Apêndice A

Especificação formal em ESTELLE do protocolo Multiponto-a-Multiponto

A listagem a seguir contém a especificação em Estelle do protocolo apresentado neste trabalho. Esta é a especificação usada nos testes finais, onde a entidade que simula o meio físico pode perder quadros.

Os nomes das transições contém informação sobre o estado presente, a PDU recebida e predicados satisfeitos para o disparo da transição. Isto facilita a leitura do arquivo de *trace* criado pelo simulador ESTELLE.

```
SPECIFICATION MPMP systemactivity ; default individual queue;
TIMESCALE seconds;

(*
  Em Estelle, os quadros nao serao reordenados, sendo enviados para o
  usuario assim que chegarem, ou seja, nao ha controle de quadros
  recebidos fora de ordem.
*)

  {$ external NetUser }

Const

NUM_ESTACOES  = 4;
JANELA  =8;
MAX_TENT= 3;

(********** Constantes de tempo **********)
TIM=6;        (* mestra em silencio *)
TR=5;         (* retransmissao geral *)
TRQ=5;        (* retransmissao de quadros de dados do usuario *)
TACK=6;       (* envio de RR *)
TIP=13;       (* silencio no canal principal *)
TI=12;        (* silencio numa estacao *)
TMT=18;       (* maximo como mestra *)
TKPR=20;      (* 4*TMT+2: aguardo da ficha. na verdade nunca dispara, pois *)
              (* TMT dispara antes e ao receber uma ficha (TG) destinada a *)
```

```
            (* outra estacao e verificar que nao se encontra na fila, a  *)
            (* ficha eh novamente pedida.                                 *)


TYPE
string = Array[0..70] Of Char;

Tipo_Dado = string;



(* Campos das PDUs, por tipo de PDU
   Al,m da identificacao da fonte e do destino e c¢digo da PDU, pode-se ter:

 1: N£mero do quadro
 2: nada
 3: N£mero do quadro, Dados (usuario ou protocolo)
 4: N£mero do quadro, Id. da nova Mestra, Fila de espera

 PDUs por tipo

 1: RR, SREJ
 2: SMPARM, MPARMAC, MPDISC, MPDISCAC, MPRP, DISC, DISCAC, RP, TP, BM e CM
 3: MPI
 4: TG, PLIST

*)

Tipo_Cod_PDU = (RR, SREJ, SMPARM, MPARMAC, MPDISC, MPDISCAC, MPRP, DISC,
                DISCAC, RP, TP, BM, CM, MPI, TG, PLIST);

Tipo_Fila = Array[1..NUM_ESTACOES] of integer;
Tipo_lista = Array[1..NUM_ESTACOES] of Boolean; (* indica se uma estacao
                                                   esta participando ou
                                                   nao da conferencia *)

Tipo_PDU =
  Record
  fonte, destino:integer;
  cod:Tipo_cod_PDU;
  num,
  mestra,
  primf,
  ultf:integer;
  fila:Tipo_FILA;
  lista:Tipo_Lista;
  dados:Tipo_Dado;
  End;

Tipo_Conf= (iniciada,nova);

Tipo_razao = (USU, NMT, SVC, ERR);


  procedure WriteLn; primitive;
  procedure WriteInt ( i : integer ); primitive;
  procedure ReadInt ( var i : integer ); primitive;
  procedure WriteString ( s : string ); primitive;
```

Apêndice A . 3

```
CHANNEL MPENL_USU_Interface (usu, enlace);
  BY usu:
    MPDL_EST;                    { MPDL_ESTABLISH.Req }
    MPDT(sdu:Tipo_Dado);         { MPDL_DATA.Req }
    MPDL_AB;                     { MPDL_ABORT.Req }
    MPDL_RL;                     { MPDL_RELEASE_PLEASE.Req }
    DL_AB(ID:Integer);           { DL_ABORT.Req }
    DL_RL;                       { DL_RELEASE_PLEASE.Req }
    MPDL_RY(ID:Integer);         { MPDL_REPLAY.Req }
    MPDL_TG;                     { MPDL_TOKEN_GIVE.Req }
    MPDL_TP;                     { MPDL_TOKEN_PLEASE.Req }

  BY enlace:
    MPDL_EST_Ind(fonte:integer);    { MPDL_ESTABLISH.Ind }
    MPDL_EST_Conf(stat:tipo_conf); { MPDL_ESTABLISH.Conf }
    MPDL_AB_Ind(razao:tipo_razao); { MPDL_ABORT.Ind }
    MPDL_AB_Conf;                  { MPDL_ABORT.Conf }
    MPDL_RL_Ind;                   { MPDL_RELEASE_PLEASE.Ind }
    MPDL_DT_Ind(sdu:Tipo_Dado);    { MPDL_DATA.Ind }
    DL_AB_Ind;                     { DL_ABORT.Ind }
    DL_AB_Conf(id:integer);        { DL_ABORT.Conf }
    DL_RL_Ind(id:integer);         { DL_RELEASE_PLEASE.Ind }
    MPDL_TP_Ind(id:integer);       { MPDL_TOKEN_PLEASE.Ind }
    MPDL_TG_Ind;                   { MPDL_TOKEN_GIVE.Ind }
    STOP_TX;                       { STOP_TRANSMISSION.Ind }


CHANNEL MPENL_PH_Interface (enlace, phy);
                  { Ser  considerado que a conexao PHY ja est
                    estabelecida e que o canal esta aberto para o envio
                    de dados (SDUs). }
  BY enlace:
    DT_Req(PDU:Tipo_PDU);
  BY phy:
    DT_Ind(PDU:Tipo_PDU);

MODULE Entidade_MPENL activity (aux_id_est:Integer);

IP UCEP : MPENL_USU_Interface (enlace);
   PCEP : MPENL_PH_Interface (enlace);

END;

BODY ENLACE_ENTITY_BODY FOR Entidade_MPENL;

type
Tipo_Controle_Quadro_Tx =
  Record
  ocupado,             (* indica se este controle esta livre *)
  srej : boolean;      (* indica se o quadro recebeu um SREJ *)
  num,                 (* numero do quadro *)
  falta_conf:integer;  (* num. de estacoes que faltam confirmar *)
  pdu:Tipo_Pdu;
  confirmou:Tipo_Lista (* true no num. da est. que confirmou *)
  End;
```

Apêndice A . 4

```
Rx_Tipo_Controle_Quadro =
  Record
  ocupado :Boolean;
  num     :Integer
  End;

VAR
buffer_tx,pdu_aux : Tipo_PDU;
data_aux:Tipo_Dado;
pdu_str:string;

cont_ti,
vr: array[1..NUM_ESTACOES] of integer; (* contador de quadros indepen- *)
                                       (* dente para cada estacao      *)
cont_tim,
cont_tr,
cont_trq,
cont_tip,
cont_tmt,
cont_tack,
cont_tkpr,

id_est,           (* identidade da estacao *)
ns,vs,      (* numerador de quadros: TX e ack RX *)
num_tent,                 (* numero de tentativas *)

falta_conf_conex:integer; (* numero de estacoes que nao *)
                          (* confirmaram conexao *)

conf_conex: Array[1..NUM_ESTACOES] of Boolean;

c_tx: Array[1..JANELA] of Tipo_Controle_Quadro_Tx;
c_rx: Array[1..JANELA,1..NUM_ESTACOES] of Rx_Tipo_Controle_Quadro;

nao_rec_cm,           (* nao recebeu CM da estacao i *)
prim_quadro,          (* primeiro quadro numerado recebido da estacao i *)
lista_part:Tipo_Lista; (* lista das estacoes participantes *)

pfrx:Array[1..NUM_ESTACOES] of integer;  (* primeiro da fila de quadros esperados
*)

pftx,

num_cm_rec,
id_nova_mestra,       (* id da estacao para onde foi enviada a ficha *)
id_mestra,            (* identidade da mestra atual *)
prim_fila,            (* indice do primeiro da fila *)
ult_fila,             (* indice do ultimo da fila   *)
num_fila,             (* num de est na fila de espera *)
num_part:integer;     (* numero de estacoes participantes *)
fila_esp:Tipo_Fila;   (* fila de espera para tx *)

to_awtkrec,          (* auxiliares de transicao *)
to_open_master,
to_closed,           (* auxiliar em AWAITING_TOKEN_RECEIVE *)
prim_q_env,          (* primeiro quadro enviado *)
canal_P_ocupado,     (* canal principal ocupado *)
mestra,              (* a estacao , a mestra *)
```

```
pediu_ficha:Boolean; (* a estacao pediu a ficha *)

cont_rr,
i,j,k,l:integer; (* auxiliares em loops, etc. *)

{$ verificar var cont_tr}
{$ verificar var cont_tmt}
{$ verificar var to_closed}
{$ verificar var cont_tim}

STATE INICIA, CLOSED, ESTABLISH, AWAITING_RELEASE, AW_UN_RELEASE,
      AW_TK_REC, OPEN_REC, OPEN_MASTER, ESTABLISH_RECEIVER;

PROCEDURE Cria_SMPARM (Var PDU:Tipo_PDU; origem,destino:integer);
(* Pedido de inicio de conferencia *)
BEGIN
pdu.cod:=SMPARM;
pdu.num:=-1;
pdu.fonte:=origem;
pdu.destino:=destino
END;

PROCEDURE Cria_RR (Var PDU:Tipo_PDU; num,origem,destino:integer);
BEGIN
pdu.cod:=RR;
pdu.num:=num;
pdu.fonte:=origem;
pdu.destino:=destino
END;

PROCEDURE Cria_SREJ (Var PDU:Tipo_PDU; num,origem,destino:integer);
BEGIN
pdu.cod:=SREJ;
pdu.num:=num;
pdu.fonte:=origem;
pdu.destino:=destino
END;

PROCEDURE Cria_MPARMAC (Var PDU:Tipo_PDU; origem,destino:integer);
BEGIN
pdu.cod:=MPARMAC;
pdu.num:=-1;
pdu.fonte:=origem;
pdu.destino:=destino;
if mestra
  then pdu.mestra:=1
  else pdu.mestra:=0
END;

PROCEDURE Cria_MPDISC (Var PDU:Tipo_PDU; origem:integer);
BEGIN
pdu.cod:=MPDISC;
pdu.fonte:=origem;
pdu.destino:=255
END;

PROCEDURE Cria_MPDISCAC (Var PDU:Tipo_PDU; origem,destino:integer);
BEGIN
```

```
pdu.cod:=MPDISCAC;
pdu.fonte:=origem;
pdu.destino:=destino
END;

PROCEDURE Cria_MPRP (Var PDU:Tipo_PDU; origem,destino:integer);
BEGIN
pdu.cod:=MPRP;
pdu.fonte:=origem;
pdu.destino:=destino
END;

PROCEDURE
     Cria_MPI (Var PDU:Tipo_PDU; UDATA:Tipo_Dado; num,origem:integer);
BEGIN
pdu.cod:=MPI;
pdu.num:=num;
pdu.dados:=UDATA;
pdu.fonte:=origem;
pdu.destino:=255
END;

PROCEDURE Cria_DISC (Var PDU:Tipo_PDU; origem,destino:integer);
BEGIN
pdu.cod:=DISC;
pdu.fonte:=origem;
pdu.destino:=destino
END;

PROCEDURE Cria_DISCAC (Var PDU:Tipo_PDU; origem,destino:integer);
BEGIN
pdu.cod:=DISCAC;
pdu.fonte:=origem;
pdu.destino:=destino
END;

PROCEDURE Cria_RP (Var PDU:Tipo_PDU; origem,destino:integer);
BEGIN
pdu.cod:=RP;
pdu.fonte:=origem;
pdu.destino:=destino
END;

PROCEDURE Cria_TP (Var PDU:Tipo_PDU; origem,destino:integer);
BEGIN
pediu_ficha:=true;
pdu.cod:=TP;
pdu.fonte:=origem;
pdu.destino:=destino
END;

PROCEDURE    Cria_TG    (Var    PDU:Tipo_PDU;    origem,new_master,pf,uf:integer;
fila:tipo_fila);
BEGIN
pdu.cod:=TG;
pdu.fonte:=origem;
pdu.destino:=255;
pdu.mestra:=new_master;
```

```
pdu.primf:=pf;
pdu.ultf:=uf;
pdu.fila:=fila
END;

PROCEDURE Cria_BM (Var PDU:Tipo_PDU; origem:integer);
BEGIN
pdu.cod:=BM;
pdu.fonte:=origem;
pdu.destino:=255
END;

PROCEDURE Cria_CM (Var PDU:Tipo_PDU; origem,destino:integer);
BEGIN
pdu.cod:=CM;
pdu.fonte:=origem;
pdu.destino:=destino
END;

PROCEDURE Cria_PLIST (Var PDU:Tipo_PDU; num,origem:integer);
BEGIN
pdu.cod:=PLIST;
pdu.lista:=lista_part;
pdu.ultf:=num_part;
pdu.num:=num;
pdu.fonte:=origem;
pdu.destino:=255
END;


PROCEDURE Data (PDU:Tipo_PDU; Var data:Tipo_Dado);
BEGIN
data:=pdu.dados
END;

Procedure Inicia_Ti_Participantes;
var i:integer;
Begin
for i:=1 to NUM_ESTACOES do if lista_part[i] Then
  begin
  if i<>id_est then cont_ti[i]:=TI; (* A2 *)
  writestring('-');
  writeint(i)
  end;
End;

FUNCTION Esta_na_Fila(id:integer):Boolean;
var
i:integer;
n_achou:Boolean;

Begin
i:=prim_fila;
n_achou:=true;
while (i<>ult_fila) and n_achou do
  if id=fila_esp[i]
    then n_achou:=false
    else i:=(i mod NUM_ESTACOES)+1;
```

```
Esta_na_Fila:=not n_achou;
end; { esta_na_fila }

FUNCTION Esta_na_Lista(id:integer; Var lista_part:Tipo_Lista):Boolean;
Begin
Esta_na_Lista:=lista_part[id]
end; { Esta_na_Lista }

Procedure Trans_Pdu_Str(cod_pdu:Tipo_Cod_Pdu);
Begin
Case cod_pdu of
 RR       :Pdu_Str:='RR';
 SREJ     :Pdu_Str:='SREJ';
 SMPARM   :Pdu_Str:='SMPARM';
 MPARMAC  :Pdu_Str:='MPARMAC';
 MPDISC   :Pdu_Str:='MPDISC';
 MPDISCAC :Pdu_Str:='MPDISCAC';
 MPRP     :Pdu_Str:='MPRP';
 DISC     :Pdu_Str:='DISC';
 DISCAC   :Pdu_Str:='DISCAC';
 RP       :Pdu_Str:='RP';
 TP       :Pdu_Str:='TP';
 BM       :Pdu_Str:='BM';
 CM       :Pdu_Str:='CM';
 MPI      :Pdu_Str:='MPI';
 TG       :Pdu_Str:='TG';
 PLIST    :Pdu_Str:='PLIST';
end {case}
end;

Procedure Msg_Id(msg:string);
Begin
Writeln;
Writeint(id_est);
Writestring(msg)
End;

Procedure Ver_Descarte(nr,fonte:integer; reinicia_trq:Boolean);
var i,j:integer;
Begin
i:=pftx;
While (c_tx[i].num<nr) and c_tx[i].ocupado and ((i mod JANELA)+1<>pftx) do
  Begin
  if not c_tx[i].confirmou[fonte] then
    Begin
    c_tx[i].confirmou[fonte]:=true;
    c_tx[i].falta_conf:=c_tx[i].falta_conf-1;
    if c_tx[i].falta_conf=0 then
      Begin
      c_tx[i].ocupado:=false;
      c_tx[i].num:=-1
      end
    end;
  i:=(i mod janela) +1
  end;
i:=JANELA;
While not c_tx[pftx].ocupado and (i>0) do
  Begin
```

```
  pftx:=(pftx mod janela)+1;  (* P4: A10 *)
  i:=i-1;
  if reinicia_trq Then cont_trq:=TRQ  (* A68 *)
  End;
if cont_rr >0 Then
  begin
  Msg_Id(': RR(');
  writeint(nr);
  writestring(') de ');
  writeint(fonte);
  writestring('. c_tx=');
  i:=pftx;
  for j:=1 to janela do
    Begin
    if c_tx[i].ocupado Then
      Begin
      writeint(c_tx[i].num);
      writestring('-')
      End;
    i:=(i mod JANELA) +1
    End;
  cont_rr:=cont_rr-1
  End
end;

Procedure Ver_Seq_Recebida(num,id_fonte:integer; cod:Tipo_Cod_Pdu);
var i,j,k:integer;
Begin
k:=vr[id_fonte];
if cod=MPI Then
  Begin
  Writestring(' esp:');
  writeint(k);
  if k=num Then writestring(' num OK')
  End;
if num < k Then
  Begin
  j:=pfrx[id_fonte];
  i:=j;
    while ((j mod janela)+1<>i) and (c_rx[j,id_fonte].num<>num) do j:=(j mod
janela)+1;
  if (c_rx[j,id_fonte].num=num)
    then Begin
         c_rx[j,id_fonte].ocupado := false;
         c_rx[j,id_fonte].num:=-1;
         writestring(' em retrans.');
         j:=JANELA;
         While not c_rx[i,id_fonte].ocupado and (j>0) do
           Begin
           i:=(i mod janela)+1;
           j:=j-1
           End;
         pfrx[id_fonte]:=i;
         end
    else if cod=MPI Then Writestring('. Descartado: ja recebido');
  end;
if num > k then
  Begin
```

```
  Msg_Id(' envia SREJ p/ ');
  writeint(id_fonte);
  writestring(' p/ os quadros');
  while k<num do
    begin
    writestring(':');
    writeint(k);
    Cria_SREJ(pdu_aux,k,id_est,id_fonte);
    Output PCEP.DT_Req(pdu_aux);
    i:=pfrx[id_fonte];
    j:=i;
    while ((i mod JANELA)+1 <>j) and (c_rx[i,id_fonte].ocupado) do i:=i+1;
    if c_rx[i,id_fonte].ocupado
      then Begin
          Writeln;
           writestring('NAO ESPERADO: FALTOU ESTRUTURA DE CONTROLE NO
                       RECEPTOR ');
           writeint(id_est)
           end
      else Begin
           c_rx[i,id_fonte].ocupado:=true;
           c_rx[i,id_fonte].num:=k
           end;
    k:=k+1
    end;
  end;
if num>=k Then vr[id_fonte]:=num+1
End;

Procedure Inic_Var_Rec;
Begin
mestra:=false;
num_tent:=1
End;

Procedure Inic_Var_Mestra; (* A13 *)
Begin
pediu_ficha:=false;
mestra:=true;
id_mestra:=id_est;
num_tent:=1;
cont_tim:=TIM
End;

Procedure Limpa_Confirm;
var i:integer;
Begin
for i:=1 to NUM_ESTACOES do conf_conex[i]:=false;
falta_conf_conex:=NUM_ESTACOES-1
End;

Procedure Inicia_Variaveis;
var i,k:integer;
Begin
prim_q_env:=true;
canal_P_ocupado:=false;
pediu_ficha:=false;
mestra:=false;
```

```
to_awtkrec:=false;
to_closed:=false;
to_open_master:=false;
cont_tack:=-1;
cont_tip:=-1;
cont_tr:=-1;
cont_tmt:=-1;
cont_tkpr:=-1;
cont_tim:=-1;
cont_rr:=0;
for i:=1 to JANELA do
  begin
  c_tx[i].ocupado:=false;
  c_tx[i].srej:=false;
  c_tx[i].num:=-1;
  c_tx[i].falta_conf:=NUM_ESTACOES-1;
  for k:=1 to NUM_ESTACOES do
    Begin
    c_tx[i].confirmou[k]:=false;
    c_rx[i,k].ocupado:=false;
    c_rx[i,k].num:=-1
    End
  End;
num_part:=0;
for i:=1 to NUM_ESTACOES do
  begin
  nao_rec_cm[i]:=true;
  conf_conex[i]:=false;
  lista_part[i]:=false;
  fila_esp[i]:=0;
  vr[i]:=0;
  prim_quadro[i]:=true;
  pfrx[i]:=1;
  cont_ti[i]:=-1;
  end;
prim_fila:=1;
ult_fila:=1;
num_fila:=0;
num_cm_rec:=0;
ns:=0;
vs:=0;
pftx:=1;
id_mestra:=0;
num_tent:=1
End;

Procedure Envia_PLIST;
var i,j:integer;
Begin
i:=pftx;
While (((i mod JANELA) +1)<>pftx) and (c_tx[i].ocupado) do i:=(i mod janela)+1;
if ((i mod JANELA) +1)<>pftx
  then begin
       ns:=vs;
       vs:=vs+1;
       Cria_PLIST(pdu_aux,ns,id_est);
       c_tx[i].ocupado:=true;
       c_tx[i].num:=ns;
```

```
       c_tx[i].falta_conf:=num_part-1;
       for j:=1 to NUM_ESTACOES do c_tx[i].confirmou[j]:=false;
       c_tx[i].pdu:=pdu_aux;
       Output PCEP.DT_Req(pdu_aux);
       cont_tim:=TIM
       end
  else begin
       Writeln;
       writestring('Encheu a janela na '); (* NP6: A14 *)
       writeint(id_est)
       End
End;

Procedure Coloca_na_Fila(id_estacao:integer);
var i:integer;
Begin
If Esta_na_fila(id_estacao)
  then writestring('*')
  Else Begin
       if ((ult_fila mod NUM_ESTACOES) +1) = prim_fila
         Then writestring(' MELDA: ENCHEU A FILA DE ESPERA ')
         else Begin
              fila_esp[ult_fila]:=id_estacao;
              ult_fila:=(ult_fila mod NUM_ESTACOES)+1
              end
       End;
writestring('). Fila atual');
i:=prim_fila;
While i<>ult_fila do
  Begin
  writestring(':');
  writeint(fila_esp[i]);
  i:=(i mod NUM_ESTACOES)+1
  End;
Writestring('. Prim:');
writeint(prim_fila);
writestring(' Ult:');
writeint(ult_fila)
End;

Procedure Retira_da_Fila(id:integer);
Begin
i:=prim_fila;
while i<>ult_fila do
  Begin
  if fila_esp[i]=id
    Then Begin  (* P12: A22 *)
         j:=(i mod NUM_ESTACOES)+1;
         while j<>ult_fila do
           Begin
           fila_esp[i]:=fila_esp[j];
           i:=j;
           j:=(j mod NUM_ESTACOES)+1
           End;
         ult_fila:=i
         End
    Else Begin i:=(i mod NUM_ESTACOES)+1 End
  End
```

```
End;

Procedure Retransmite(num_q:integer);
var i,j:integer;
Begin
cont_tim:=TIM;
j:=pftx;
for i:=1 to JANELA do (* P22: MPI(Nr) *)
  if (c_tx[j].ocupado) and (c_tx[j].num=num_q) Then
    Begin
    pdu_aux:=c_tx[j].pdu;
    pdu_aux.dados:='retrans: SREJ ';
    writeln;
    writestring('retransmitindo quadro ');
    writeint(num_q);
    Output PCEP.DT_Req(pdu_aux);
    j:=(j mod JANELA)+1
    end
End;

Procedure Transmite_MPI;
var i,j:Integer;
Begin
i:=pftx;
While (((i mod JANELA) +1)<>pftx) and (c_tx[i].ocupado) do i:=(i mod janela)+1;
if ((i mod JANELA) +1)<>pftx
  Then Begin      (* P6 *)
       ns:=vs;    (* A9 *)
       vs:=vs+1;
       c_tx[i].ocupado:=true;
       c_tx[i].num:=ns;
       c_tx[i].falta_conf:=num_part-1;
       for j:=1 to NUM_ESTACOES do c_tx[i].confirmou[j]:=false;
       Cria_MPI(pdu_aux,'Teste de dados',ns,id_est);
       c_tx[i].pdu:=pdu_aux;
       Output PCEP.DT_Req(pdu_aux);    (* MPI(Ns) *)
       if prim_q_env then
         begin
         cont_trq:=TRQ; (* P3: A68 *)
         num_tent:=1;
         prim_q_env:=false
         end;
       cont_rr:=NUM_ESTACOES;
       cont_tim:=TIM
       end
  Else Begin
       Writeln;
       writestring('Encheu a janela na '); (* NP6: A14 *)
       writeint(id_est)
       End
End;

Procedure Env_Stop_Tx_Tp(dest:integer);
Begin
Msg_Id('=> STOP_TRANSMISSION.Ind'); (* A59 *)
{ nao retirar primitiva da fila (COMO?) }  (* A14 *)
Cria_TP(pdu_aux,id_est,dest);
Output PCEP.DT_Req(pdu_aux);
```

```
cont_tkpr:= TKPR                          (* A23 *)
End;

Procedure Env_3BM_e_Retrans;
var i:integer;
Begin
Cria_BM(pdu_aux,id_est);
Output PCEP.DT_Req(pdu_aux);
Output PCEP.DT_Req(pdu_aux);
Output PCEP.DT_Req(pdu_aux);
Inicia_Ti_Participantes;
Inic_Var_Mestra; (* A13 *)
cont_tmt:=TMT; (* A26 *)
cont_tack:=-1; (* A40 *)
cont_trq:=TRQ;   (* A68 *)
num_tent:=1;
id_mestra:=id_est;
Msg_Id(': MPL_TOKEN_GIVE.Ind');
j:=pftx;
cont_rr:=NUM_ESTACOES;
for i:=1 to JANELA do (* NP5: A11 *)
  if c_tx[j].ocupado Then
    Begin
    pdu_aux:=c_tx[j].pdu;
    pdu_aux.dados:='re-Tx: rec=>mestra';
    Output PCEP.DT_Req(pdu_aux);
    j:=(j mod JANELA)+1
    end;
End;

Procedure Estourou(msg:string);
var i:Integer;
Begin
cont_tim:=TIM;  (* A65 *)
Msg_Id(': estourou ');
writestring(msg);
writestring(' =>');
j:=pftx;
if c_tx[j].ocupado
  Then Begin
       for i:=1 to JANELA do (* NP5: A11 *)
         if c_tx[j].ocupado Then
           Begin
           pdu_aux:=c_tx[j].pdu;
           pdu_aux.dados:=msg;
           writeint(pdu_aux.num);
           writestring(':');
           Output PCEP.DT_Req(pdu_aux);
           j:=(j mod JANELA)+1;
           cont_rr:=NUM_ESTACOES
           end
       End
  Else Begin
       writestring('PLIST');
       Envia_PLIST (* P5: PLIST *)
       End
End;
```

```
Procedure Confirmado_Todos(id:integer);
var i,k:integer;
begin
i:=pftx;
k:=JANELA;
writeint(id_est);
writestring(': CONFTODOS:');
While k>0 do   (* A61 A10 *)
  Begin
  if not c_tx[k].confirmou[id] and c_tx[k].ocupado then
    Begin
    c_tx[k].confirmou[id]:=true;
    c_tx[k].falta_conf:=c_tx[k].falta_conf-1;
    writeint(k);
    writestring(':');
    if c_tx[k].falta_conf=0 then
      Begin
      c_tx[k].ocupado:=false;
      c_tx[k].num:=-1
      End
    end;
  k:=k-1
  end;
i:=JANELA;
While not c_tx[pftx].ocupado and (i>0) do
  Begin
  pftx:=(pftx mod janela)+1;
  i:=i-1
  End
End;

Procedure Awunrel_Discac(fonte:integer);
Begin
lista_part[fonte]:=false;  (* A21 *)
num_part:=num_part-1;
cont_ti[fonte]:=-1;  (* A34 *)
Msg_Id(': DISCAC. Envia DL_ABORT.Conf(');
writeint(fonte);
writestring(')')
End;

Procedure Pdu_Nao_Esperada(msg:string; pdu:Tipo_Pdu);
Begin
Trans_Pdu_Str(pdu.cod);
Msg_Id(' em ');
writestring(msg);
writestring(pdu_str);
writestring(' nao esperada da ');
writeint(pdu.fonte);
if pdu.destino=255 then writestring(' por difusao');
End;

Procedure Atualiza_Verifica_Trq;
Begin
if cont_trq>=0 then cont_trq:=cont_trq-1;
if cont_trq=0 Then  (* estourou TR *)
  Begin
  Estourou('TRQ');
```

```
  cont_trq:=TRQ  (* A68 *)
  End
End;

Procedure Ver_Prim_Quadro(var pdu:Tipo_PDU);
Begin
if prim_quadro[pdu.fonte]
  Then Begin
       prim_quadro[pdu.fonte]:=false;
       vr[pdu.fonte]:=pdu.num+1
       end
  Else Ver_Seq_Recebida(pdu.num,pdu.fonte,pdu.cod)
End;

Procedure Env_Stop_TG(i:Integer);
Begin
Msg_Id(': STOP_TRANSMISSION.Ind'); (* A59 *)
cont_tmt:=-1;  (* A28 *)
cont_tr:= TR;  (* A0 *)
num_tent:=1;
cont_trq:=-1;  (* A70 *)
Envia_PLIST;
Cria_TG(pdu_aux,id_est,i,prim_fila,ult_fila,fila_esp);
Output PCEP.DT_Req(pdu_aux);
id_nova_mestra:=i;
End;

Procedure Mostra_Fila;
Begin
Writeint(fila_esp[prim_fila]);
While i<>ult_fila do
  Begin
  writestring('-');
  writeint(fila_esp[i]);
  i:=(i mod NUM_ESTACOES)+1
  End;
writestring(' prim:');
writeint(prim_fila);
writestring(' ult:');
writeint(ult_fila);
End;


INITIALIZE
  TO INICIA
  name iniciando:
  BEGIN
  id_est:=aux_id_est
  END;


Trans { transicoes por variaveis auxiliares }
 From AW_TK_REC
   Provided to_closed
     to INICIA
     name AWTKRECTOCLOSED:
       Begin
       Msg_Id(' Fechou!');
```

```
      Cria_MPDISC(pdu_aux,id_est);
      Output PCEP.DT_Req(pdu_aux);
      to_closed:=false
      End;
  Provided to_open_master
    to OPEN_MASTER
    name AWTKRECOPENMAS:
      Begin
      Msg_Id(': Estourou MAX_TENT, e fila de espera vazia => OPEN_MASTER.');
      to_open_master:=false
      End;

From ESTABLISH
 Provided to_open_master
   to OPEN_MASTER
   name ESTABOPMASTTRP2N1:
     Begin
     WriteString('. Algumas confirmaram:=>MPDL_EST.Conf('); (* A49 *)
     Inicia_Ti_Participantes;  (* A2 *)
     writestring(')');
     Cria_BM(pdu_aux,id_est);
     Output PCEP.DT_Req(pdu_aux);
     Envia_PLIST;
     Inic_Var_Mestra;
     to_open_master:=false
     End;
 Provided to_closed
   to INICIA
   name ESTABCLOSEDTRP2P1:
     Begin
     writestring(' envia MPDL_ABORT.Ind(NMT)');
     to_closed:=false
     End;

From OPEN_REC
 Provided to_closed
  to INICIA
  name OPRECCLOSEDP2:
    Begin
    End;
 Provided to_open_master
  to OPEN_MASTER
  name OPRECOPMAST:
   Begin
   to_open_master:=false
   End;

From OPEN_MASTER
  Provided to_closed
  to INICIA
  name OPENMASTERCLOSED:
   Begin
   writeint(id_est);
   to_closed:=false;
   writestring(': fechou')
   End;
  Provided to_awtkrec
  to AW_TK_REC
```

```
   name OPMASTAWTKRECTMT:
    Begin
    cont_tr:=TR; (* A0 *)
    num_tent:=1;
    to_awtkrec:=false
    End;

TRANS

 FROM INICIA
   TO CLOSED
    name reset:
     Begin
     Inicia_Variaveis;
     END;

 From AW_UN_RELEASE
  When PCEP.DT_Ind
   Provided (pdu.cod=DISCAC) and (num_part>2) and
            Esta_na_Lista(pdu.fonte,lista_part)
   to OPEN_MASTER                  (* NP11 *)           (* P10 *)
   name AWUNRELDISACP10N11:
     Begin
     Awunrel_Discac(pdu.fonte);
     writestring(')');
     Retira_da_Fila(pdu.fonte);
     Envia_PLIST;
     Confirmado_Todos(pdu.fonte) (* A61 A10 *)
     End;
   Provided (pdu.cod=DISCAC) and (num_part<3) and
           Esta_na_Lista(pdu.fonte,lista_part)
     to INICIA  (* CLOSED *)
     name AWUNRELDISACP10P11:
     Begin
     Awunrel_Discac(pdu.fonte);
     writestring(' e nao existe outra conectada.
                  Envia MPDL_ABORT.Ind(LVZ)');
     Cria_MPDISC(pdu_aux,id_est);
     Output PCEP.DT_Req(pdu_aux)
     End;
   Provided (pdu.cod=DISCAC) and not Esta_na_Lista(pdu.fonte,lista_part)
     to same
     name AWUNRELDISACN10:
      Begin
      Msg_Id(': DISCAC de ');
      writeint(pdu.fonte);
      writestring(', mas ela nao esta na lista de participantes!?!')
      End;
   Provided (pdu.cod=TP) and Esta_na_Lista(pdu.fonte,lista_part)
     to same
     name AWUNRELTPP10:
      Begin
      Msg_Id(': MPDL_TOKEN_PLEASE.Ind(');
      writeint(pdu.fonte);
      Coloca_na_Fila(pdu.fonte)
      End;
   Provided (pdu.cod=TP) and not Esta_na_Lista(pdu.fonte,lista_part)
     to AW_UN_RELEASE
```

```
    name AWUNRELTPN10:
     Begin
     Msg_Id(': TP de ');
     writeint(pdu.fonte);
     writestring(' que nao esta na lista de participantes. Ignorado')
     End;
  Provided pdu.cod=RR
   to same
   name AWUNRELRR:
    Begin
    cont_ti[pdu.fonte]:=TI;
    Ver_Descarte(pdu.num,pdu.fonte,true)
    End;
  Provided pdu.cod=SREJ
    to same
    name AWUNRELSREJ:
     Begin
     Retransmite(pdu.num)
     end;
  Provided Otherwise
   to same
   name PNEAWUNREL:
    begin
    Pdu_nao_Esperada('AWAITING_UNIQUE_RELEASE: ',pdu);
    end;
 When UCEP.MPDT
   to same
   name AWUNRELDT:
    Begin
    Transmite_MPI
    end;
 When UCEP.MPDL_AB
  to AWAITING_RELEASE
  name AWUNRELMPDLAB:
    Begin
    {descartar lista de q arm } (* A12 *)
    cont_tr:=TR;              (* A0 *)
    num_tent:=1;
    {finalizar TMT}             (* A28 *)
    for i:=1 to NUM_ESTACOES do cont_ti[i]:=-1; (* A8 *)
    Cria_MPDISC(pdu_aux,id_est);
    Output PCEP.DT_Req(pdu_aux);
    {descartar lista de q esperados} (* A42 *)
    {descartar lista de q rec fora de ordem} (* A43 *)
    falta_conf_conex:=num_part-1
    End;

From AW_TK_REC
 When UCEP.MPDT
  to same
  name AWTKRECMPDT:
   Begin
   Env_Stop_Tx_Tp(id_nova_mestra)
   End;
 When PCEP.DT_Ind
  Provided pdu.cod=BM
   to OPEN_REC
   name AWTKRECBM:
```

```
    Begin
    for i:=1 to NUM_ESTACOES do cont_ti[i]:=-1; (* A8 *)
    num_tent:=1;
    if c_tx[pftx].ocupado
      Then cont_tr:=TR       (* A0 *)
      Else cont_tr:=-1;      (* A4 *)
    cont_tip:=TIP;           (* A17 *)
    id_mestra:=pdu.fonte;    (* A63 *)
    cont_tack:=TACK;         (* A38 *)
    Inic_Var_Rec;            (* A16 *)
    Msg_Id(' Rec BM da ');
    writeint(id_mestra);
    writestring(': MESTRA => RECEPTOR. pftx=');
    writeint(pftx);
    if c_tx[pftx].ocupado then writestring(' OC')
    End;
  Provided pdu.cod=RR
   to same
   name AWTKRECRR:
    Begin
    cont_ti[pdu.fonte]:=TI;
    Ver_Descarte(pdu.num,pdu.fonte,false)
    End;
  Provided pdu.cod=SREJ
    to same
    name AWTKRECSREJ:
     Begin
     for i:=1 to JANELA do (* P22 *)
       if (c_tx[i].ocupado) and (c_tx[i].num=pdu.num) Then
         Begin
         c_tx[i].srej:=true; (* A44 *)
         Cria_TP(pdu_aux,id_est,id_nova_mestra);
         Output PCEP.DT_Req(pdu_aux);
         cont_tkpr:=TKPR
         end
     end;
  Provided Otherwise
   to same
   name PNEAWTKREC:
    begin
    Pdu_Nao_Esperada('AWAITING_TOKEN_RECEIVER: ',pdu)
    end;

From CLOSED
  When UCEP.MPDL_EST
   Provided canal_p_ocupado
    to ESTABLISH_RECEIVER
    name CLOSEDESTABREC:
      Begin
      cont_tr:=TR;   (* A0 *)
      num_tent:=1;
      falta_conf_conex:=NUM_ESTACOES-1;
      Cria_SMPARM(pdu_aux,id_est,id_mestra);
      OUTPUT PCEP.DT_Req(pdu_aux)
      End;
   Provided otherwise
    To ESTABLISH
    name CLOSEDESTAREQMASTER:
```

```
        Begin
        num_tent:=1;
        num_part:=1;
        lista_part[id_est]:=true;
        falta_conf_conex:=NUM_ESTACOES-1;
        cont_tr:=TR;
        Writeln;
        writestring('Difundindo pedido de conexao');
        Cria_SMPARM(pdu_aux,id_est,255);
        OUTPUT PCEP.DT_Req(pdu_aux)
        End;
    When PCEP.DT_Ind
     Provided pdu.cod=SMPARM
      to OPEN_REC
      name CLOSEDOPRECSMPARM:
       Begin
{       output UCEP.MPDL_EST_Ind(pdu.fonte);} (* A48*)
       Inicia_Variaveis; (* A16 *)
       Msg_Id(' abriu como receptora a pedido da ');
       id_mestra:=pdu.fonte;
       writeint(id_mestra);
       Cria_MPARMAC(pdu_aux,id_est,id_mestra);
       output PCEP.DT_Req(pdu_aux); (* MPARMAC *)
       cont_tip:=TIP; (* A17 *)
       cont_tack:=TACK (* A38 *)
       end;
     Provided pdu.cod=MPDISC
      to same
      name CLOSEDMPDISAC:
        Begin
        Cria_MPDISCAC(pdu_aux,id_est,pdu.fonte);
        Output PCEP.DT_Req(pdu_aux);
        canal_p_ocupado:=false
        End;
     Provided pdu.cod=DISC
      to same
      name CLOSEDISAC:
        Begin
        Cria_DISCAC(pdu_aux,id_est,pdu.fonte);
        Output PCEP.DT_Req(pdu_aux)
        End;
     Provided Otherwise
      to same
       name PNECLOSED:
        Begin
        canal_p_ocupado:=true;
        id_mestra:=pdu.fonte
        End;
    When UCEP.MPDL_AB
     Begin
     Msg_Id(': Ja em CLOSED. MPDL_ABORT.Conf.') (* A51 *)
     end;

 From ESTABLISH
    When PCEP.DT_Ind
     Provided (pdu.cod=MPARMAC) and (falta_conf_conex > 1) and
              (not conf_conex[pdu.fonte]) and (pdu.mestra=0)
      to same
```

```
 name ESTMPARACN21N0:
  begin
  Msg_Id(': conf. da ');
  falta_conf_conex:=falta_conf_conex-1;
  conf_conex[pdu.fonte]:=true;        (* A1 *)
  num_part:=num_part+1;
  lista_part[pdu.fonte]:=true;    (* A15 *)
  WriteInt(pdu.fonte);
  end;
Provided (pdu.cod=MPARMAC) and (falta_conf_conex < 2) and
         (not conf_conex[pdu.fonte]) and (pdu.mestra=0)
 to OPEN_MASTER
 name ESTMPARMACN21P0:
  begin
  Msg_Id(': conf. da ');
  falta_conf_conex:=falta_conf_conex-1;
  conf_conex[pdu.fonte]:=true;  (* A1 *)
  num_part:=num_part+1;
  lista_part[pdu.fonte]:=true;  (* A15 *)
  cont_tr:=TRQ;                 (* A68 *)
  cont_tr:=-1;                  (* A6 *)
  cont_tmt:=TMT;                (* A26 *)
  Inicia_Ti_Participantes;      (* A2 *)
  Inic_Var_MESTRA;  (* A13 *)
  WriteInt(pdu.fonte);
  WriteString('. Todas confirmaram:=>MPDL_EST.Conf('); (* A49 *)
  for i:=1 to NUM_ESTACOES do if lista_part[i] Then
    begin
    writestring('-');
    writeint(i)
    end;
  writestring(')');
  Cria_BM(pdu_aux,id_est);
  Output PCEP.DT_Req(pdu_aux);
  Envia_PLIST
  end;
Provided (pdu.cod=MPARMAC) and (pdu.mestra=1)
 to OPEN_REC
 name ESTABMPARMACP21:
  Begin
  Inic_Var_Rec; (* A16 *)
  cont_tip:=TIP; (* A17 *)
  cont_tack:= Tack; (* A38 *)
  End;
Provided (pdu.cod=MPI) or (pdu.cod=TG) or (pdu.cod=BM)
 to ESTABLISH_RECEIVER
 name ESTABPDUMESTRA:
  Begin
  canal_P_ocupado:=true; (* A46 *)
  id_mestra:=pdu.fonte
  End;
Provided (pdu.cod=TP) or (pdu.cod=RR) or (pdu.cod=SREJ)
 to same
 name ESTABAPDUREC:
  Begin
  canal_P_ocupado:=true; (* A46 *)
  id_mestra:=pdu.destino
  End;
```

```
    Provided pdu.cod=PLIST
     to ESTABLISH_RECEIVER
     name ESTPLIST:
      Begin
      canal_P_ocupado:=true; (* A46 *)
      if prim_quadro[pdu.fonte] Then
        Begin
        prim_quadro[pdu.fonte]:=false;
        vr[pdu.fonte]:=pdu.num+1
        end;
      lista_part:=pdu.lista; (* A47 *);
      num_part:=pdu.ultf;
      Msg_Id(': PLIST de ');
      writeint(pdu.fonte);
      writestring(' -*');
      id_mestra:=pdu.fonte;
      for i:=1 to NUM_ESTACOES do if lista_part[i] Then
        begin
        writestring('-');
        writeint(i)
        end;
      if not Esta_na_Lista(id_est,lista_part) Then (* NP24 *)
       Begin
       cont_tr:=TR; (* A4 *)
       Cria_SMPARM(pdu_aux,id_est,id_mestra);
       OUTPUT PCEP.DT_Req(pdu_aux)
       End;
      End;
    Provided (pdu.cod=SMPARM) and (pdu.fonte < id_est) (* P9 *)
     to OPEN_REC
     name ESTSMPARP9:
      Begin
      Msg_Id(': MPDL_EST.Conf. Rec SMPARM de '); (* A49 *)
      writeint(pdu.fonte);
      writeln;
        writestring(' Tambem envia um STOP_TRANSMISSION.Ind, pois ela nao eh a
mestra.'); (* A59 *)
      Inic_Var_Rec; (* A16 *)
      id_mestra:=pdu.fonte;
      Cria_MPARMAC(pdu_aux,id_est,id_mestra);
      Output PCEP.DT_Req(pdu_aux);
      cont_tip:=TIP;  (* A17 *)
      cont_tack:= Tack; (* A38 *)
      End;
    Provided (pdu.cod=SMPARM) and (pdu.fonte > id_est) (* NP9 *)
     to same
     name ESTSMPARN9:
      Begin
      cont_tr:=TR;   (* A4 *)
      num_tent:=1;       (* A4 *)
      Cria_SMPARM(pdu_aux,id_est,255);
      Output PCEP.DT_Req(pdu_aux)
      End;
    Provided pdu.cod=MPDISC
     to CLOSED
     name ESTMPDISC:
      Begin
      Msg_Id(': envia MPDL_ABORT.Ind[USU]');  (* A50 *)
```

```
     Cria_MPDISCAC(pdu_aux,id_est,pdu.fonte);
     Output PCEP.DT_Req(pdu_aux)
     End;
   Provided pdu.cod=DISC
    to CLOSED
    name ESTDISC:
     Begin
     Msg_Id(': DL_ABORT.Ind');  (* A54 *)
     Cria_DISCAC(pdu_aux,id_est,pdu.fonte);
     Output PCEP.DT_Req(pdu_aux)
     End;
   Provided otherwise
    to same
     name PNEESTAB:
     Begin
     Pdu_nao_Esperada('ESTABLISH: ',pdu)
     end;
 When UCEP.MPDL_AB
  to AWAITING_RELEASE
  name ESTMPABORT:
   Begin
   Limpa_Confirm;
   cont_tr:=TR; (* A4 *)
   num_tent := 1;  (* A4 *)
   Cria_MPDISC(pdu_aux,id_est);
   Output PCEP.DT_Req(pdu_aux)
   End;

From OPEN_REC
 When PCEP.DT_Ind

  Provided pdu.cod=SREJ
    to same
    name OPRECSREJ:
     Begin
     for i:=1 to JANELA do (* P22 *)
       if (c_tx[i].ocupado) and (c_tx[i].num=pdu.num) Then
         Begin
         c_tx[i].srej:=true; (* A44 *)
         Cria_TP(pdu_aux,id_est,id_mestra);
         Output PCEP.DT_Req(pdu_aux);
         cont_tkpr:=TKPR
         end;
     end;

  Provided pdu.cod=CM
   to same
   name OPRECCM:
     Begin
     if nao_rec_cm[pdu.fonte] Then (* NP20 *)
       Begin
       nao_rec_cm[pdu.fonte]:=false;
       num_cm_rec:=num_cm_rec+1;
       If num_cm_rec>=(num_part/2)
         Then Begin  (* P19 *)
              cont_tack:=-1; (* A40 *)
              for i:=1 to NUM_ESTACOES do
                 lista_part[i]:=not nao_rec_cm[i];
```

```
            lista_part[id_est]:=true;
            num_part:=num_cm_rec+1;
            Env_3BM_e_Retrans;    (* A68 A40 BM A13 A26 A2 NP23:A45 *)
            cont_tkpr:=-1; (* A31 *)
            prim_fila:=1;  (* A36 *)
            ult_fila:=1;
            writestring('. tornou-se mestra por CM.');
            Envia_PLIST;
            to_open_master:=true
            End
     End
   End;
Provided (pdu.cod=TG) and (pdu.mestra=id_est) (* P14 *)
 to OPEN_MASTER
 name OPRECTGP14:
  Begin
  if c_rx[pfrx[id_mestra],id_mestra].ocupado  (* Envia RR *)
    Then j:=c_rx[pfrx[id_mestra],id_mestra].num
    Else j:=vr[id_mestra];
  Cria_RR(pdu_aux,j,id_est,id_mestra);
  Output PCEP.DT_Req(pdu_aux);
  cont_tkpr:=-1; (* A31 *)
  fila_esp:=pdu.fila; (* A29 *)
  prim_fila:=pdu.primf;
  ult_fila:=pdu.ultf;
  Env_3BM_e_Retrans;
  cont_TIP:=-1;
  num_tent:=1;
  Msg_Id(': TG da ');
  writeint(pdu.fonte);
  writestring('. => MESTRA');
  Writeln;
  writestring('Fila recebida');
  i:=prim_fila;
  While i<>ult_fila do
    Begin
    writestring(':');
    writeint(fila_esp[i]);
    i:=(i mod NUM_ESTACOES)+1
    End;
  writestring(' prim:');
  writeint(prim_fila);
  writestring(' ult:');
  writeint(ult_fila)
  End;
Provided (pdu.cod=TG) and (pdu.mestra<>id_est) (* NP14 *)
 to same
 name OPRECTGN14:
  Begin
  cont_TIP:=TIP;
  num_tent:=1;
  if c_rx[pfrx[id_mestra],id_mestra].ocupado  (* Envia RR *)
    Then j:=c_rx[pfrx[id_mestra],id_mestra].num
    Else j:=vr[id_mestra];
  Cria_RR(pdu_aux,j,id_est,id_mestra);
  Output PCEP.DT_Req(pdu_aux);
  if pediu_ficha Then (* P15 *)
    Begin
```

```
    fila_esp:=pdu.fila; (* A29 *)
    prim_fila:=pdu.primf;
    ult_fila:=pdu.ultf
    End
  End;
Provided pdu.cod=BM
  to same
  name OPRECBM:
   Begin
   cont_tack:=TACK; (* A39 *)
   id_mestra:=pdu.fonte;
   cont_TIP:=TIP;
   num_tent:=1;
   if pediu_ficha Then (* P15 *)
     Begin
     If not(Esta_na_Fila(id_est)) then (* NP16 *)
       Begin
       cont_tkpr:= TKPR; (* A30 *)
       Cria_TP(pdu_aux,id_est,id_mestra);
       Output PCEP.DT_Req(pdu_aux)
       End
     End
   End;
Provided (pdu.cod=PLIST) and (not Esta_na_Lista(id_est,pdu.lista))
  to ESTABLISH_RECEIVER
  name OPRECPLISTN24:
   Begin
   Msg_Id(': PLIST sem estar nela.');
   Ver_Prim_Quadro(pdu);
   lista_part:=pdu.lista;
   num_part:=pdu.ultf;
   for i:=1 to NUM_ESTACOES do  (* A71 *)
     If not lista_part[i] then
       prim_quadro[i]:=TRUE;
   num_tent:=1; (* A0 *)
   cont_tr:=TR;
   cont_tip:=TIP;   (* A35 *)
   num_tent:=1;
   canal_P_ocupado:=true;
   Cria_SMPARM(pdu_aux,id_est,id_mestra);
   OUTPUT PCEP.DT_Req(pdu_aux);
   End;
Provided (pdu.cod=PLIST) and (Esta_na_Lista(id_est,pdu.lista))
  to same
  name OPRECPLISTP24:
   Begin
   Ver_Prim_Quadro(pdu);
   cont_tip:=TIP;   (* A35 *)
   num_tent:=1;
   lista_part:=pdu.lista;
   num_part:=pdu.ultf;
   for i:=1 to NUM_ESTACOES do
     If not lista_part[i] then
       prim_quadro[i]:=TRUE
   End;
Provided pdu.cod=MPDISC
 to INICIA
 name OPRECMPDISC:
```

```
  Begin
  Msg_Id(': MPDL_ABORT.Ind[USU]');  (* A50 *)
  Cria_MPDISCAC(pdu_aux,id_est,pdu.fonte);
  Output PCEP.DT_Req(pdu_aux);
  cont_tack:=-1; (* A40 *)
  {descartar lista de quadros fora de ordem} (* A43 *)
  {Descartar lista de quadros armazenados} (* A12 *)
  cont_tkpr:=-1  (* P15: A31 *)
  End;
Provided pdu.cod=DISC
 to INICIA
 name OPRECDISC:
  Begin
  Cria_DISCAC(pdu_aux,id_est,pdu.fonte);
  Output PCEP.DT_Req(pdu_aux);
  Msg_Id(': DL_ABORT.Ind');  (* A54 *)
  cont_tack:=-1;  (* A40 *)
  {descartar lista de quadros fora de ordem} (* A43 *)
  {Descartar lista de quadros armazenados} (* A12 *)
  cont_tkpr:=-1  (* P15: A31 *)
  End;
Provided pdu.cod=MPI
 to same
 name OPRECMPI:
  Begin
  cont_tip:=TIP;  (* A35 *)
  num_tent:=1;
  l:=pdu.fonte;
  Msg_Id(' rec. msg:"');
  writestring(pdu.dados);
  writestring('" numero ');
  writeint(pdu.num);
  writestring(' da ');
  writeint(l);
  Ver_Prim_Quadro(pdu)
  end;
Provided pdu.cod=SMPARM
 to same name OPRECSMPARM:
   Begin
   cont_tip:=TIP
   End;
Provided pdu.cod=RR
 to same
 name OPRECRR:
  Begin
  Msg_Id(' RR como receptora: ');
  writeint(pdu.fonte);
  Ver_Descarte(pdu.num,pdu.fonte,false)
  End;
Provided pdu.cod=TP
 to same
 name OPRECTP:
  Begin
  End;

Provided Otherwise
 to same
  name PNEOPREC:
```

```
    begin
    Pdu_Nao_Esperada('OPEN_RECEIVER: ',pdu)
    end;
 When UCEP.MPDL_TP
  to same
   name OPRECMPDLTP:
    Begin
    cont_tkpr:= TKPR; (* A23 *)
    Cria_TP(pdu_aux,id_est,id_mestra);
    Output PCEP.DT_Req(pdu_aux)
    End;
 When UCEP.DL_RL
  to same
  name OPRECDLRL:
   Begin
   Cria_RP(pdu_aux,id_est,id_mestra);
   Output PCEP.DT_Req(pdu_aux)
   End;
 When UCEP.MPDT
  to same
  name OPENRECMPDT:
   Begin
   Env_Stop_Tx_Tp(id_mestra)
   End;

From ESTABLISH_RECEIVER

 When PCEP.DT_Ind
  Provided pdu.cod=PLIST
    to same
    name ESTABRECPLIST:
     Begin
     Msg_Id(': PLIST');
     Ver_Prim_Quadro(pdu);
     lista_part:=pdu.lista;
     num_part:=pdu.ultf;
     cont_tip:=TIP;   (* A35 *)
     if not Esta_na_Lista(id_est,pdu.lista) Then  (* NP24 *)
       Begin
       Writestring(' sem estar nela.');
       num_tent:=1; (* A0 *)
       cont_tr:=TR;
       canal_P_ocupado:=true;
       Cria_SMPARM(pdu_aux,id_est,id_mestra);
       OUTPUT PCEP.DT_Req(pdu_aux)
       End
     End;
  Provided (pdu.cod=MPARMAC) and (pdu.mestra=1)
   to OPEN_REC
   name ESTABRECMPARMAC:
    Begin
    Inic_Var_Rec; (* A16 *)
    cont_tip:=TIP; (* A17 *)
    cont_tack:= Tack; (* A38 *)
    End;
  Provided Otherwise
   to same
   name PNEESTABREC:
```

```
    begin
    Pdu_Nao_Esperada('ESTABREC: ',pdu)
    end;

From OPEN_MASTER
  When UCEP.MPDL_RY
   to AW_TK_REC
   name OPMASTREPLAY:
    Begin
    Repeat
      Writeln;
      Writestring('Qual estacao para MPDL_PEPLAY:');
      Readint(k)
    Until (k>=0) and (k<=NUM_ESTACOES);
    Retira_da_Fila(k);
    Env_Stop_TG(k)
    End;
 When UCEP.DL_AB
  to AW_UN_RELEASE
  name OPMASTDLAB:
   Begin
   Repeat
     Writeln;
     Writestring('Qual estacao para DL_ABORT:');
     Readint(i)
   Until (i>=0) and (i<=NUM_ESTACOES);
   Cria_DISC(pdu_aux,id_est,i);
   Output PCEP.DT_Req(pdu_aux);
   cont_tr:=TR;              (* A0 *)
   num_tent:=1
   End;
 When PCEP.DT_Ind
  Provided pdu.cod=RR
   to same
   name OPMASTRR:
    Begin
    cont_ti[pdu.fonte]:=TI;
    Ver_Descarte(pdu.num,pdu.fonte,true)
    End;
  Provided (pdu.cod=TP) and Esta_na_Lista(pdu.fonte,lista_part)
    to same
    name OPMASTTPP10:
     Begin
     Msg_Id(': MPDL_TOKEN_PLEASE.Ind(');
     writeint(pdu.fonte);
     Coloca_na_Fila(pdu.fonte)
     End;
  Provided (pdu.cod=TP) and not Esta_na_Lista(pdu.fonte,lista_part)
    to AW_UN_RELEASE
    name OPMASTTPN10:
     Begin
     Cria_DISC(pdu_aux,id_est,pdu.fonte);
     Output PCEP.DT_Req(pdu_aux);
     cont_tr:=TR; (* A0 *)
     num_tent:=1;
     End;
  Provided pdu.cod=SMPARM
    to same
```

```
   name OPENMASTSMPARM:
    Begin
    If not Esta_na_Lista(pdu.fonte,lista_part) Then  (* NP10 *)
      Begin
      lista_part[pdu.fonte]:=true; (* A15 *)
      num_part:=num_part+1
      End;
    Envia_PLIST;
    Coloca_na_Fila(pdu.fonte);
    Cria_MPARMAC(pdu_aux,id_est,pdu.fonte);
    Output PCEP.DT_Req(pdu_aux)
    End;
 Provided pdu.cod=SREJ
   to same
   name OPMASTSREJ:
    Begin
    Retransmite(pdu.num)
    end;
 Provided pdu.cod=RP
   to same
   name OPMASTRP:
    Begin
    Msg_Id(': MPDL_INDIVIDUAL_RELEASE_PLEASE.Ind(');
    writeint(pdu.fonte);
    WRITESTRING(')')
    End;
 Provided pdu.cod=MPRP
   to same
   name OPMASTMPRP:
    Begin
    Msg_Id(': MPDL_RELEASE_PLEASE.Ind(');
    writeint(pdu.fonte);
    WRITESTRING(')')
    End;
 Provided Otherwise
  to same
  name PNEOPENMAST:
   begin
   Pdu_Nao_Esperada('OPEN_MASTER: ',pdu)
   end;
When UCEP.MPDT
  to same
  name OPMASDT:
   Begin
   Transmite_MPI
   end;
When UCEP.MPDL_AB
  to AWAITING_RELEASE
  name OPMASTMPDLAB:
   Begin
   {descartar lista de q arm } (* A12 *)
   cont_tr:= TR;               (* A0 *)
   num_tent:=1;
   cont_tmt:=-1;               (* A28 *)
   for i:=1 to NUM_ESTACOES do cont_ti[i]:=-1; (* A8 *)
   Cria_MPDISC(pdu_aux,id_est);
   Output PCEP.DT_Req(pdu_aux);
   {descartar lista de q esperados} (* A42 *)
```

```
    {descartar lista de q rec fora de ordem} (* A43 *)
    falta_conf_conex:=num_part-1
    End;
 When UCEP.MPDL_EST
   to same name OPMASTMPDLEST:
    Begin
    Writeln;
    WriteString('Ja aberto: =>MPDL_EST.Conf('); (* A49 *)
    for i:=1 to NUM_ESTACOES do if lista_part[i] Then
      begin
      writestring('-');
      writeint(i)
      end;
    writestring(')')
    End;
 When UCEP.MPDL_TG
   Provided prim_fila=ult_fila
    to same
    name OPMAMPDLTGP13:
     Begin
     Msg_Id(': Fila vazia => MPDL_TOKEN_GIVE.Ind'); (* A58 *)
     cont_TMT:= TMT (* A27 *)
     End;
   Provided otherwise
    to AW_TK_REC
    name OPMASTMPDLTGN13:
     Begin
     Writeln;
     writestring('Fila atual:');
     i:=(prim_fila mod NUM_ESTACOES)+1;
     Mostra_Fila;
     i:=fila_esp[prim_fila];
     prim_fila:=(prim_fila mod NUM_ESTACOES)+1; (* A24 *)
     Env_Stop_TG(i);
     Writeln;
     writestring('Ficou');
     Mostra_Fila
     End;

From AWAITING_RELEASE
 When PCEP.DT_Ind
  Provided (pdu.cod=MPDISCAC) and (num_part>2) (* NP11 *)
   to same
   name AWRELMPDISCAC:
    Begin
    Msg_Id(': MPDISCAC de ');
    j:=pdu.fonte;
    writeint(j);
    if lista_part[j] Then (* P10 *)
      Begin
      lista_part[j]:=false;            (* A18 *)
      num_part:=num_part-1;
      falta_conf_conex:=falta_conf_conex-1;
      writestring(' que foi retirada da lista_part. Num na lista:');
      writeint(num_part)
      End
    End;
  Provided (pdu.cod=MPDISCAC) and (num_part<3) and
```

```
            Esta_na_Lista(pdu.fonte,lista_part) (* P11 *)
    to CLOSED
    name AWRELCLOSED:
     Begin
     Msg_Id(': MPDISCAC de ');
     writeint(pdu.fonte);
     Msg_Id(' Desconectando. Envia MPDL_ABORT.Conf.')
     End;
   Provided pdu.cod=SREJ
     to same
     name AWRELSREJ:
      Begin
      Retransmite(pdu.num)
      end;
   Provided Otherwise
    to same
    name PNEAWREL:
     begin
     Pdu_Nao_Esperada('AWAITING_RELEASE: ',pdu)
     end;

Trans { timer }
 From AWAITING_RELEASE
   to same
   delay(1)
   name AWRELTIMER:
     Begin
     if cont_tr>=0 then cont_tr:=cont_tr-1;
     if cont_tr=0 Then  (* estourou TR *)
       Begin
       Msg_Id(': estourou TR em AWAITING_RELEASE.');
       if num_tent>MAX_TENT
         Then Begin  (* P2 *)
              writestring(' Envia MPDL_ABORT.Conf.');
              to_closed:=true
              End
         Else Begin    (* NP2 *)
              writestring(' Difundindo novo pedido de desconexao.');
              Cria_MPDISC(pdu_aux,id_est);
              Output PCEP.DT_Req(pdu_aux);
              cont_tr:=TR;
              num_tent:=num_tent+1  (* A7 *)
              End
       End; { estourou TR }
     Atualiza_Verifica_Trq
     End;

 From AW_TK_REC
   to same
   delay(1)
   name AWTKRECTIMER:
     Begin
     if cont_tr>=0 then cont_tr:=cont_tr-1;
     if cont_tr=0 Then  (* estourou TR *)
       Begin
       Msg_Id(': estourou TR em AW_TK_REC.');
       if num_tent>MAX_TENT
         Then Begin  (* P2 *)
```

```
            lista_part[id_nova_mestra]:=false;  (* A21 *)
            num_part:=num_part-1;
            cont_ti[id_nova_mestra]:=-1;  (* A34 *)
            Cria_DISC(pdu_aux,id_est,id_nova_mestra);
            Output PCEP.DT_Req(pdu_aux);
            if num_part < 2
              Then Begin    (* P11 *)
                   Msg_Id(': envia MPDL_ABORT.Ind(LVZ)');
                   (* A42 A43 A12 *)
                   to_closed:=true
                   End
              Else Begin     (* NP11 *)
                   Envia_PLIST;
                   Confirmado_Todos(id_nova_mestra);   (* A61 A10 *)
                   if prim_fila=ult_fila
                     Then Begin  (* P13 *)
                          writestring(' envia MPDL_TKEN_GIVE.Ind');
                          cont_TMT:=TMT;  (* A26 *)
                          to_open_master:=true
                          End
                     Else Begin  (* NP13 *)
                          id_nova_mestra:=fila_esp[prim_fila]; (* A24 *)
                          prim_fila:=(prim_fila mod NUM_ESTACOES)+1;
                          writestring(' envia TG p/ ');
                          writeint(id_nova_mestra);
                          cont_tr:=TR; (* A4 *)
                          num_tent:=1;
                          Cria_TG(pdu_aux,id_est,id_nova_mestra,
                                  prim_fila,ult_fila,fila_esp);
                          Output PCEP.DT_Req(pdu_aux)
                          end
                   End
            End
        Else Begin    (* NP2 *)
            Cria_TG(pdu_aux,id_est,id_nova_mestra,prim_fila,
                    ult_fila,fila_esp);
            for i:=1 to 2*numtent do Output PCEP.DT_Req(pdu_aux);
            Writestring(' tenta de novo');
            cont_tr:=TR;  (* A7 *)
            num_tent:=num_tent+1
            End
      End; { estourou TR }
    Atualiza_Verifica_Trq
    End;


From ESTABLISH
  to same
  delay(1)
  name ESTABLISHTIMER:
    Begin
    if cont_tr>=0 then cont_tr:=cont_tr-1;
    if cont_tr=0 Then  (* estourou TR *)
      Begin
      Msg_Id(': estourou TR em ESTABLISH.');
      if num_tent>MAX_TENT
        Then Begin  (* P2 *)
             if falta_conf_conex=NUM_ESTACOES-1  { num_part < 2 }
```

```
              Then to_closed:=true       (* P1 *)
              Else to_open_master:=true  (* NP1 *)
            End
       Else Begin    (* NP2 *)
            writestring(' Difundindo novo pedido de conexao.');
            Cria_SMPARM(pdu_aux,id_est,255);
            Output PCEP.DT_Req(pdu_aux);
            cont_tr:=TR;
            num_tent:=num_tent+1  (* A7 *)
            End
     End; { estourou TR }
   End;

From OPEN_REC
  to same
  delay(1)
  name OPRECTIMER:
   Begin
   if cont_tkpr>=0 Then cont_tkpr:=cont_tkpr-1;
   if (cont_tkpr=0) and pediu_ficha Then
     Begin
     Msg_Id(': Estourou TKPR*************');
     cont_tkpr:=TKPR;
     Cria_TP(pdu_aux,id_est,id_mestra);
     Output PCEP.DT_Req(pdu_aux)
     End;
   if cont_tack>0 Then cont_tack:=cont_tack-1;
   if cont_tack=0 Then                 (* estourou TACK *)
     Begin
     if cont_rr>0 Then
       Begin
       Msg_Id(': TACK ')
       End;
     if c_rx[pfrx[id_mestra],id_mestra].ocupado
       Then Begin
            j:=c_rx[pfrx[id_mestra],id_mestra].num;
            if cont_rr>0 Then Begin writestring('usou prim. fila') End
            End
       Else Begin
            j:=vr[id_mestra];
            if cont_rr>0 Then Begin writestring('usou Vr') End
            End;
     Cria_RR(pdu_aux,j,id_est,id_mestra);
     Output PCEP.DT_Req(pdu_aux);
     cont_tack:=TACK
     end;
   if cont_tip>=0 Then Begin cont_tip:=cont_tip-1 end;
   if cont_tip=0 Then
     Begin            (* estourou TIP *)
     Msg_Id(': TIP. num_tent=');
     writeint(num_tent);
     if num_tent=1
       Then Begin  (* P18 *)
            cont_tack:=-1;      (* A40 *)
            lista_part[id_mestra]:=false;
            num_part:=num_part-1;
            i:=1;
            while not lista_part[i] and (i<NUM_ESTACOES) do i:=i+1;
```

```
     cont_tip:=TIP;
     num_tent:=num_tent+1;
     id_nova_mestra:=i;
     if i<>id_est Then  (* NP17 *)
       Begin
       writestring(' env.CM p/');
       writeint(id_nova_mestra);
       Cria_CM(pdu_aux,id_est,id_nova_mestra);
       Output PCEP.DT_Req(pdu_aux)
       End
     End
Else Begin
     if num_tent > MAX_TENT
       Then Begin               (* P2 *)
            if id_nova_mestra<>id_est
              then begin
                   lista_part[id_nova_mestra]:=false;  (* A21 *)
                   num_part:=num_part-1;
                   Confirmado_Todos(id_nova_mestra);
                   if num_part<2
                     then Begin   (* P11 *)
                          (* A42 A43 A12 *)
                          writestring(' Envia
                                       MPDL_ABORT.Ind(NMT)');
                          to_closed:=true
                          end
                     else Begin   (* NP11 *)
                          i:=1;    (* A32 *)
                          while not lista_part[i] and
                                (i<NUM_ESTACOES) do i:=i+1;
                          cont_tip:=TIP;   (* A17 *)
                          num_tent:=1;
                          if i<>id_est Then  (* NP17 *)
                            Begin
                            id_nova_mestra:=i;
                            writestring(' env.CM p/');
                            writeint(id_nova_mestra);
                            Cria_CM(pdu_aux,id_est,
                                    id_nova_mestra);
                            Output PCEP.DT_Req(pdu_aux)
                            End
                          end
                   end
              else Begin
                   (* A42 A43 A12 *)
                   writestring(' Envia MPDL_ABORT.Ind(NMT)');
                   to_closed:=true
                   end
            End
       Else Begin    (* NP2 *)
            cont_tip:=TIP;
            num_tent:=num_tent+1;
            if (id_nova_mestra<>id_est) Then  (* NP17 *)
              Begin
              writestring(' env.CM p/');
              writeint(id_nova_mestra);
              Cria_CM(pdu_aux,id_est,id_nova_mestra);
              Output PCEP.DT_Req(pdu_aux)
```

```
                  End
                End
            End
    End;
  if cont_tr>=0 then Begin cont_tr:=cont_tr-1 end;
  if cont_tr=0 Then  (* estourou TR *)
    Begin
    if c_tx[pftx].ocupado Then (* NP5 *)
      Begin
      Msg_Id(': estourou TR em OPEN_REC.');
      cont_tkpr:=TKPR;
      Cria_TP(pdu_aux,id_est,id_mestra);
      Output PCEP.DT_Req(pdu_aux)
      End
    End
  End;

From OPEN_MASTER
 to same
 delay(1)
 name OPENMASTERTIMER:
   Begin
   Atualiza_Verifica_Trq;
   if cont_tim>=0 then Begin cont_tim:=cont_tim-1 end;
   if cont_tim=0 Then  (* estourou TIM *)
     Begin
     Estourou('TIM')
     End;
   For i:=1 to NUM_ESTACOES do
     Begin
     If cont_ti[i]>=0 Then Begin cont_ti[i]:=cont_ti[i]-1 end;
     If cont_ti[i]=0 Then (* Estourou TI *)
       Begin
       Msg_Id(': estourou TI da ');
       writeint(i);
       Cria_DISC(pdu_aux,id_est,i);
       Output PCEP.DT_Req(pdu_aux);
       prim_quadro[i]:=true;
       lista_Part[i]:=false;       (* A21 *)
       num_part:=num_part-1;
       Confirmado_Todos(i); (* A61 A10 *)
       Retira_da_Fila(i);      (* P12: A22 *)
       If num_part<2
         Then Begin   (* P11 *)
              writestring('. num.part.<2: Envia MPDL_ABORT_Ind(INF)');
              cont_tr:=-1; (* A6 *)
              to_closed:=true
              End
         Else Envia_PLIST
       End
     End;
   if cont_tmt>=0 Then Begin cont_tmt:=cont_tmt-1 end;
   if cont_tmt=0 Then
     Begin
     Msg_Id(': estourou TMT');
     if prim_fila=ult_fila
       Then cont_tmt:=TMT           (* P13 *)
       Else Begin
```

```
              writestring('. Envia STOP_TRANSMISSION.Ind'); (* A59 *)
              id_nova_mestra:=fila_esp[prim_fila]; (* A24 *)
              prim_fila:=(prim_fila mod NUM_ESTACOES)+1;
              Cria_TG(pdu_aux,id_est,id_nova_mestra,prim_fila,
                      ult_fila,fila_esp);
              Output PCEP.DT_Req(pdu_aux);
              to_awtkrec:=true
              end
       End;

     End;


END; (* ENLACE_ENTITY_BODY *)

MODULE Tipo_MACPHY activity ;
 IP
 PCEP : Array[1..NUM_ESTACOES] Of MPENL_PH_Interface(phy);
END;

BODY MACPHY_BODY FOR Tipo_MACPHY;
 TRANS
   WHEN PCEP[1].DT_Req
    Provided pdu.destino=255
    name DT_1_all:
     BEGIN
     OUTPUT PCEP[2].DT_Ind(PDU);
     if pdu.cod<>SMPARM Then Begin OUTPUT PCEP[3].DT_Ind(PDU) end;
     OUTPUT PCEP[4].DT_Ind(PDU);
     END;
    Provided Otherwise
     name DT_1:
     Begin
     Output PCEP[pdu.destino].DT_Ind(PDU)
     End;
   WHEN PCEP[2].DT_Req
    Provided pdu.destino=255
    name DT_2_all:
     BEGIN
     OUTPUT PCEP[1].DT_Ind(PDU);
     OUTPUT PCEP[3].DT_Ind(PDU);
     OUTPUT PCEP[4].DT_Ind(PDU);
     END;
    Provided Otherwise
     name DT_2:
     Begin
     Output PCEP[pdu.destino].DT_Ind(PDU)
     End;
   WHEN PCEP[3].DT_Req
    Provided pdu.destino=255
    name DT_3_all:
     BEGIN
     OUTPUT PCEP[1].DT_Ind(PDU);
     OUTPUT PCEP[2].DT_Ind(PDU);
     OUTPUT PCEP[4].DT_Ind(PDU);
     END;
    Provided Otherwise
     name DT_3:
```

```
      Begin
      Output PCEP[pdu.destino].DT_Ind(PDU)
      End;
    WHEN PCEP[4].DT_Req
     Provided pdu.destino=255
     name DT_4_ALL:
      BEGIN
      OUTPUT PCEP[1].DT_Ind(PDU);
      OUTPUT PCEP[2].DT_Ind(PDU);
      OUTPUT PCEP[3].DT_Ind(PDU);
      END;
     name DT_4_ALL1:
      BEGIN
      OUTPUT PCEP[1].DT_Ind(PDU);
      if pdu.cod<>TG
       then Begin OUTPUT PCEP[2].DT_Ind(PDU) end (*sempre perde ficha*)
       Else Begin writestring(' perdeu ficha...') end;
      OUTPUT PCEP[3].DT_Ind(PDU);
      END;

     name DT_4_ALL_PERDE:
      BEGIN
      OUTPUT PCEP[1].DT_Ind(PDU);
      OUTPUT PCEP[3].DT_Ind(PDU);
      Writeln;
      Writestring('quadro perdido: 2 nao recebeu quadro ');
      writeint(pdu.num);
      Writestring(' da ');
      writeint(pdu.fonte)
      END;

     name DT_4_ALL2:
      BEGIN
      OUTPUT PCEP[1].DT_Ind(PDU);
      if pdu.cod<>TG
       then Begin OUTPUT PCEP[2].DT_Ind(PDU) end (* perde ficha *)
       Else Begin writestring(' perdeu ficha...') end;
      OUTPUT PCEP[3].DT_Ind(PDU);
      END;
     Provided Otherwise
      name DT_4:
      Begin
      Output PCEP[pdu.destino].DT_Ind(PDU)
      End;

END;

 ModVar
  ENL_MPMPa: Entidade_MPENL;
  ENL_MPMPb: Entidade_MPENL;
  ENL_MPMPc: Entidade_MPENL;
  ENL_MPMPd: Entidade_MPENL;

  MacPhy : Tipo_MACPHY;

INITIALIZE
 name INICIO:
 BEGIN
```

```
   INIT MacPhy WITH MACPHY_BODY;

   INIT ENL_MPMPa WITH ENLACE_ENTITY_BODY(1);
   INIT ENL_MPMPb WITH ENLACE_ENTITY_BODY(2);
   INIT ENL_MPMPc WITH ENLACE_ENTITY_BODY(3);
   INIT ENL_MPMPd WITH ENLACE_ENTITY_BODY(4);

   CONNECT ENL_MPMPa.PCEP TO MacPhy.PCEP[1];
   CONNECT ENL_MPMPb.PCEP TO MacPhy.PCEP[2];
   CONNECT ENL_MPMPc.PCEP TO MacPhy.PCEP[3];
   CONNECT ENL_MPMPd.PCEP TO MacPhy.PCEP[4]

 END;

END.
```

Apêndice B

Acompanhamento de uma sessão de simulação

O simulador Estelle tem uma facilidade opcional: gerar um arquivo com todas as interações disparadas durante uma sessão de simulação. Neste apêndice é mostrada o acompanhamento de uma pequena sessão com o protocolo multiponto a multiponto descrito neste trabalho. A sessão apresentada inicia com um pedido de conexão, realizado pela estação 4. Nesta simulação, a estação 2 foi intencionalmente colocada como sendo problemática, perdendo quadros com freqüência.

O simulador ESTELLE não reconhece *underscore* (_). Assim, os nomes usados na especificação ESTELLE e que usam este caractere, aparecem aqui sem ele.

O número zero (0) no campo *Ident* indica transição do sistema (inicialização, por exemplo) ou da instância MACPHY. Os números de 1 a 4 identificam a instância de módulo Entidade_MPENL em que a transição ocorreu.

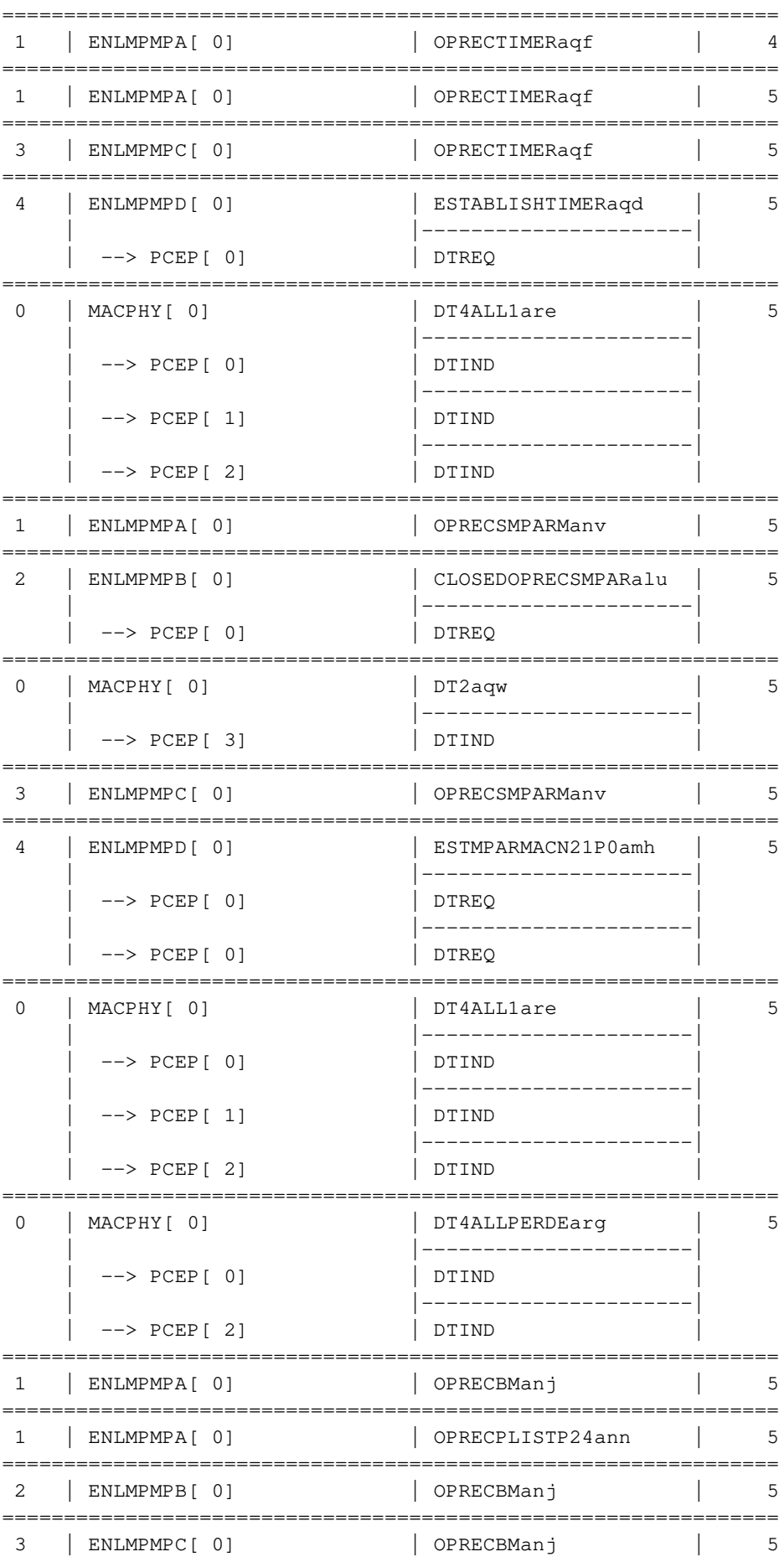
Os sufixos em letras minúsculas são acrescentadas de modo aleatório pelo Compilador ESTELLE, para evitar erros causados por duplicidades de nomes.

```
Ident| Entidade / CEP            | Transição / primitiva| Tempo
 0   | corpo especif.            | INICIOarw            |     0
=============================================================
 0   | MACPHY[ 0]                | LINHA2201aqnaqo      |     0
=============================================================
 0   | ENLMPMPA[ 0]              | INICIANDOajs         |     0
=============================================================
 0   | ENLMPMPB[ 0]              | INICIANDOajs         |     0
=============================================================
 0   | ENLMPMPC[ 0]              | INICIANDOajs         |     0
=============================================================
 0   | ENLMPMPD[ 0]              | INICIANDOajs         |     0
=============================================================
 3   | ENLMPMPC[ 0]              | RESETakk             |     0
=============================================================
 4   | ENLMPMPD[ 0]              | RESETakk             |     0
=============================================================
 2   | ENLMPMPB[ 0]              | RESETakk             |     0
```

Apêndice B . 2

1	ENLMPMPA[0]	RESETakk	0
4	INTERACAO DO USUARIO ->		0
	--> UCEP[0]	MPDLEST	
4	ENLMPMPD[0]	CLOSEDESTAREQMASals	0
	--> PCEP[0]	DTREQ	
0	MACPHY[0]	DT4ALLarc	0
	--> PCEP[0]	DTIND	
	--> PCEP[2]	DTIND	
1	ENLMPMPA[0]	CLOSEDOPRECSMPARalu	0
	--> PCEP[0]	DTREQ	
0	MACPHY[0]	DT1aqs	0
	--> PCEP[3]	DTIND	
3	ENLMPMPC[0]	CLOSEDOPRECSMPARalu	0
	--> PCEP[0]	DTREQ	
0	MACPHY[0]	DT3ara	0
	--> PCEP[3]	DTIND	
4	ENLMPMPD[0]	ESTMPARACN21N0amf	0
4	ENLMPMPD[0]	ESTMPARACN21N0amf	0
4	ENLMPMPD[0]	ESTABLISHTIMERaqd	1
1	ENLMPMPA[0]	OPRECTIMERaqf	1
3	ENLMPMPC[0]	OPRECTIMERaqf	1
1	ENLMPMPA[0]	OPRECTIMERaqf	2
3	ENLMPMPC[0]	OPRECTIMERaqf	2
4	ENLMPMPD[0]	ESTABLISHTIMERaqd	2
4	ENLMPMPD[0]	ESTABLISHTIMERaqd	3
3	ENLMPMPC[0]	OPRECTIMERaqf	3
1	ENLMPMPA[0]	OPRECTIMERaqf	3
3	ENLMPMPC[0]	OPRECTIMERaqf	4
4	ENLMPMPD[0]	ESTABLISHTIMERaqd	4

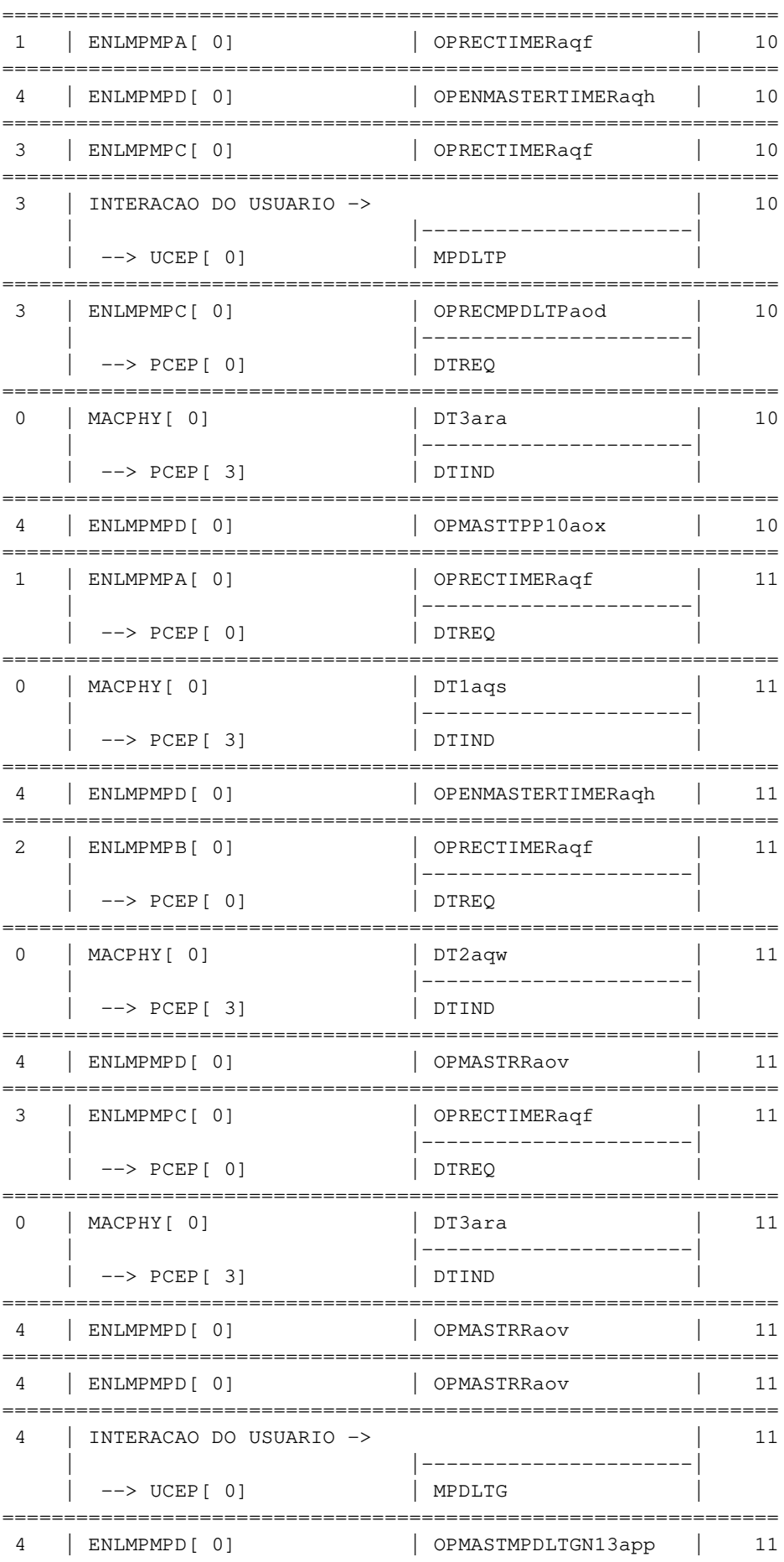
Apêndice B . 3


1 | ENLMPMPA[0] | OPRECTIMERaqf | 4
1 | ENLMPMPA[0] | OPRECTIMERaqf | 5
3 | ENLMPMPC[0] | OPRECTIMERaqf | 5
4 | ENLMPMPD[0] | ESTABLISHTIMERaqd | 5
--> PCEP[0] | DTREQ
0 | MACPHY[0] | DT4ALL1are | 5
--> PCEP[0] | DTIND
--> PCEP[1] | DTIND
--> PCEP[2] | DTIND
1 | ENLMPMPA[0] | OPRECSMPARManv | 5
2 | ENLMPMPB[0] | CLOSEDOPRECSMPARalu | 5
--> PCEP[0] | DTREQ
0 | MACPHY[0] | DT2aqw | 5
--> PCEP[3] | DTIND
3 | ENLMPMPC[0] | OPRECSMPARManv | 5
4 | ENLMPMPD[0] | ESTMPARMACN21P0amh | 5
--> PCEP[0] | DTREQ
--> PCEP[0] | DTREQ
0 | MACPHY[0] | DT4ALL1are | 5
--> PCEP[0] | DTIND
--> PCEP[1] | DTIND
--> PCEP[2] | DTIND
0 | MACPHY[0] | DT4ALLPERDEarg | 5
--> PCEP[0] | DTIND
--> PCEP[2] | DTIND
1 | ENLMPMPA[0] | OPRECBManj | 5
1 | ENLMPMPA[0] | OPRECPLISTP24ann | 5
2 | ENLMPMPB[0] | OPRECBManj | 5
3 | ENLMPMPC[0] | OPRECBManj | 5

Apêndice B . 4

3	ENLMPMPC[0]	OPRECPLISTP24ann	5
2	ENLMPMPB[0]	OPRECTIMERaqf	6
3	ENLMPMPC[0]	OPRECTIMERaqf	6
1	ENLMPMPA[0]	OPRECTIMERaqf	6
4	ENLMPMPD[0]	OPENMASTERTIMERaqh	6
3	ENLMPMPC[0]	OPRECTIMERaqf	7
4	ENLMPMPD[0]	OPENMASTERTIMERaqh	7
2	ENLMPMPB[0]	OPRECTIMERaqf	7
1	ENLMPMPA[0]	OPRECTIMERaqf	7
4	INTERACAO DO USUARIO ->		7
	--> UCEP[0]	MPDT	
4	ENLMPMPD[0]	OPMASDTaph	7
	--> PCEP[0]	DTREQ	
0	MACPHY[0]	DT4ALL2ari	7
	--> PCEP[0]	DTIND	
	--> PCEP[1]	DTIND	
	--> PCEP[2]	DTIND	
1	ENLMPMPA[0]	OPRECMPIant	7
2	ENLMPMPB[0]	OPRECMPIant	7
3	ENLMPMPC[0]	OPRECMPIant	7
2	ENLMPMPB[0]	OPRECTIMERaqf	8
1	ENLMPMPA[0]	OPRECTIMERaqf	8
3	ENLMPMPC[0]	OPRECTIMERaqf	8
4	ENLMPMPD[0]	OPENMASTERTIMERaqh	8
3	ENLMPMPC[0]	OPRECTIMERaqf	9
1	ENLMPMPA[0]	OPRECTIMERaqf	9
2	ENLMPMPB[0]	OPRECTIMERaqf	9
4	ENLMPMPD[0]	OPENMASTERTIMERaqh	9
2	ENLMPMPB[0]	OPRECTIMERaqf	10

Apêndice B . 5

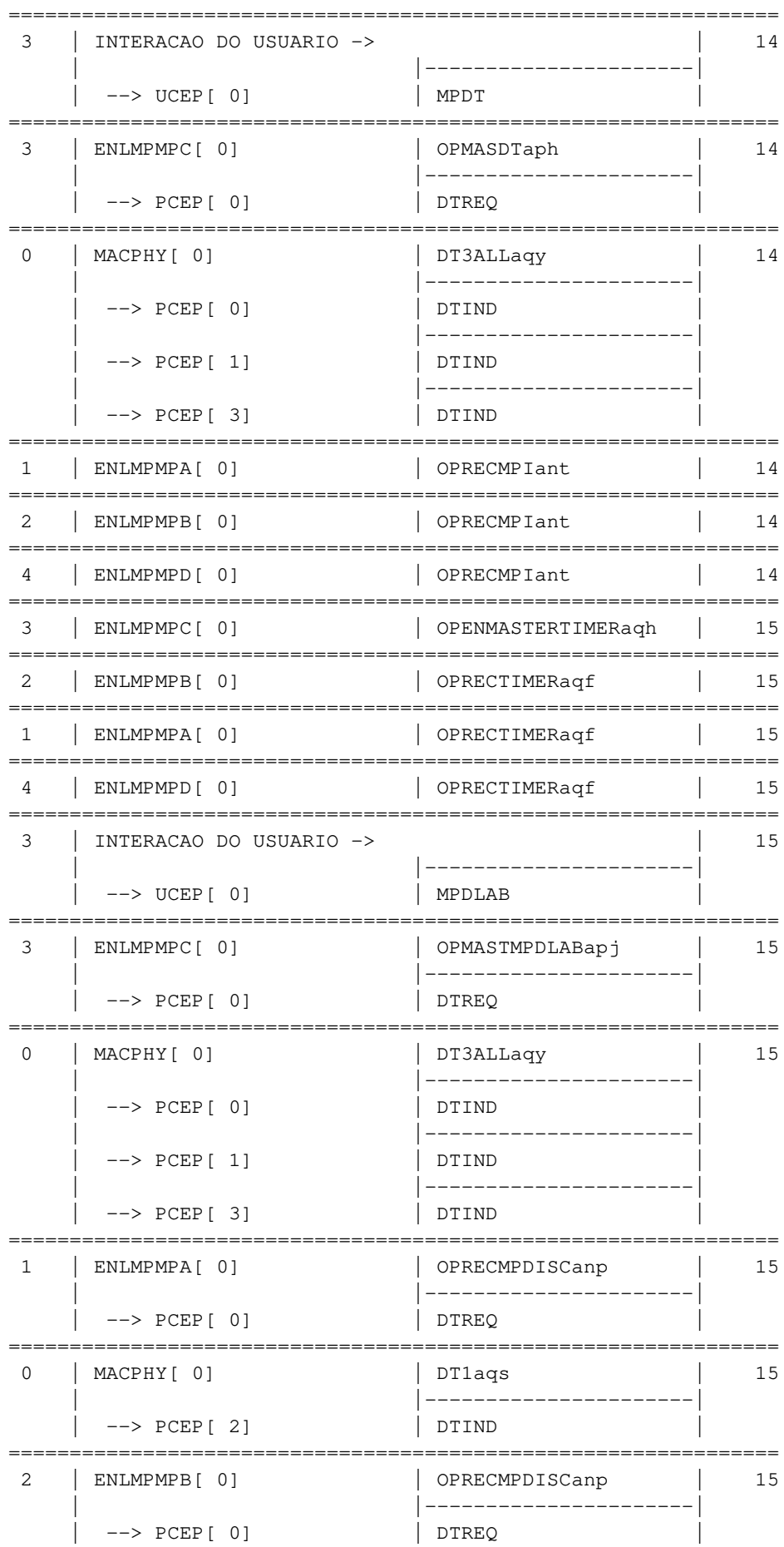

1 | ENLMPMPA[0] | OPRECTIMERaqf | 10
4 | ENLMPMPD[0] | OPENMASTERTIMERaqh | 10
3 | ENLMPMPC[0] | OPRECTIMERaqf | 10
3 | INTERACAO DO USUARIO -> | 10
--> UCEP[0] | MPDLTP
3 | ENLMPMPC[0] | OPRECMPDLTPaod | 10
--> PCEP[0] | DTREQ
0 | MACPHY[0] | DT3ara | 10
--> PCEP[3] | DTIND
4 | ENLMPMPD[0] | OPMASTTPP10aox | 10
1 | ENLMPMPA[0] | OPRECTIMERaqf | 11
--> PCEP[0] | DTREQ
0 | MACPHY[0] | DT1aqs | 11
--> PCEP[3] | DTIND
4 | ENLMPMPD[0] | OPENMASTERTIMERaqh | 11
2 | ENLMPMPB[0] | OPRECTIMERaqf | 11
--> PCEP[0] | DTREQ
0 | MACPHY[0] | DT2aqw | 11
--> PCEP[3] | DTIND
4 | ENLMPMPD[0] | OPMASTRRaov | 11
3 | ENLMPMPC[0] | OPRECTIMERaqf | 11
--> PCEP[0] | DTREQ
0 | MACPHY[0] | DT3ara | 11
--> PCEP[3] | DTIND
4 | ENLMPMPD[0] | OPMASTRRaov | 11
4 | ENLMPMPD[0] | OPMASTRRaov | 11
4 | INTERACAO DO USUARIO -> | 11
--> UCEP[0] | MPDLTG
4 | ENLMPMPD[0] | OPMASTMPDLTGN13app | 11

Apêndice B . 6

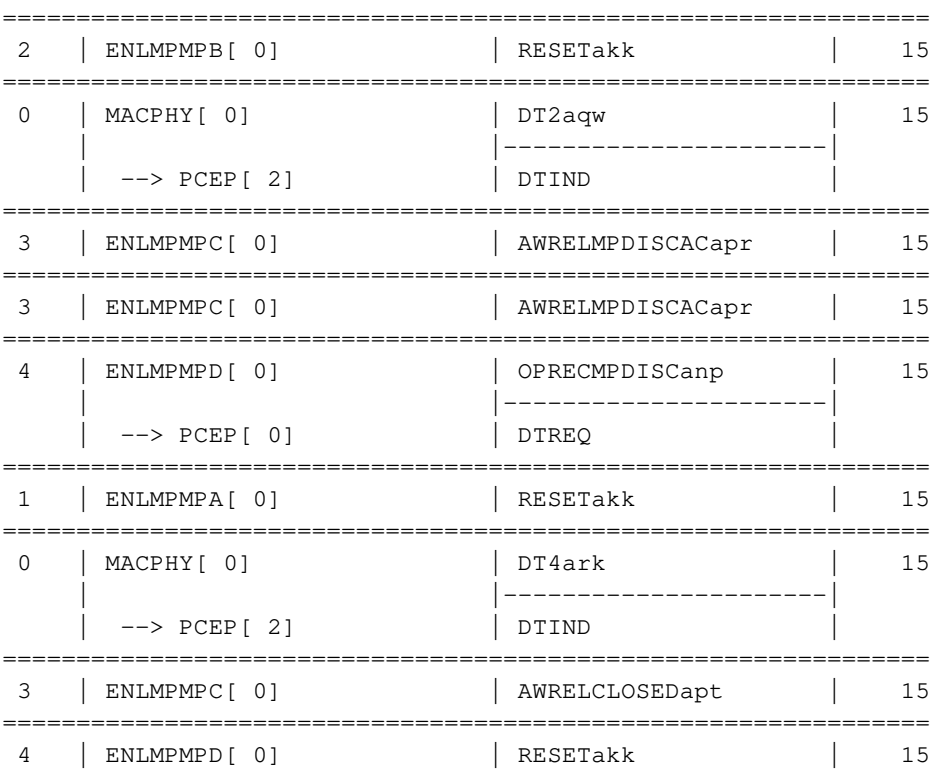
```
     |                          |----------------------|
     |  --> PCEP[ 0]            | DTREQ                |
     |                          |----------------------|
     |  --> PCEP[ 0]            | DTREQ                |
=================================================================
 0   | MACPHY[ 0]               | DT4ALL1are           |    11
     |                          |----------------------|
     |  --> PCEP[ 0]            | DTIND                |
     |                          |----------------------|
     |  --> PCEP[ 1]            | DTIND                |
     |                          |----------------------|
     |  --> PCEP[ 2]            | DTIND                |
=================================================================
 0   | MACPHY[ 0]               | DT4ALL2ari           |    11
     |                          |----------------------|
     |  --> PCEP[ 0]            | DTIND                |
     |                          |----------------------|
     |  --> PCEP[ 2]            | DTIND                |
=================================================================
 1   | ENLMPMPA[ 0]             | OPRECPLISTP24ann     |    11
=================================================================
 1   | ENLMPMPA[ 0]             | OPRECTGN14anh        |    11
     |                          |----------------------|
     |  --> PCEP[ 0]            | DTREQ                |
=================================================================
 0   | MACPHY[ 0]               | DT1aqs               |    11
     |                          |----------------------|
     |  --> PCEP[ 3]            | DTIND                |
=================================================================
 2   | ENLMPMPB[ 0]             | OPRECPLISTP24ann     |    11
=================================================================
 3   | ENLMPMPC[ 0]             | OPRECPLISTP24ann     |    11
=================================================================
 3   | ENLMPMPC[ 0]             | OPRECTGP14anf        |    11
     |                          |----------------------|
     |  --> PCEP[ 0]            | DTREQ                |
     |                          |----------------------|
     |  --> PCEP[ 0]            | DTREQ                |
     |                          |----------------------|
     |  --> PCEP[ 0]            | DTREQ                |
     |                          |----------------------|
     |  --> PCEP[ 0]            | DTREQ                |
=================================================================
 0   | MACPHY[ 0]               | DT3ara               |    11
     |                          |----------------------|
     |  --> PCEP[ 3]            | DTIND                |
=================================================================
 0   | MACPHY[ 0]               | DT3ALLaqy            |    11
     |                          |----------------------|
     |  --> PCEP[ 0]            | DTIND                |
     |                          |----------------------|
     |  --> PCEP[ 1]            | DTIND                |
     |                          |----------------------|
     |  --> PCEP[ 3]            | DTIND                |
=================================================================
 0   | MACPHY[ 0]               | DT3ALLaqy            |    11
     |                          |----------------------|
     |  --> PCEP[ 0]            | DTIND                |
```

```
     |                          |----------------------|
     |  --> PCEP[ 1]            | DTIND                |
     |                          |----------------------|
     |  --> PCEP[ 3]            | DTIND                |
=============================================================
 0   | MACPHY[ 0]               | DT3ALLaqy            |    11
     |                          |----------------------|
     |  --> PCEP[ 0]            | DTIND                |
     |                          |----------------------|
     |  --> PCEP[ 1]            | DTIND                |
     |                          |----------------------|
     |  --> PCEP[ 3]            | DTIND                |
=============================================================
 1   | ENLMPMPA[ 0]             | OPRECBManj           |    11
=============================================================
 1   | ENLMPMPA[ 0]             | OPRECBManj           |    11
=============================================================
 1   | ENLMPMPA[ 0]             | OPRECBManj           |    11
=============================================================
 2   | ENLMPMPB[ 0]             | OPRECBManj           |    11
=============================================================
 2   | ENLMPMPB[ 0]             | OPRECBManj           |    11
=============================================================
 2   | ENLMPMPB[ 0]             | OPRECBManj           |    11
=============================================================
 4   | ENLMPMPD[ 0]             | AWTKRECRRalk         |    11
=============================================================
 4   | ENLMPMPD[ 0]             | AWTKRECRRalk         |    11
=============================================================
 4   | ENLMPMPD[ 0]             | AWTKRECBMali         |    11
=============================================================
 4   | ENLMPMPD[ 0]             | OPRECBManj           |    11
=============================================================
 4   | ENLMPMPD[ 0]             | OPRECBManj           |    11
=============================================================
 4   | ENLMPMPD[ 0]             | OPRECTIMERaqf        |    12
=============================================================
 3   | ENLMPMPC[ 0]             | OPENMASTERTIMERaqh   |    12
=============================================================
 2   | ENLMPMPB[ 0]             | OPRECTIMERaqf        |    12
=============================================================
 1   | ENLMPMPA[ 0]             | OPRECTIMERaqf        |    12
=============================================================
 2   | ENLMPMPB[ 0]             | OPRECTIMERaqf        |    13
=============================================================
 1   | ENLMPMPA[ 0]             | OPRECTIMERaqf        |    13
=============================================================
 3   | ENLMPMPC[ 0]             | OPENMASTERTIMERaqh   |    13
=============================================================
 4   | ENLMPMPD[ 0]             | OPRECTIMERaqf        |    13
=============================================================
 2   | ENLMPMPB[ 0]             | OPRECTIMERaqf        |    14
=============================================================
 4   | ENLMPMPD[ 0]             | OPRECTIMERaqf        |    14
=============================================================
 1   | ENLMPMPA[ 0]             | OPRECTIMERaqf        |    14
=============================================================
 3   | ENLMPMPC[ 0]             | OPENMASTERTIMERaqh   |    14
```

Apêndice B . 8


3 | INTERACAO DO USUARIO -> | 14
--> UCEP[0] | MPDT
3 | ENLMPMPC[0] | OPMASDTaph | 14
--> PCEP[0] | DTREQ
0 | MACPHY[0] | DT3ALLaqy | 14
--> PCEP[0] | DTIND
--> PCEP[1] | DTIND
--> PCEP[3] | DTIND
1 | ENLMPMPA[0] | OPRECMPIant | 14
2 | ENLMPMPB[0] | OPRECMPIant | 14
4 | ENLMPMPD[0] | OPRECMPIant | 14
3 | ENLMPMPC[0] | OPENMASTERTIMERaqh | 15
2 | ENLMPMPB[0] | OPRECTIMERaqf | 15
1 | ENLMPMPA[0] | OPRECTIMERaqf | 15
4 | ENLMPMPD[0] | OPRECTIMERaqf | 15
3 | INTERACAO DO USUARIO -> | 15
--> UCEP[0] | MPDLAB
3 | ENLMPMPC[0] | OPMASTMPDLABapj | 15
--> PCEP[0] | DTREQ
0 | MACPHY[0] | DT3ALLaqy | 15
--> PCEP[0] | DTIND
--> PCEP[1] | DTIND
--> PCEP[3] | DTIND
1 | ENLMPMPA[0] | OPRECMPDISCanp | 15
--> PCEP[0] | DTREQ
0 | MACPHY[0] | DT1aqs | 15
--> PCEP[2] | DTIND
2 | ENLMPMPB[0] | OPRECMPDISCanp | 15
--> PCEP[0] | DTREQ

Apêndice B . 9


2 | ENLMPMPB[0] | RESETakk | 15
0 | MACPHY[0] | DT2aqw | 15
--> PCEP[2] | DTIND
3 | ENLMPMPC[0] | AWRELMPDISCACapr | 15
3 | ENLMPMPC[0] | AWRELMPDISCACapr | 15
4 | ENLMPMPD[0] | OPRECMPDISCanp | 15
--> PCEP[0] | DTREQ
1 | ENLMPMPA[0] | RESETakk | 15
0 | MACPHY[0] | DT4ark | 15
--> PCEP[2] | DTIND
3 | ENLMPMPC[0] | AWRELCLOSEDapt | 15
4 | ENLMPMPD[0] | RESETakk | 15

Apêndice C

Seqüências de teste

Os testes no ambiente do simulador Estelle utilizaram uma seqüência de testes gerada automaticamente, a partir da máquina de estados dos serviços (primitivas). Este apêndice apresenta a seqüência utilizada.

As primitivas que antecedem a "/" são enviadas ao simulador e aqulas que aparecem depois da "/" devem ser recebidas pelo usuário. O evento "r/N" representa o *reset* do simulador, enquanto que Null indica que nenhuma primitiva é enviada e/ou recebida.

```
 1 = r/N, MPDL_Establish_Req/MPDL_Abort_Ind, MPDL_Establish_Req/MPDL_Abort_Ind
 2 = r/N, Null/MPDL_Establish_Ind, Null/MPDL_Abort_Ind
 3 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf, Null/MPDL_Abort_Ind,
     MPDL_Establish_Req/MPDL_Abort_Ind
 4 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf, Null/MPDL_Individual_Abort_Ind,
     MPDL_Establish_Req/MPDL_Abort_Ind
 5 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf, Null/MPDL_Individual_Abort_Ind,
     Null/MPDL_Abort_Ind
 6 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf,
     MPDL_Data_Req/Stop_Transmission_Ind, Null/MPDL_Abort_Ind
 7 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf,
     MPDL_Abort_Req/Stop_Transmission_Ind, Null/MPDL_Abort_Ind
 8 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf, Null/MPDL_Data_Ind,
     Null/MPDL_Abort_Ind
 9 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf, Null/Stop_Transmission_Ind,
     Null/MPDL_Abort_Ind
10 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf,
     MPDL_Individual_Release_Please_Req/Null, Null/MPDL_Abort_Ind
11 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf, MPDL_Release_Please_Req/Null,
     Null/MPDL_Abort_Ind
12 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf, MPDL_Token_Please_Req/Null,
     Null/MPDL_Abort_Ind
13 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf, Null/MPDL_Token_Give_Ind,
     MPDL_Abort_Req/MPDL_Abort_Conf
14 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf, Null/MPDL_Token_Give_Ind,
     MPDL_Abort_Req/MPDL_Abort_Conf
15 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf, MPDL_Abort_Req/MPDL_Abort_Conf,
     MPDL_Establish_Req/MPDL_Abort_Ind
16 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf,
     Stop_Transmission_Response/Stop_Transmission_Ind, Null/MPDL_Abort_Ind
17 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf, MPDL_Replay_Req/Null,
     Null/MPDL_Abort_Ind
```

Apêndice C . 2

```
18 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf,
     MPDL_Token_Give_Req/Stop_Transmission_Ind, Null/MPDL_Abort_Ind
19 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf, MPDL_Data_Req/Null,
     MPDL_Abort_Req/MPDL_Abort_Conf
20 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf, Null/MPDL_Release_Ind,
     MPDL_Abort_Req/MPDL_Abort_Conf
21 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf,
     MPDL_Individual_Abort_Req/MPDL_Individual_Abort_Conf,
     MPDL_Abort_Req/MPDL_Abort_Conf
22 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf,
     Null/MPDL_Individual_Release_Ind, MPDL_Abort_Req/MPDL_Abort_Conf
23 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf, Null/MPDL_Token_Please_Ind,
     MPDL_Abort_Req/MPDL_Abort_Conf
24 = r/N, r/N, MPDL_Establish_Req/MPDL_Abort_Ind
25 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf, r/N,
     MPDL_Establish_Req/MPDL_Abort_Ind
26 = r/N, MPDL_Establish_Req/MPDL_Establish_Conf, r/N,
     MPDL_Establish_Req/MPDL_Abort_Ind
```

Apêndice D

Manual de utilização do sistema de texto-conferência

Este apêndice apresenta um manual de inicialização para a utilização do sistema de texto-conferência implementado neste trabalho, apresentando cópias da tela de vídeo de várias situações possíveis.

Em termos de equipamento (*hardware*) para a utilização do sistema, é necessário a utilização de dois a quatro microcomputadores IBM-PC compatíveis, rodando sistema operacional DOS (3.1 ou superior) e interligados com placas de rede tipo Ethernet.

Em termos de arquivos, são necessários:

- o executável (MPMP.EXE) do sistema de texto-conferência;
- o programa NE1000.COM ou NE2000.COM (*Packet driver*), conforme a compatibilidade das placas Ethernet com a NE1000 ou NE2000;
- o arquivo texto (ASCII) de nome ADDRESS.PKT, com os endereços Ethernet das placas de rede. Este arquivo pode ser criado com qualquer editor de texto, conforme explicado mais adiante.

Primeiramente deve-se instalar o *Packet Driver*, em seguida criar (se não existir) o arquivo ADDRESS.PKT e então chamar o programa MPMP.EXE.

A instalação do *Packet Driver* deve seguir as instruções contidas no pacote. A linha de comando a seguir mostra um exemplo de utilização:

NE1000 0x60 0xINT 0xEND

onde: INT é número da interrupção setada na placa NE1000. Em geral valores entre 3 e 7 são utilizados; END é o endereço (em hexadecimal) da placa (também setado por *jumpers*). Os valores normalmente usados são 300, 320, 340 ou 360. O número 60 representa o número da interrupção (vetor de interrupção - arquitetura dos microprocessadores Intel) utilizada e pode ser substituído por qualquer valor hexadecimal de 60 a 80.

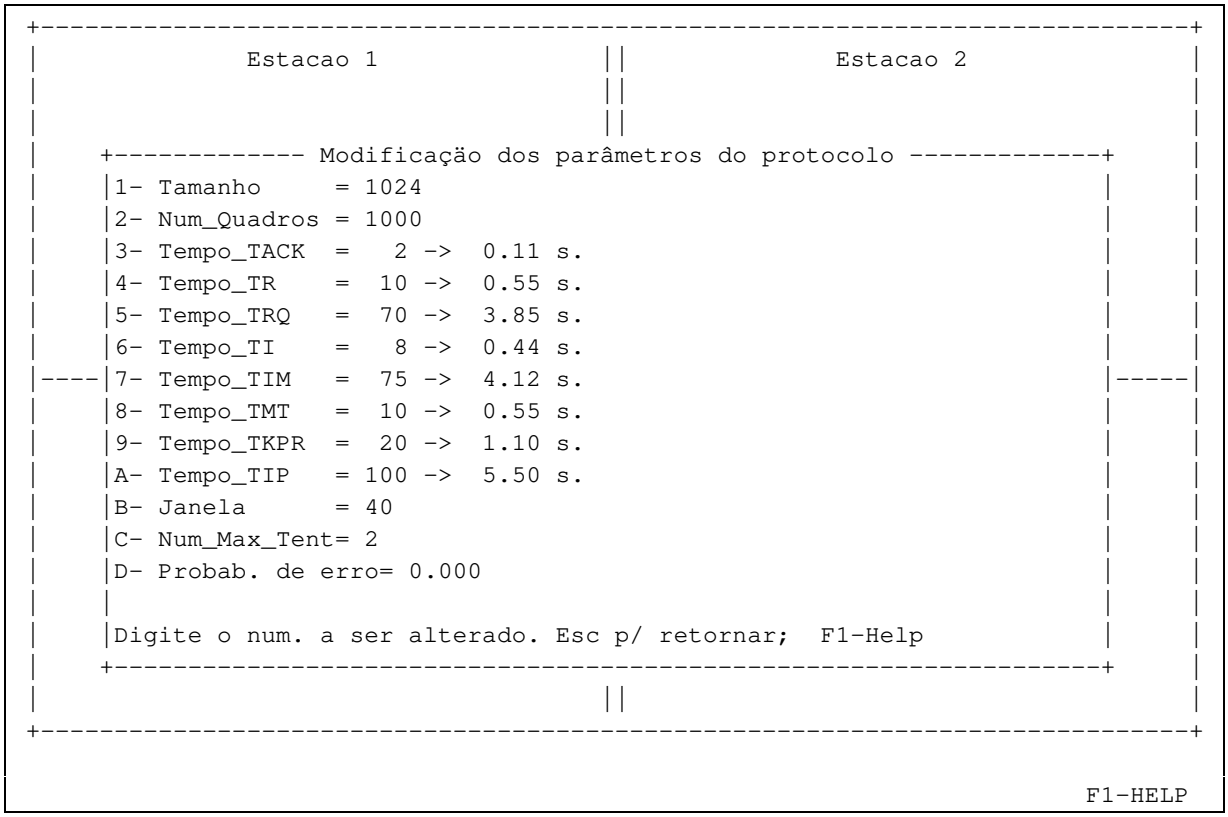
Após digitar a linha acima, o *software* mostra várias informações sobre a placa. Uma informação que deve ser anotada é o endereço Ethernet da placa, composto por seis octetos. Estes octetos são mostrados na tela com seus valores em representação hexadecimal.

De posse dos endereços das placas o arquivo ADDRESS.PKT deve ser criado conforme o formato exemplificado a seguir:

```
0x00,0x00,0x1B,0x1C,0x71,0xE7
0x00,0x00,0x1B,0x34,0x84,0xD6
0x00,0x00,0x1B,0x1C,0x7D,0xEF
0x00,0x00,0x1B,0x1C,0x40,0x00
F
```

Cada linha contém o endereço de uma placa Ethernet. Neste exemplo tem-se quatro estações que poderão participar da conferência. É importante observar a última linha do arquivo, que deve possuir apenas um "F". Este arquivo deve estar no mesmo diretório do programa MPMP.EXE.

Em seguida deve-se chamar o programa de texto conferência digitando-se MPMP, seguido da tecla ENTER. A tela deverá se apresentar conforme a seguir:



Se esta tela não aparecer é possível que tenha ocorrido algum erro nos passos anteriores. Um dos erros detectados pelo sistema é a falta da instalação do *Packet Driver*. Neste caso, a

tela anterior não aparecerá e, em seu lugar, será mostrada a mensagem "PACKET DRIVER não instalado".

Na janela intitulada "Modificaçäo dos parâmetros do protocolo" aparecem os parâmetros para medidas (tamanho, número de quadros e probabilidade de erros) parâmetros do protocolo (demais parâmetros). Os números relativos aos temporizadores são expressos em *Timer-Tick* e aparecem ao lado convertidos para segundos. Os números que aparecem como valores iniciais são adequados para uma conexão. Para editar um valor deve-se digitar o número (índice) que aparece à esquerda do parâmetro. Por exemplo, para mudar o valor do tempo máximo de transmissão (Tempo_TMT) deve-se pressionar a tecla 8. Após digitar o novo valor, digita-se ENTER.

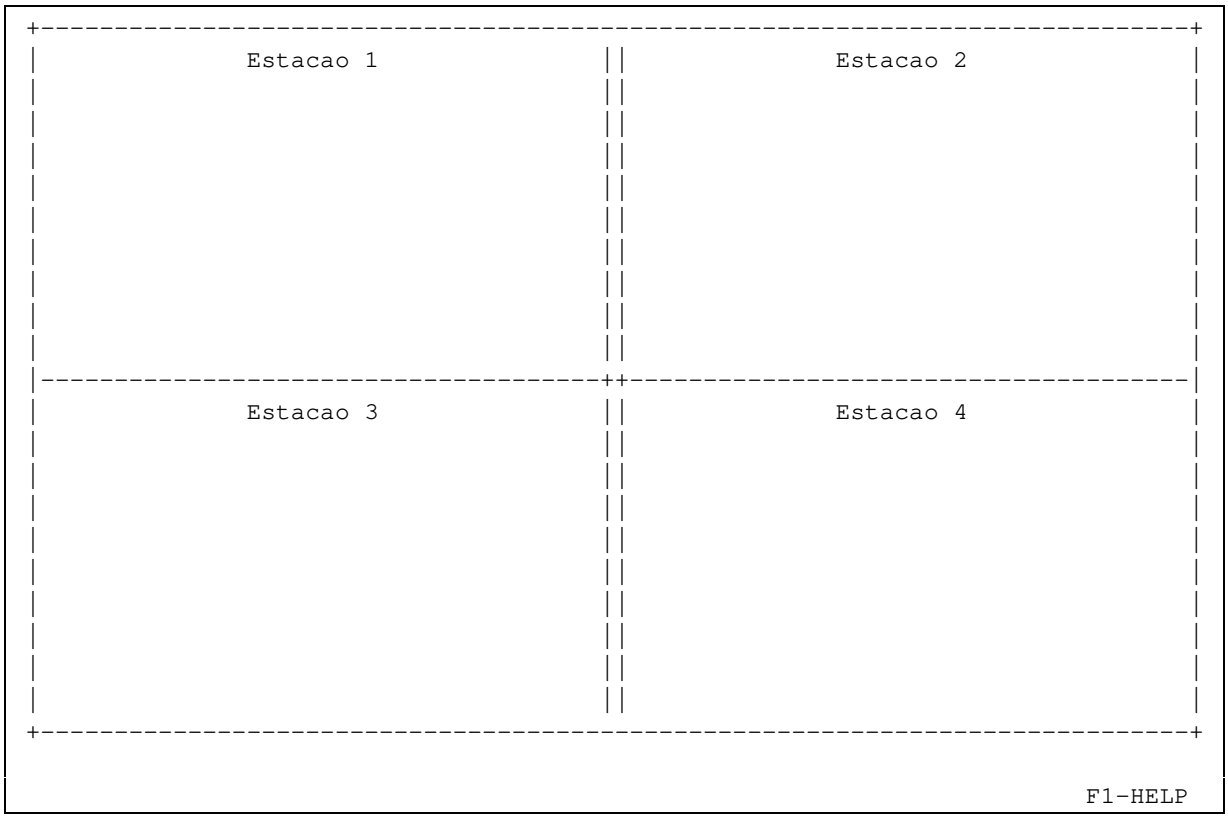
Com esta de edição de parâmetros aberta, a tecla F1 abre uma nova janela com explicações sobre os parâmetros do protocolo, conforme mostrado a seguir:

```
+-------------------------------------------------------------------------------+
+-------------- Descrição dos parametros de tempo do protocolo -----------------+
| Deve ser digitado o número de 'timer ticks'. à direita aparece o equivalente |
|em segundos.                                                                   |
|                                                                               |
|TACK-Tempo entre dois reconhecimentos positivos (RR).                          |
|TR  -Tempo para uma nova tentativa (conexäo, desconexäo, envio de ficha).      |
|TRQ -Tempo para a retransmissäo dos quadros após o último RR em que houve      |
|       reconhecimento de quadros por todos os receptores.                      |
|TI  -Tempo máximo de inatividade dos receptores. O estouro causa a desconexäo  |
|       do receptor que ficou em silêncio. Só existe enquanto a estaçäo for     |
|       mestra (transmissora).                                                  |
|TIM -Tempo de inatividade da mestra, após o qual a mestra irá ou retransmitir  |
|       todos os quadros a espera de reconhecimento, ou irá transmitir a lista  |
|       de participantes da conferência (deve ser menor que TIP).               |
|TMT -Tempo máximo que a estaçäo ficará com a ficha (mestra - transmissora)     |
|TKPR-Após este tempo, uma estaçäo que pediu a ficha e näo foi atendida, irá    |
|       fazê-lo novamente.                                                      |
|TIP -Tempo de inatividade da mestra, controlado pelos receptores. Se ultrapas-|
|       sado, se iniciará um processo de votaçäo para eleger uma nova mestra.   |
|                                                                               |
+-------------------------------------------------------------------------------+
+-------------------------------------------------------------------------------+

                                                                        F1-HELP
```

Para voltar a tela de edição deve-se pressionar F1 novamente. Após a edição dos parâmetros (que podem ser deixados com seus valores *default*), pressiona-se a tecla "Esc". A tela irá se apresentar conforme mostrado a seguir:

Apêndice D . 4

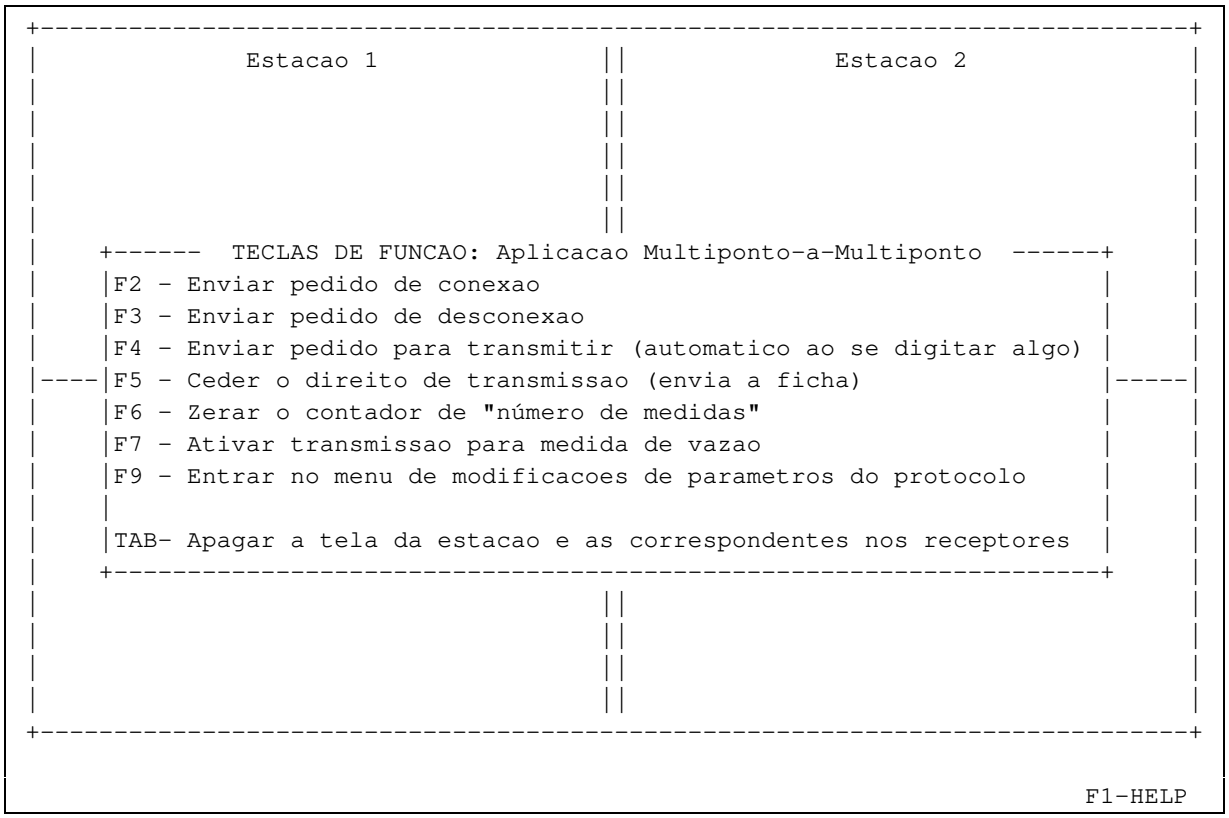


Pode-se notar que a cor de fundo do título da janela de cada estação (na tela de vídeo) é o preto, exceto na janela de uma estação que apresenta título com fundo azul. Este fundo azul indica qual é a estação local. Neste ponto o sistema já está operacional e se for recebido um pedido de conexão, ele será respondido positivamente.

Ao ser iniciada uma conferência as estações que responderem ao pedido de conexão aparecem com o título em fundo vermelho. Quando a estação local for a Mestra da conferência, ou seja, quando seus dados estão sendo imediatamente transmitidos, a cor do título da janela da estação é verde.

As teclas de comando e suas funções podem ser visualizadas pressionando-se a tecla F1. Uma janela com explicação de cada tecla de comando irá se sobrepor à tela. As funções realizadas pelas teclas F2 a F5 são utilizadas durante sessões de conferência (ver quadro a seguir). As teclas F6 e F7 são usadas somente para medidas de vazão. A tecla F9 só pode ser usada quando não houver conferência em andamento para a estação local, ou seja, quando a estação estiver desconectada. A tela de vídeo com a janela explicativas destas teclas se apresentará como mostrado a seguir:

Apêndice D . 5


Estacao 1
Estacao 2
TECLAS DE FUNCAO: Aplicacao Multiponto-a-Multiponto
F2 - Enviar pedido de conexao
F3 - Enviar pedido de desconexao
F4 - Enviar pedido para transmitir (automatico ao se digitar algo)
F5 - Ceder o direito de transmissao (envia a ficha)
F6 - Zerar o contador de "número de medidas"
F7 - Ativar transmissao para medida de vazao
F9 - Entrar no menu de modificacoes de parametros do protocolo
TAB- Apagar a tela da estacao e as correspondentes nos receptores
F1-HELP