PROTÓTIPO DE UM HARDWARE PARA CONTROLE DE FREQUÊNCIA ACADÊMICA

FURB – Fundação Universidade Regional de Blumenau

PROTÓTIPO DE UM HARDWARE PARA CONTROLE DE FREQUÊNCIA ACADÊMICA

Aluno: **Fernando Luiz Melati da Silva**

Orientador: **Miguel Alexandre Wisintainer**

ROTEIRO

- Objetivo
- Introdução
- TCP/IP
- SMTP – Simple Mail Transfer Protocol
- Kit Desenvolvimento RCM 2200
- Especificação
- Implementação
- Operacionalidade
- Conclusões
- Extensões

OBJETIVO

- O objetivo deste trabalho é desenvolver um software para um hardware, que controle a freqüência dos alunos em sala de aula, enviando um *e-mail* ao professor da mesma, com a lista dos alunos presentes.

INTRODUÇÃO

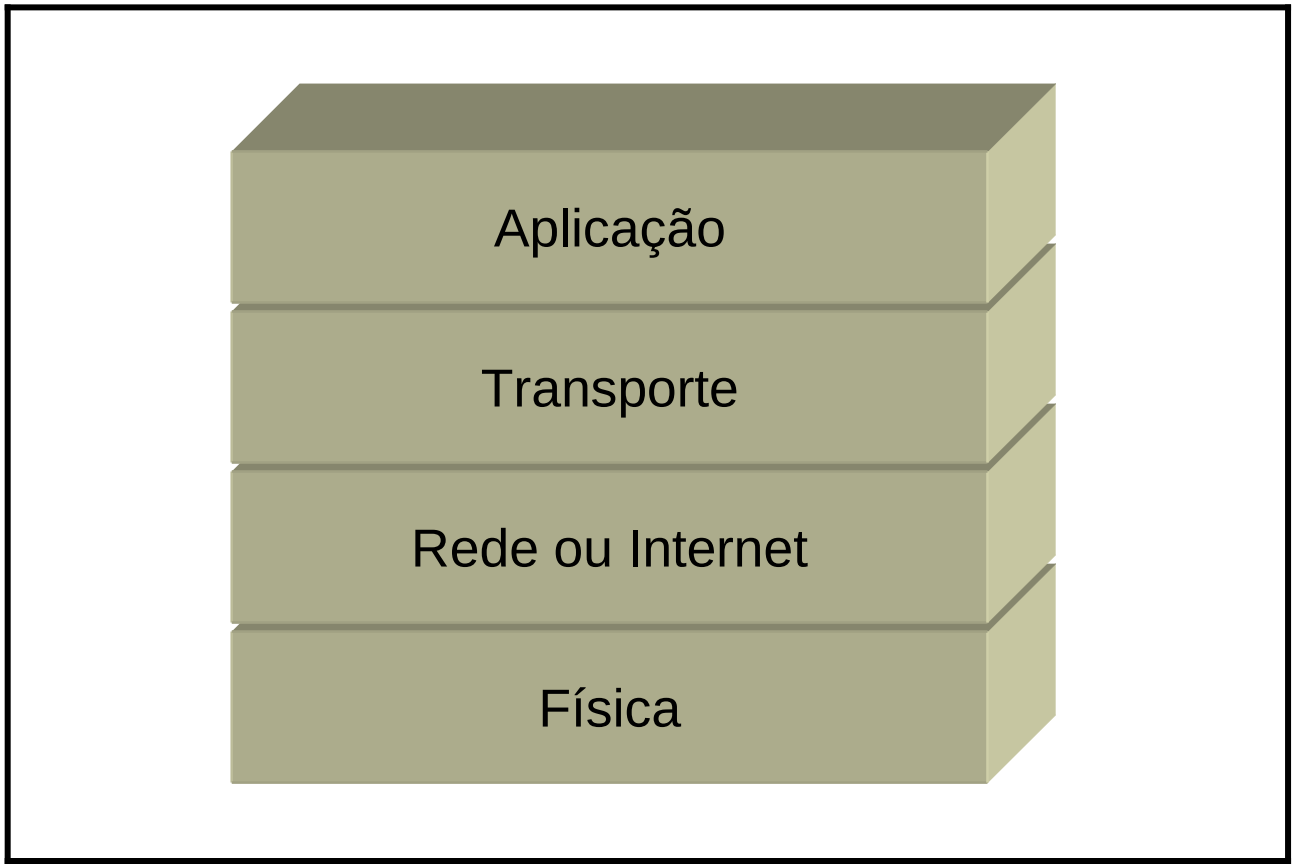
As universidades são centros de criação, transmissão e difusão da cultura, da ciência e da tecnologia que, através da articulação do estudo, da docência e da investigação, se integram na vida da sociedade (Marcovitch, 1998).

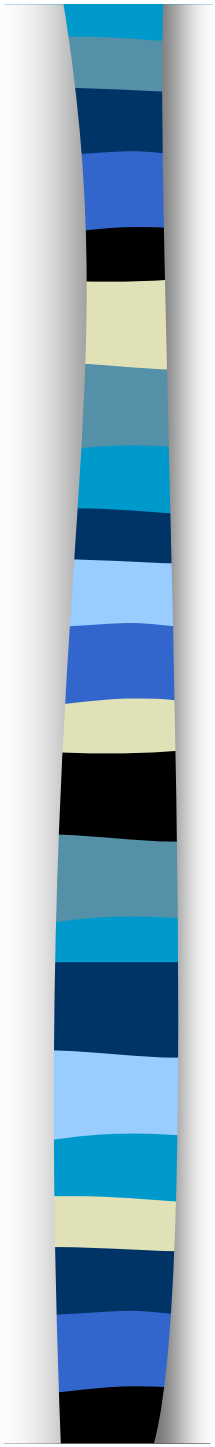
UNIVERSIDADES

- Formação de profissionais
- Verificação de aprendizagem
- 75% carga horária no mínimo
- Buscam aprimoramento tecnológico

TCP/IP

Camadas da arquitetura TCP/IP





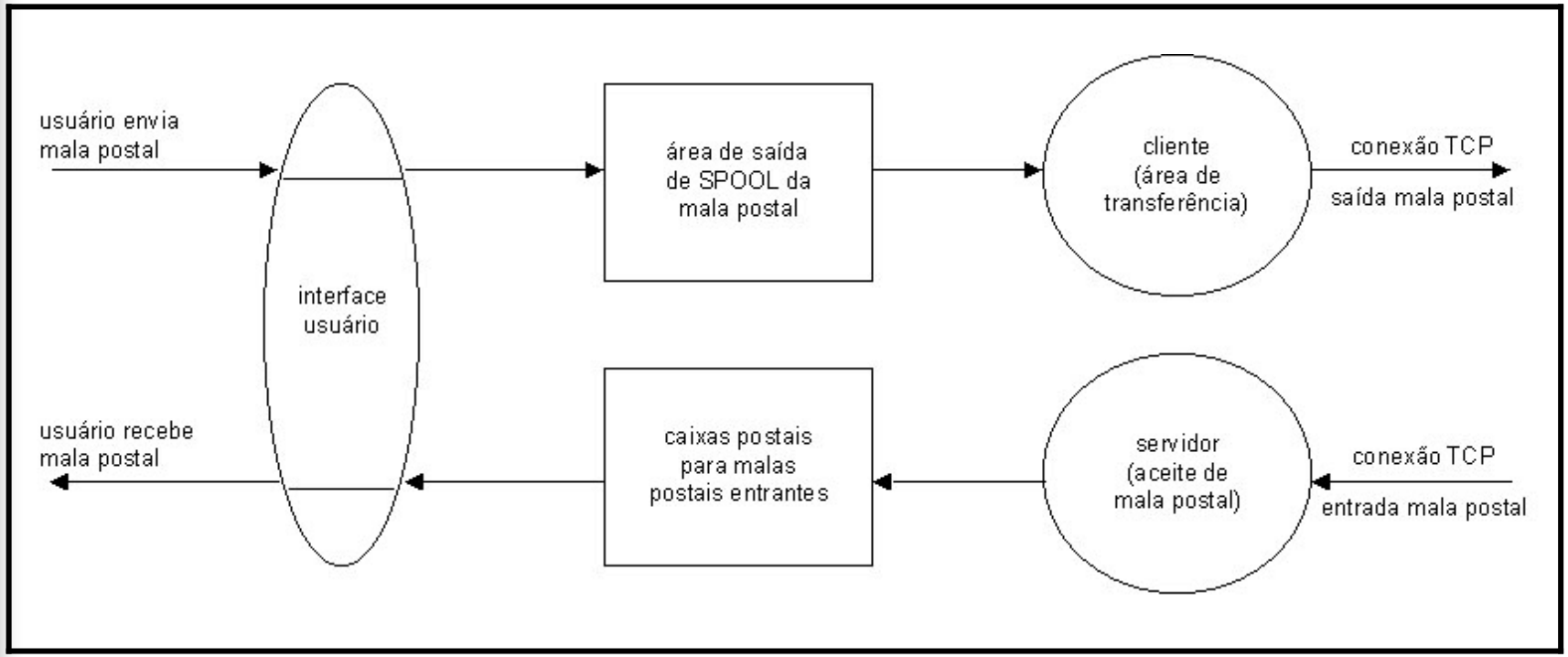
TCP/IP – camada de transporte

- TCP – Transmission Control Protocol
- UDP – User Datagram Protocol
- Socket

TCP/IP – camada de aplicação

- Transferência de arquivos (FTP – *File Transfer Protocol*)
- Transferência de arquivos, documentos e aplicações (HTTP – *Hyper Text Transfer Protocol)*
- Emuladores de terminal (telnet)
- Correio eletrônico (SMTP – *Simple Mail Transfer Protocol*)

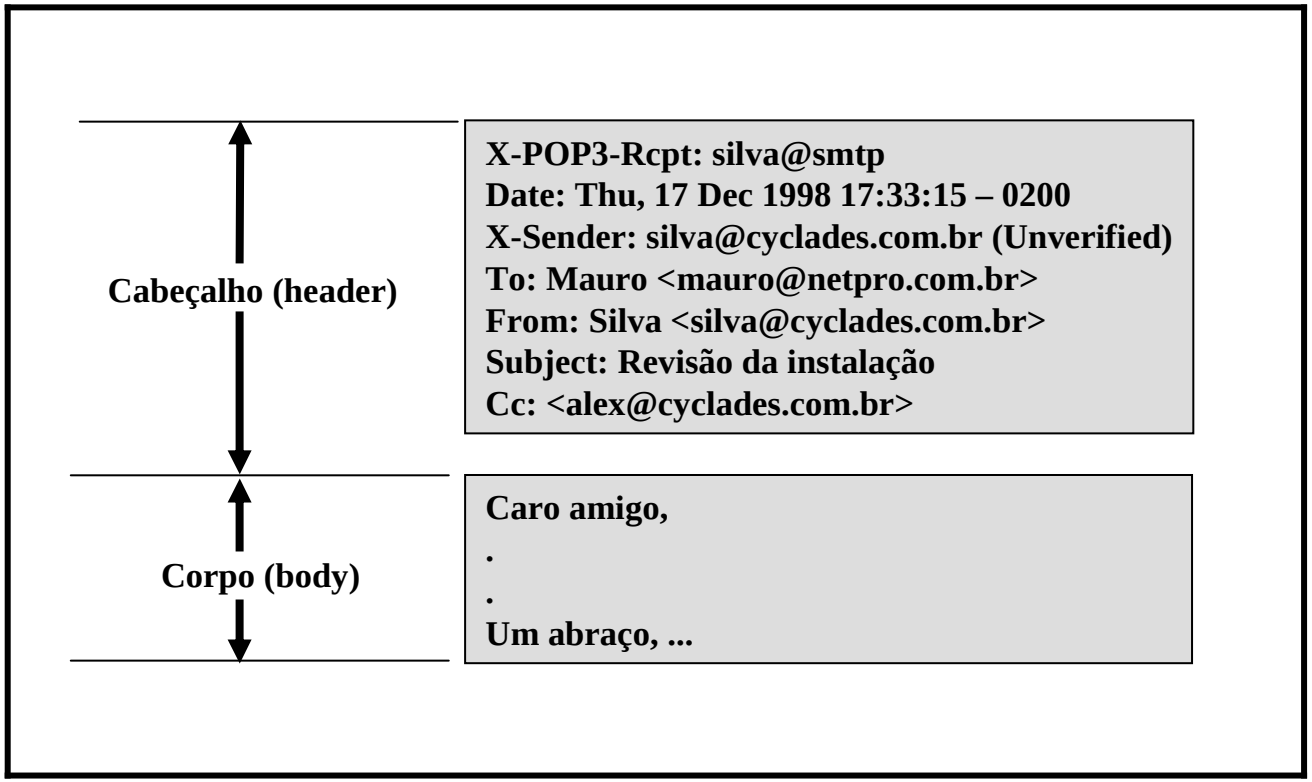
SMTP – simple mail transfer protocol



SMTP – formato do endereço

- Definido pela RFC 822
- *user@host*
- *user@domain* (utilizando DNS)

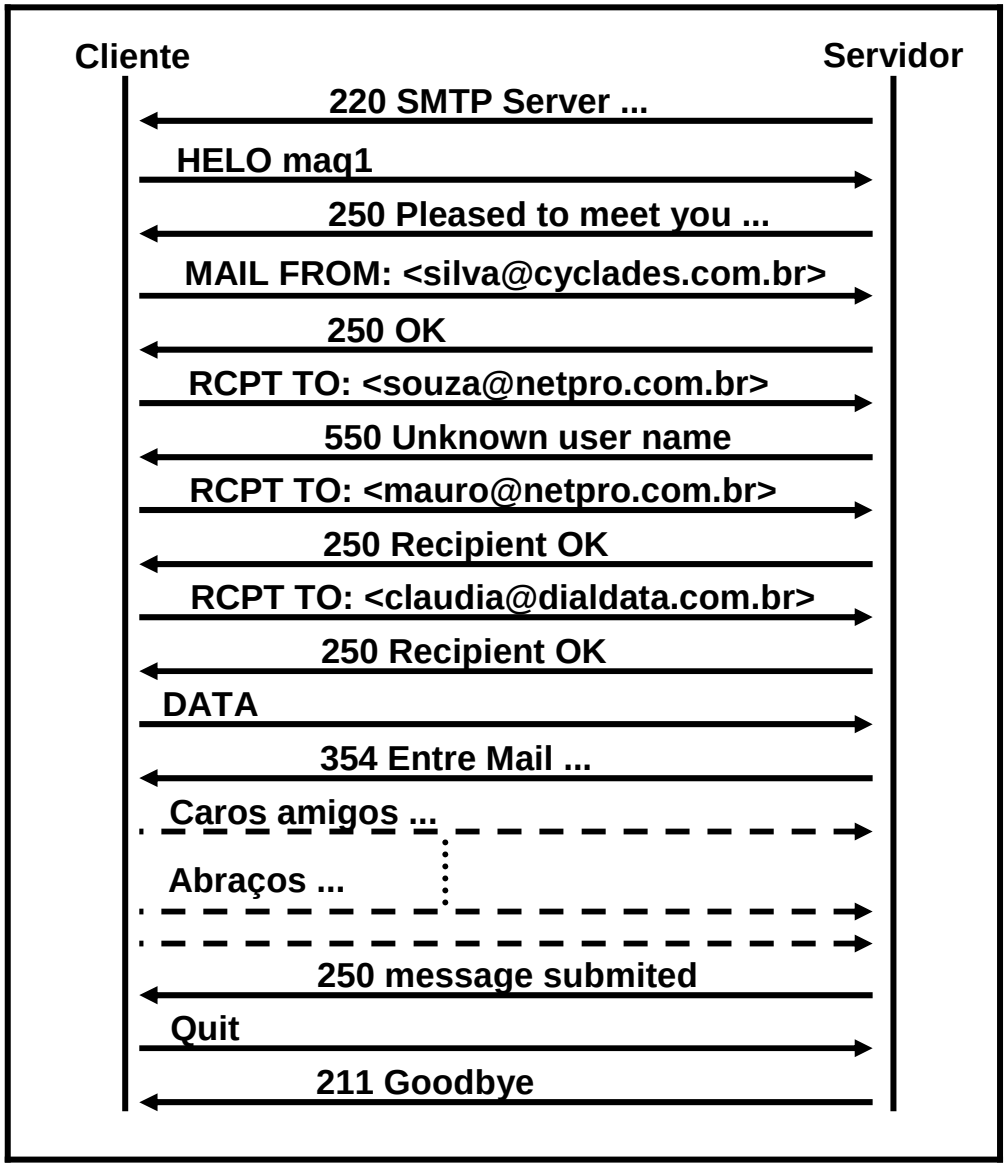
SMTP – formato das mensagens



SMTP – principais comandos

- HELO
- MAIL FROM
- RCPT TO
- DATA
- QUIT

SMTP – exemplo



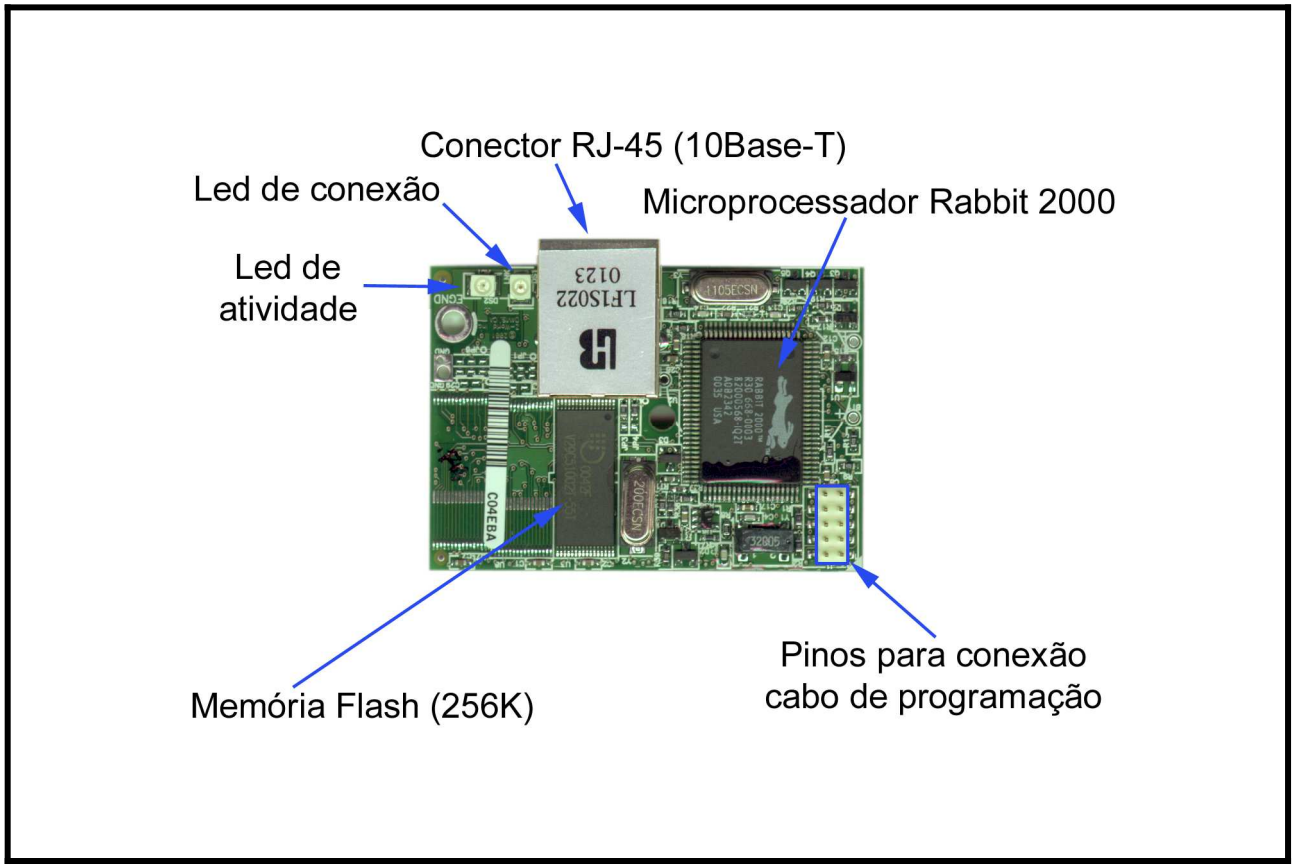
KIT DE DESENVOLVIMENTO RCM 2200

KIT DE DESENVOLVIMENTO RCM 2200

- Módulo RCM 2200
- Placa de Protótipo
- Ambiente de Programação Dynamic C
- Cabo de Programação
- Manuais do Usuário

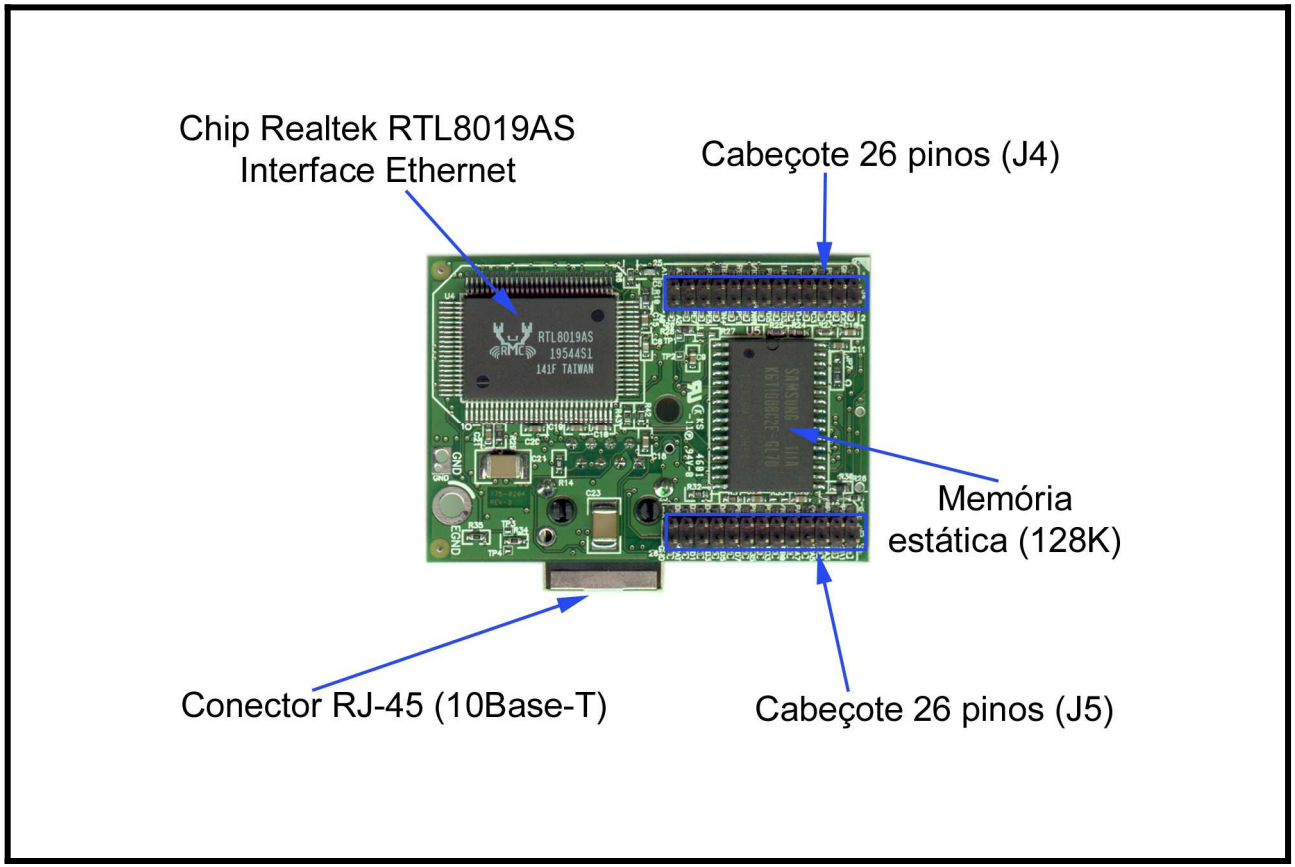
MÓDULO RCM 2200

- Vista Superior

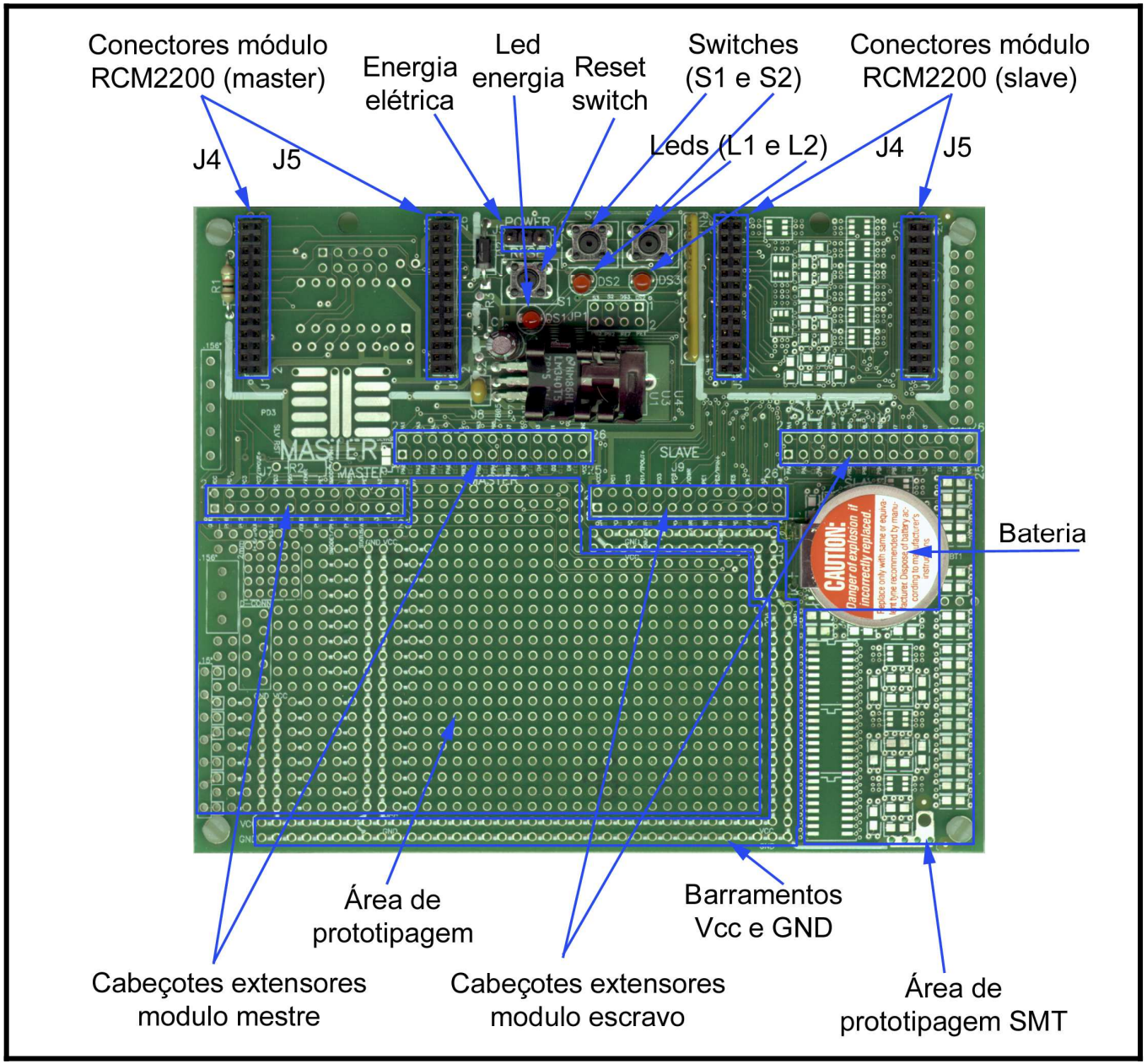


MÓDULO RCM 2200

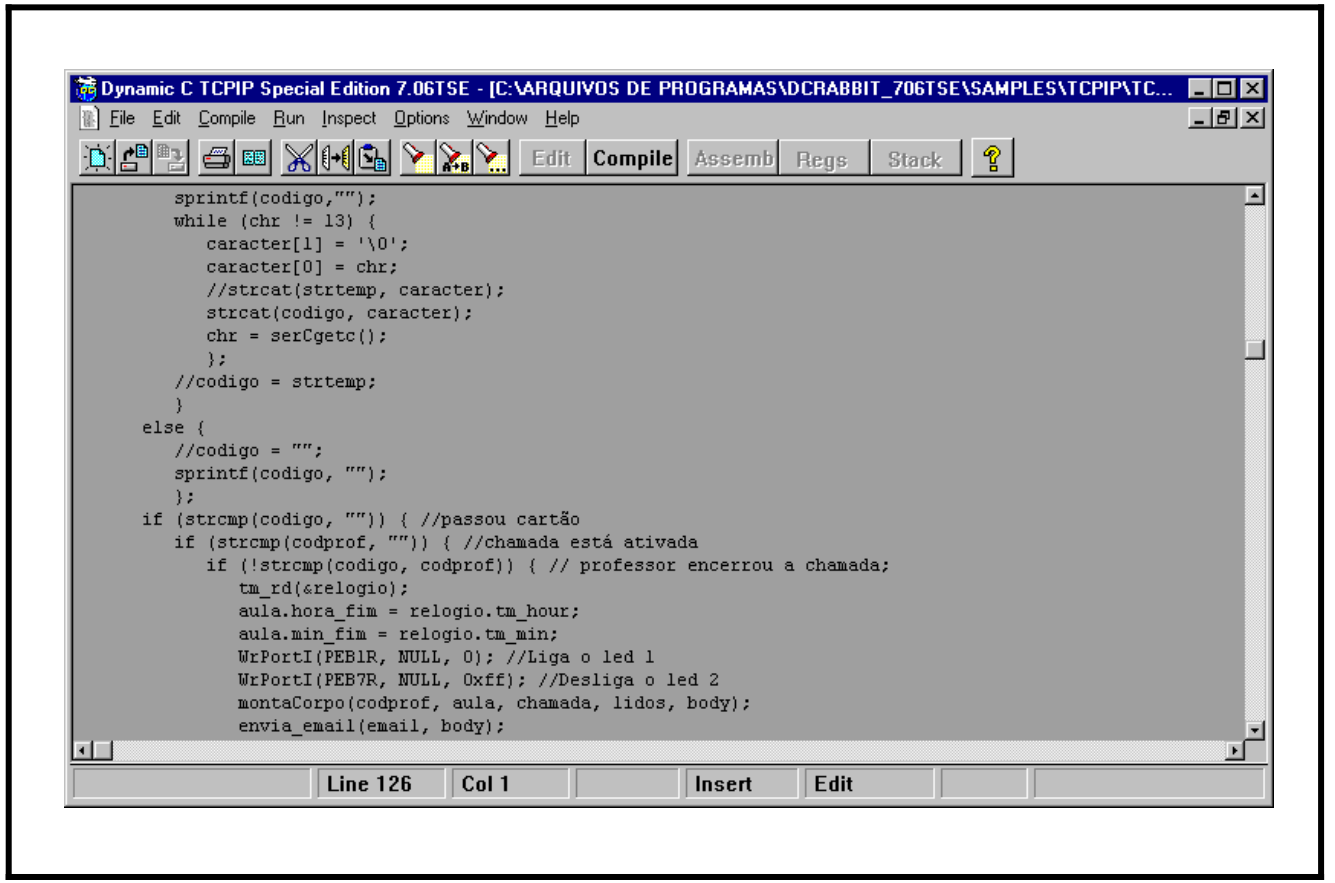
- Vista Inferior



PLACA PROTÓTIPO



AMBIENTE DE DESENVOLVIMENTO DYNAMIC C



DYNAMIC C - bibliotecas

- costate.lib – gerenciamento multitarefa
- math.lib – funções matemáticas
- rs232.lib – transferência de dados
- rtclock.lib – controle de tempo
- string.lib – tratamento de caracteres
- xmem – controle de acesso à memória

DYNAMIC C – bibliotecas tcp/ip

- arp.lib – funções para resolução de endereços IP
- drctcp.lib – funções para os protocolos UDP e TCP
- smtp.lib – funções para o protocolo SMTP
- pop3.lib – funções para o protocolo POP3
- http.lib – funções para o protocolo HTTP

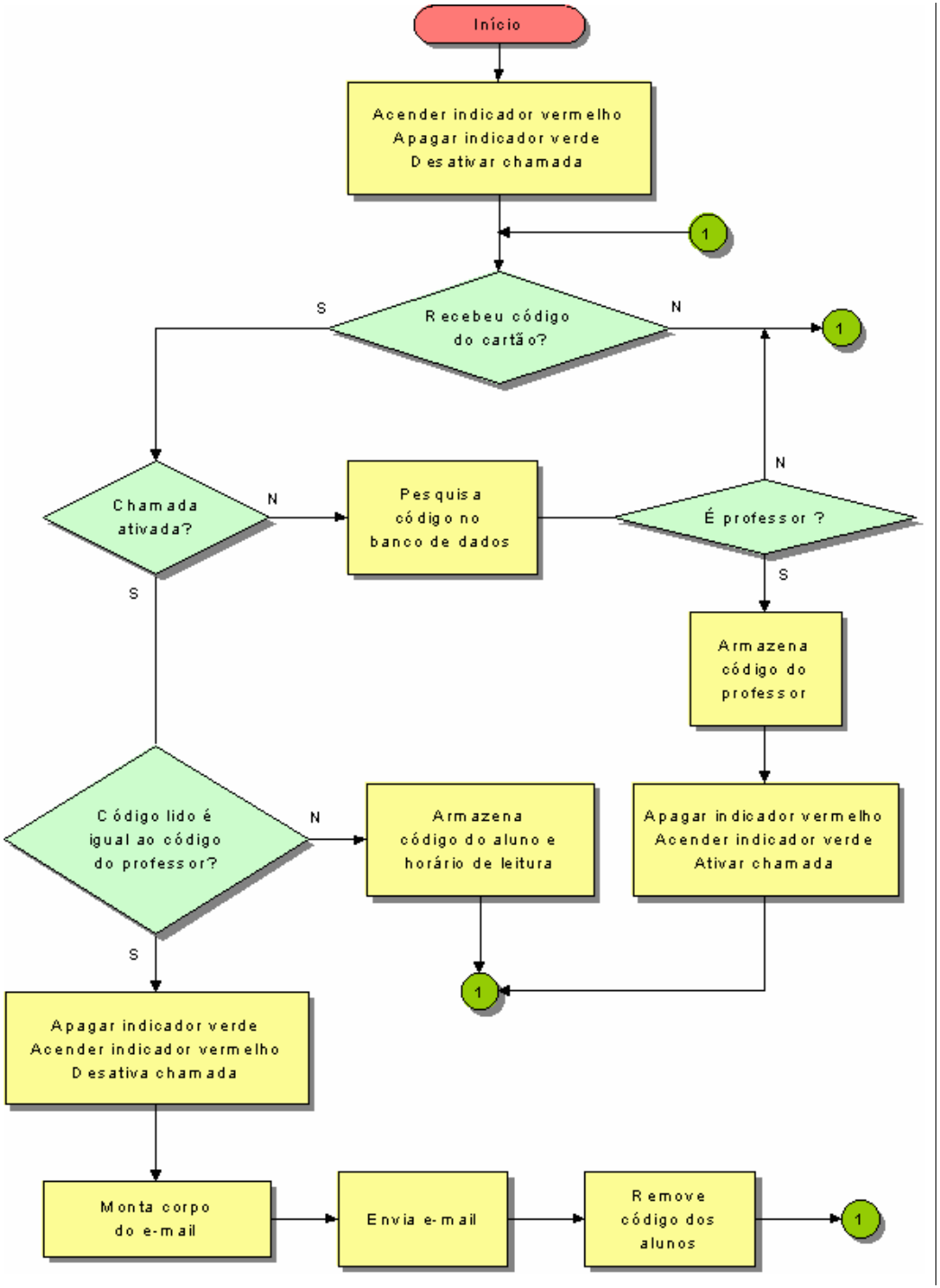
ESPECIFICAÇÃO

O protótipo deverá ler os cartões dos alunos e professores, armazenando-os na memória juntamente com seu horário de leitura. A partir destes dados, deve montar o corpo do *e-mail* na memória, enviando-o logo após, para o endereço eletrônico do professor.

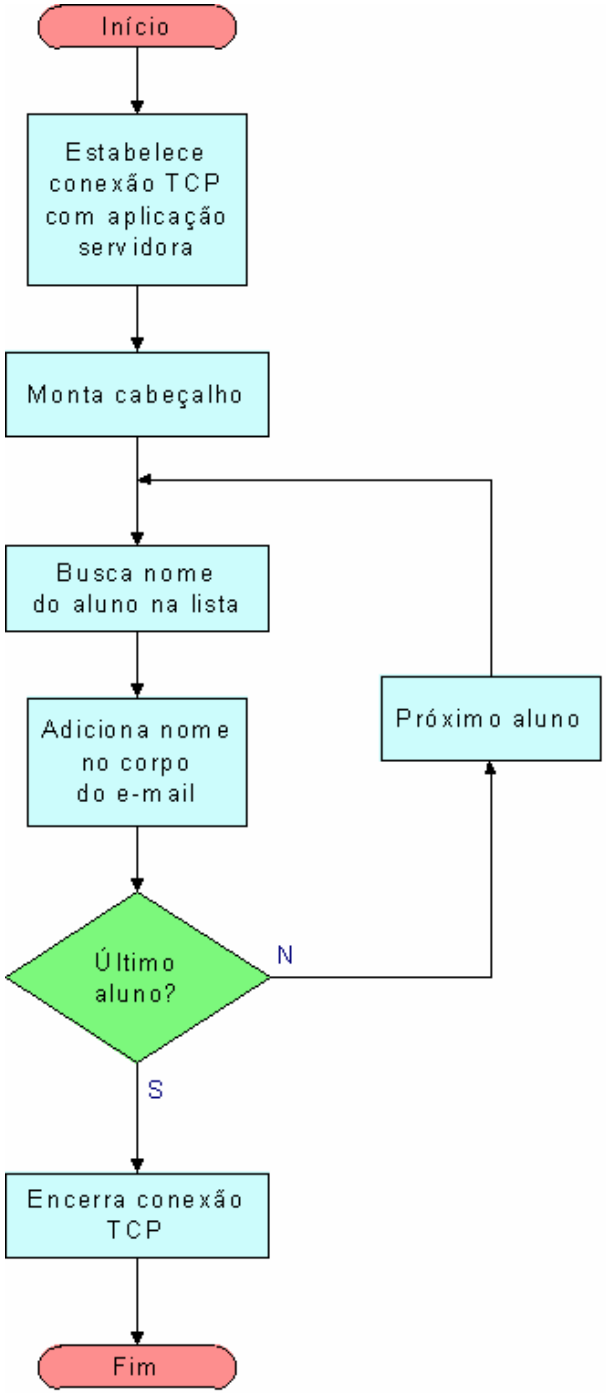
ESPECIFICAÇÃO - funcionamento

- Professor “passa” seu cartão (inicia a chamada)
- Alunos “passam” seus cartões
- Professor “passa” seu cartão (encerra a chamada)
- Enviar e-mail (buscar informações na base de dados)

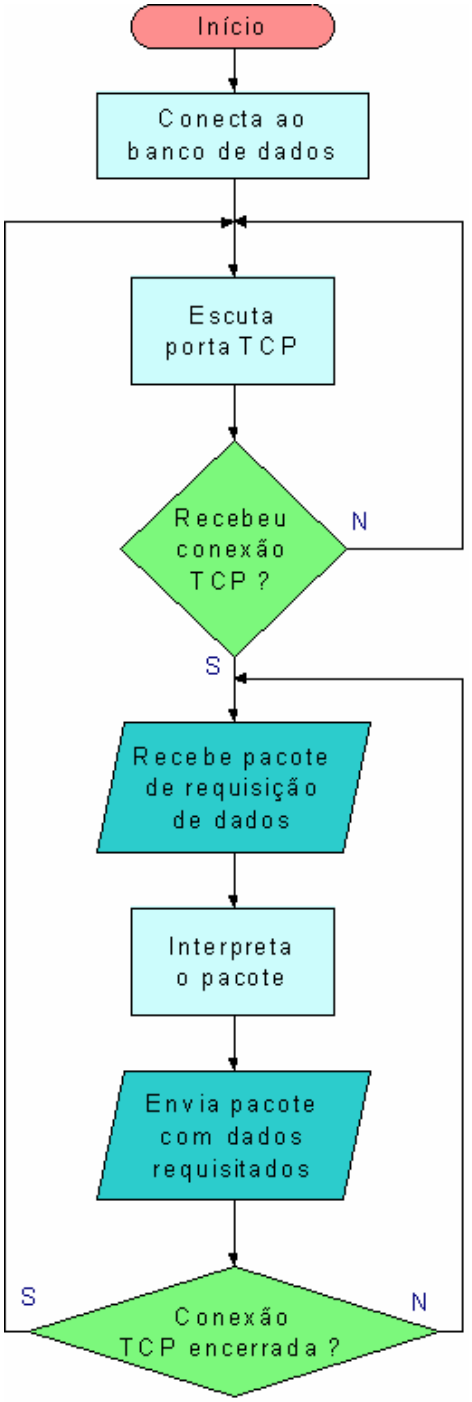
VISÃO GERAL DA APLICAÇÃO CLIENTE



MONTAGEM DO CORPO DO E-MAIL



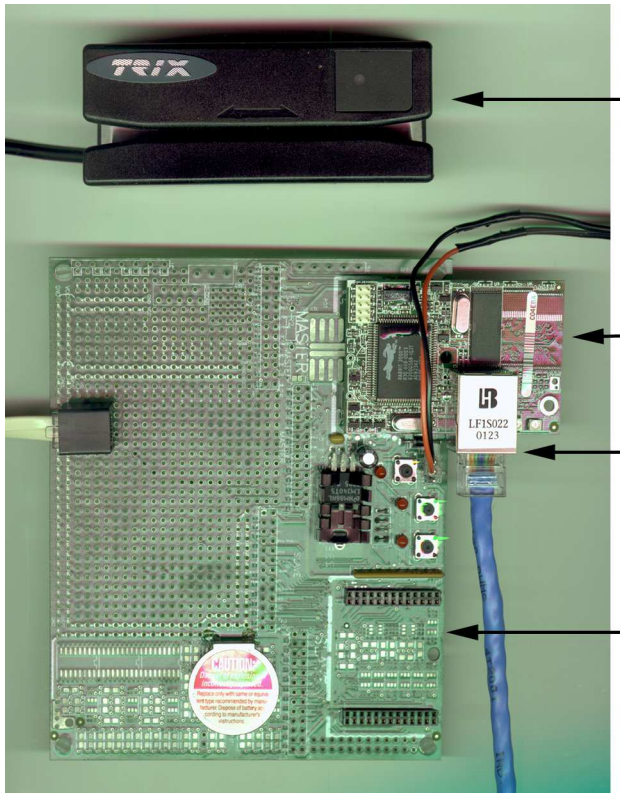
VISÃO GERAL DA APLICAÇÃO SERVIDORA



Comunicação cliente / servidor (mensagens)

- “EHPROFESSOR”
- “NOMEPROFESSOR”
- “NOMEALUNO”
- “CODIGOCADEIRA”
- “CODIGOCURSOCADEIRA”
- “NOMECURSO”
- “NOMECADEIRA”
- “MATRICULADO”

IMPLEMENTAÇÃO



IMPLEMENTAÇÃO

- APLICAÇÃO SERVIDORA
- APLICAÇÃO CLIENTE

APLICAÇÃO SERVIDORA

- DELPHI 5
- SQL (*Structured Query Language*)
- Componente **TSocketServer**
- Evento **OnClientRead**

APLICAÇÃO CLIENTE

- Dynamic C

- Contantes que devem ser inicializadas

```
#define MY_IP_ADDRESS      "200.135.24.126“ //Endereço IP local
#define MY_NETMASK         "255.255.255.0"  //Máscara de rede
#define MY_GATEWAY         "200.135.24.40"  //Gateway
#define MY_NAMESERVER      "200.135.24.7"   //Servidor DNS
```

Socket aplicação servidora

```
#define REMOTE_IP          "200.135.24.67"  //Endereço IP remoto
#define REMOTE_PORT        1000             //Porta remota
```

MÓDULO DE ENVIO DE *E-MAIL*

SMTP.LIB

- smtp_sendmail(remetente, destinatário, assunto, corpo)
- smtp_mailtick()
- smtp_status()

MÓDULO DE ENVIO DE *E-MAIL*

smtp_mailtick / smtp_status

- SMTP_SUCCESS
- SMTP_PENDING
- SMTP_TIME
- SMTP_UNEXPECTED

```
smtp_sendmail(remetente, destinatário, assunto, corpo);

    while(smtp_mailtick()==SMTP_PENDING)
        continue;
```

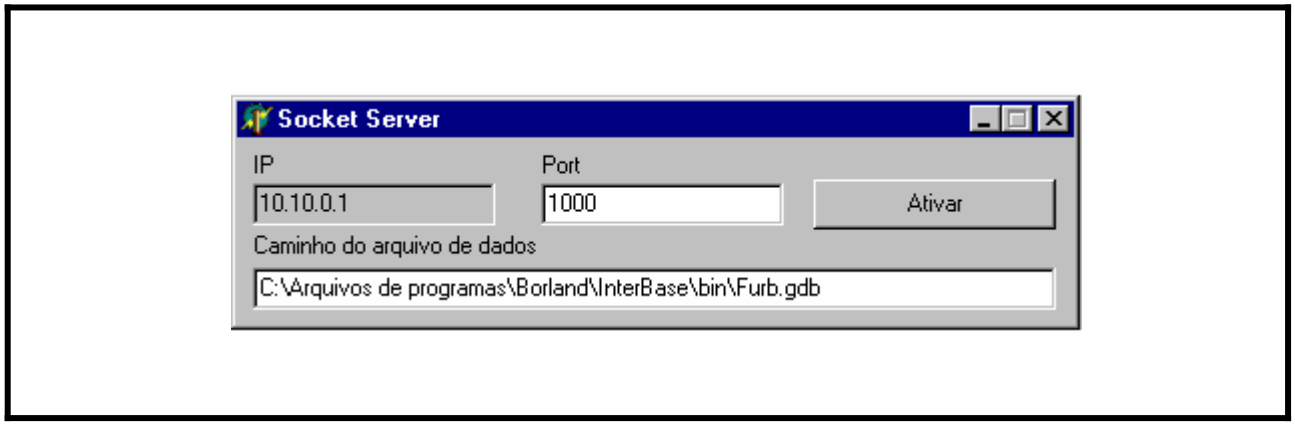
MÓDULO DE MONTAGEM DO *E-MAIL*

- Buscar o nome do curso

```
sprintf(buffer, "NOMECURSO|");
strcat(buffer, cod_curso);
costate{
        waitfor(sock_puts(&s,buffer));
        };
sock_wait_input(&s,0,NULL,&status);
if(sock_gets(&s,buffer,30)) {
        strcat(corpo, "Curso: ");
        strcat(corpo, buffer);
        strcat(corpo, "\n");
        };
```

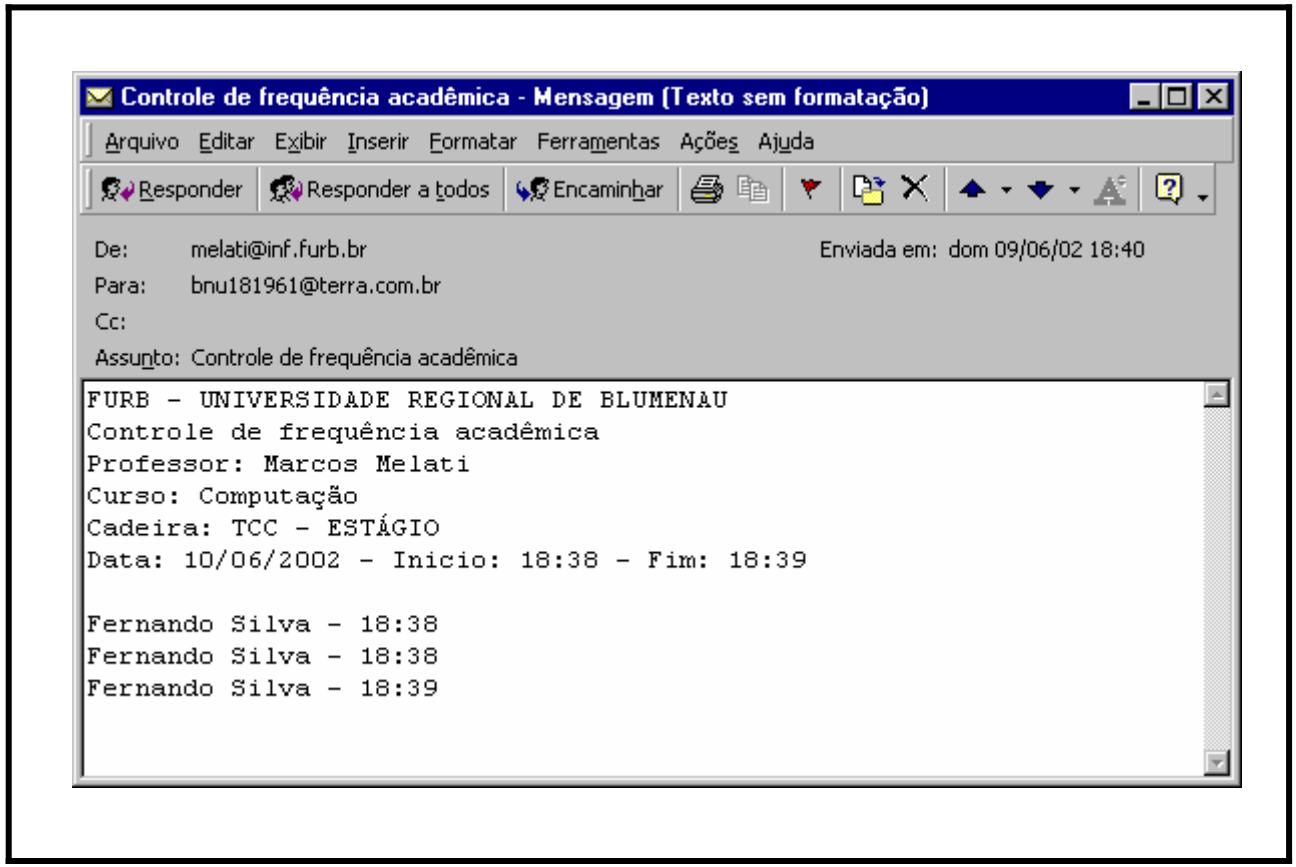
OPERACIONALIDADE

- Ativar a Aplicação Servidora
- Conectar o protótipo na rede, devidamente configurado



OPERACIONALIDADE

- *E-MAIL* MONTADO E ENVIADO PELO PROTÓTIPO

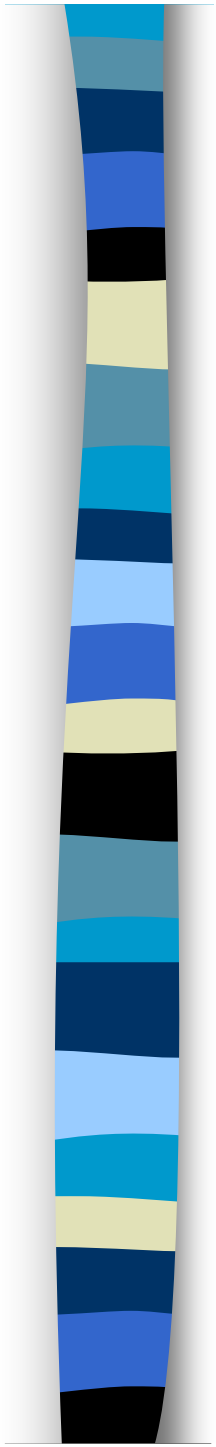


CONSIDERAÇÕES FINAIS

- O trabalho atingiu o objetivo proposto
- Proposta de automação do processo de freqüência acadêmica

EXTENSÕES

- Rotinas de configuração automática das aplicações clientes através da aplicação servidora, configurando automaticamente o endereço IP, horário
- Implementação de uma página na Internet para pesquisa às listas de freqüências
- Implementação da aplicação servidora tornando-a compatível com outros bancos de dados e configurável a diversas modelagens de dados



FIM