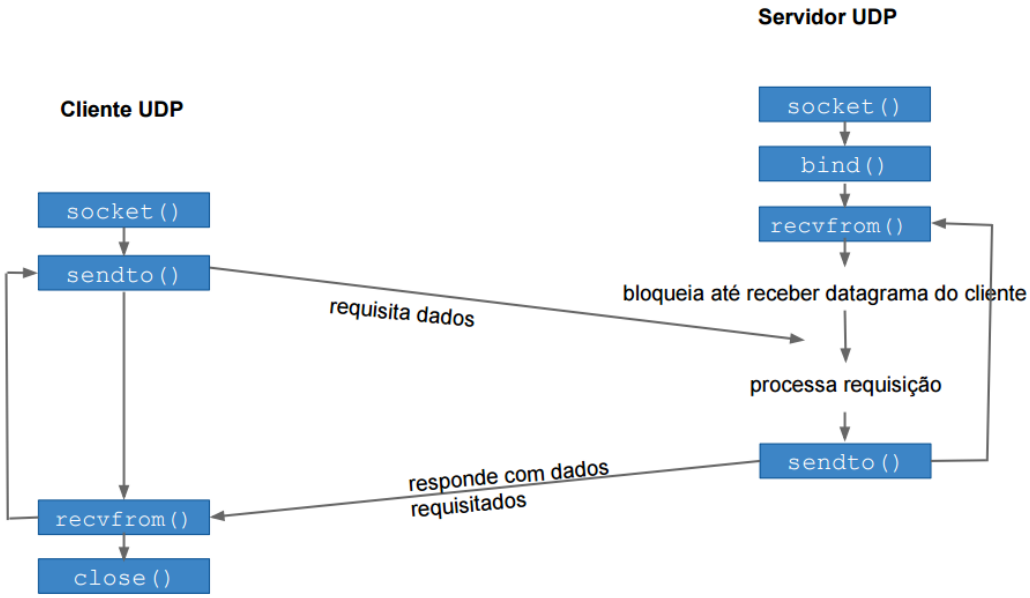
- cada ponto final é identificado por uma tupla: (porta TCP, endereço IP)
- a conexão entre dois pontos finais é identificada pelo par [(IP, porta)origem, (IP, porta)destino]


Application
Process
Process
sd
sd
sd
Sockets API
Transport
TCP/UDP
Port
Port
Port
Network
IP
IP Address
Link
Hardware Interface


Client
Server
socket
bind
connect
send/recv
shutdown
socket
bind
listen
accept
send/recv
shutdown
SYN
SYN/ACK
ACK
Request: "GET /"
Response: "<html>..."
One side starts shutdown



4. Modelo P2P (processos simétricos)



5. Atividade 1

Implementação de um cliente para um servidor HTTP utilizando socket.

- Crie um projeto Java.
- Implemente a classe ClienteHTTP.

```
package br.com.atech.socket;


import java.io.FileOutputStream;

import java.io.InputStream;

import java.io.OutputStream;

import java.io.OutputStreamWriter;

import java.io.PrintWriter;

import java.net.Socket;

import java.net.URL;


public class ClienteHTTP {


    public static void main(String[] args){

    try {


      OutputStream to_file;

      //to_file = new FileOutputStream("output.html");

      to_file = System.out;


      URL url = new URL("http://www.atech.com.br/Clientes.html");

      String protocol = url.getProtocol();

      if (!protocol.equals("http"))

        throw new IllegalArgumentException("URL must use 'http:' protocol");

      String host = url.getHost();

      int port = url.getPort();

      if (port == -1) port = 80;

      String filename = url.getFile();

      System.out.println(host + ":" + port + ":" + filename);

      Socket socket = new Socket(host, port);

      InputStream from_server = socket.getInputStream();

      PrintWriter to_server = new PrintWriter(new
OutputStreamWriter(socket.getOutputStream()));


      to_server.print("GET "+filename + " HTTP/1.1\r\n");

       to_server.print("Host: " + host + "\r\n");
```

```
      to_server.print("\r\n");
      to_server.flush();

      byte[] buffer = new byte[4096];
      int bytes_read;
      while((bytes_read = from_server.read(buffer)) != -1)
        to_file.write(buffer, 0, bytes_read);

      socket.close();
      to_file.close();
    }
    catch (Exception e) {
      System.err.println(e);
    }
  }
}
```

6. Atividade 2

Implementação de um servidor HTTP utilizando socket.

- Crie um projeto Java.
- Implemente a classe ServidorHTTP.
- Modifique a classe ClienteHTTP para acessar este servidor.
- Crie uma pasta /www
- Coloque nesta pasta o arquivo output.html
- Requisitar o recurso através do browser (http://localhost:8080/www/output.html

```
package br.com.atech.socket;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.io.InputStreamReader;
```

```
import java.io.OutputStreamWriter;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;

public class ServidorHTTP {

    public static void main(String[] args) throws Exception {

        int port = 8080;
        ServerSocket serverSocket = new ServerSocket(port);
        Socket clientSocket = serverSocket.accept();

        BufferedReader in = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));
        PrintWriter out = new PrintWriter(new BufferedWriter(new
OutputStreamWriter(clientSocket.getOutputStream())), true);
        String s;
        byte[] buffer = new byte[4096];
        int bytes_read;

        StringBuilder request = new StringBuilder();
        String httpCommand = "";
        String resource = "";

        while ((s = in.readLine()) != null) {

            if (s.contains("HTTP")) {
                String[] tokens = s.split(" ");
                httpCommand = tokens[0];
                resource = tokens[1];
                System.out.println("HTTP Command=" + httpCommand);
                System.out.println("REQUESTED RESOURCE=" + resource);
            }
            if (httpCommand.equals("GET")) {
                httpCommand = "";
```

```
                String content = null;
                File file = new File(resource);
                FileReader reader = null;
                int fileLength = 0 ;
                try {
                    reader = new FileReader(file);
                    fileLength = (int) file.length();
                    char[] chars = new char[fileLength];
                    reader.read(chars);
                    content = new String(chars);
                    reader.close();
                } catch (IOException e) {
                    e.printStackTrace();
                } finally {
                    if (reader != null) {
                        reader.close();
                    }
                }
                out.write("HTTP/1.1 200 OK\r\n");
                out.write("Date: Tue, 01 Sep 2015 18:45:38 GMT\r\n");
                out.write("Server: Apache\r\n");
                out.write("Expires: -1\r\n");
                out.write("Cache-Control: no-store, no-cache, must-revalidate, post-check=0,
pre-check=0\r\n");
                out.write("Pragma: no-cache\r\n");
                out.write("Content-Type: text/html\r\n");
                out.write("Content-Length: " + fileLength+ "\r\n");
                out.write("Last-modified: Fri, 09 Aug 1996 14:21:40 GMT\r\n");
                out.write("Connection: keep-alive\r\n");
                out.write("Accept-Ranges: bytes\r\n");
                out.write("\r\n");
                out.flush();
                out.write(content);
                out.flush();
            }
```

```
        }

        out.close();

        in.close();

        clientSocket.close();
    }
}
```

output.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
        <html xmlns="http://www.w3.org/1999/xhtml">
        <head>
                <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
                <meta http-equiv="Expires" content="-1" />
                <meta http-equiv="Pragma" content="no-cache" />
                <meta http-equiv="Cache-Control" content="no-store, no-cache, must-
revalidate" />
                <meta name="Description" content="FAB - Fora></li>
<li><a href='Portfolio.html'>Portfólio</a></li>
<li><a href='Absorcao-de-Tecnologias.html'>Absorção de Tecnologias</a></li>
                                                        </ul>
                                                </div>
                                        </div>
                                        <div  class='special-link' style='left:193px;'><a
href="#" style="border-color:#80CDD1;">Quem somos</a>
                                                <div class='submenu-special' style="background-
color:#019BA3;">
                                                        <ul>
                                                                <li><a href='Missao%2C-Visao-e-
Valores.html'>Missão, Visão e Valores</a></li>
<li><a href='A-Atech.html'>A Atech</a></li>
```

```
                                        </ul>

                                    </div>

                                </div>



                            </div>

                            <div class='int-wrap'>

                            <div class='left-wrap'>


                                    <ul class='left_menu'><li><a class='itemmenu selected'
id='supermenu_2' href='Clientes.html'>Clientes</a></li>

<li><a class='itemmenu' id='supermenu_4' href='Noticias.html'>Notícias</a></li>

<li><a class='itemmenu' id='supermenu_5' href='Desafio-profissional.html'>Desafio
profissional</a></li>

<ul id='left_menu' style='display:none'><li><a  href='Envio-de-curr-iacuteculo.html'>Envio de
curr&iacuteculo</a></li>

<li><a  href='Oportunidades.html'>Oportunidades</a></li>

</ul><li><a class='itemmenu' id='supermenu_6' href='Fale-Conosco.html'>Fale
Conosco</a></li>

<li><a class='itemmenu' id='supermenu_7' href='Politicas-Corporativas.html'>Pol□cas
Corporativas</a></li>

</ul>                                   </div>

                                    <div id="general-content" class="internal">

                                        <h1><span id='subtitulo'></span></h1>

                                        <span id='intro'></span>?<div id='bloco63'
class='bloco'><div class='texto'><div style="font-weight: bold;" id="contcont"><ul><li>FAB -
For衡 A 곭 a Brasileira<br /><br /></li><li>DECEA - Departamento de Controle do Espa 诠
A 곭 o<br /><br /></li><li>DEPENS - Departamento de Ensino da Aeron□ica<br /><br
/></li><li>CISCEA - Comiss髚de Implanta 褯 do Sistema de Controle do Espa 诠 A 곭 o<br
/><br /></li><li>PAME - Parque de Material Eletro<br /><br /></li><li>ICEA - Instituto de
Controle do Espa 诠 A 곭 o<br /><br /></li><li>EEAR - Escola de Especialistas da
Aeron□ica<br /><br /></li><li>AES Eletropaulo</li></ul></div></div></div>
                                </div>

                                    </div>

                                </div>

                        </div>

                    </div>

                    <div id="footer">
```

```
                        <!-- <a href="#" class="link_fundatech"><img
src="mestri/bancoarquivos/imagens/logo-fundacaoatech.gif" alt="Funda 褦 Atech" /></a> -->

                        <div class="footer-wrapper">

                            <div class="footer-up"><span class='address'>Rua do
Rocio, 313 - 2ξandar | Vila Ol□ia - S龒Paulo - SP | Tel.: 55 (11) 3103-4600</span> <a
href="mailto:contato@atech.com.br">contato@atech.com.br</a></p></div>

                            <div class="footer-down"><span>www.atech.com.br -
&copy; Atech Neg□s em Tecnologias S/A</span><a href="politica-de-privacidade.html"
class="politica_privacidade_link">Pol□ca de Privacidade</a><a href="http://www.unitri.com.br"
class="unitri_link">unitri</a></div>

                        </div>

                    </div></body></html>
```