RELATÓRIO TÉCNICO

UMA PROPOSTA DE ESPECIFICAÇÃO E IMPLEMENTAÇÃO PARA O RTS NO MHS

Luci Pirmez
Suzan Karina Almada Mendes

NCE-08/90
Abril/90

Universidade Federal do Rio de Janeiro
Núcleo de Computação Eletrônica
Caixa Postal 2324
20001 - Rio de Janeiro - RJ
BRASIL

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
NÚCLEO DE COMPUTAÇÃO ELETRÔNICA

UMA PROPOSTA DE ESPECIFICAÇÃO E IMPLEMENTAÇÃO PARA O RTS NO MHS

RESUMO

O objetivo desse relatório é apresentar uma proposta para a especificação e implementação do RTS, módulo integrante do modelo funcional do MHS proposto pelo NCE, baseado na série X400, que será adotado no projeto REDE-RIO. Este projeto irá permitir a interligação das Universidades do Rio de Janeiro através da RENPAC.

A SPECIFICATION AND IMPLEMENTATION PROPOSAL TO THE RTS ON MHS

ABSTRACT

This report presents a proposal from the NCE for the specification and implementation of the RTS, a module which integrates the functional Model of MHS, based on the X400 series, to be adopted in the project REDE-RIO.

The project will allow the interconnection of the Universities of Rio de Janeiro through the RENPAC.

NCE UFRJ UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
NÚCLEO DE COMPUTAÇÃO ELETRÔNICA

UMA PROPOSTA DE ESPECIFICAÇÃO E IMPLEMENTAÇÃO DO RTS PARA O MHS

I. INTRODUÇÃO

O propósito de um Sistema de Manipulação de mensagens (MHS) é proporcionar recursos e suporte para que seus usuários possam trocar mensagens entre si através de um meio rápido e eficiente. A série X400 define um MHS de acordo com os princípios do modelo OSI e, dessa forma, este sistema pode ser construído sobre qualquer rede física.

Nesse relatório, será apresentado apenas a especificação do RTS, módulo integrante do modelo funcional proposto para o MHS no relatório NCE.

A especificação de um Sistema de Manipulação de Mensagens e sua implementação fazem parte de um projeto denominado REDE-RIO. Este projeto tem como objetivo principal, possibilitar a interconexão dos computadores de grande porte das Universidades do Rio de Janeiro.

A implementação dos serviços a serem oferecidos pela REDE-RIO segue a tendência internacional de basear os desenvolvimentos de software/hardware segundo o modelo OSI/ISO. As sete camadas especificadas por este modelo são: físico, enlace, rede, transporte, sessão, apresentação e aplicação.

As três camadas inferiores já estão definidas pelo CCITT e constituem o protocolo X.25. O padrão X.25 é oferecido pela RENPAC (Rede Nacional de Pacotes) e será utilizado como meio de interconexão entre os vários centros participantes da Rede-Rio. O hardware e software necessários para permitir a interconexão serão adquiridos diretamente dos fabricantes.

O projeto Rede-Rio tem como tarefas o estudo, a especificação e a implementação das seguintes camadas:

- a camada de transporte
- a camada de sessão
- a camada de aplicação
  - o serviço de Manipulação de Mensagem
  - o serviço de Manipulação e Transferência de Tarefas
  - o serviço de Manipulação, Acesso e Transferência de Arquivo
  - o serviço de Terminal Virtual

Na Universidade Federal do Rio de Janeiro, o computador que se utilizará o RENPAC é o VAX 8810.

O objetivo desse relatório é apresentar uma proposta do NCE para a especificação do RTS, módulo integrante do modelo funcional proposto para o MHS. Ele está organizado da seguinte forma:

(1) O RTS no MHS;
(2) O funcionamento do RTS;
(3) Tabela de estados;
(4) Estrutura de dados utilizada.

II - O RTS NO MHS

II.1 - O MODELO FUNCIONAL DO MHS

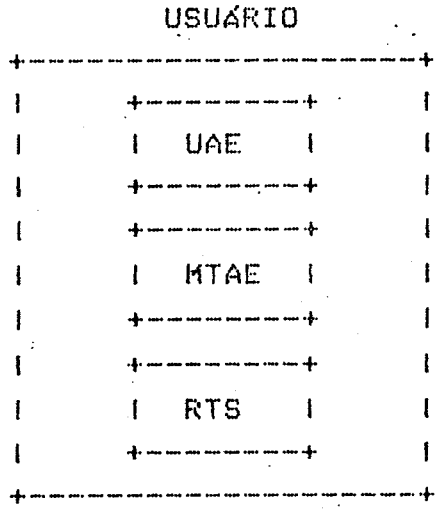
O procedimento básico do funcionamento do MHS (X400) consiste de um ciclo de interrogações. Este ciclo analisa a comunicação com a camada de sessão, analisa a comunicação com o usuário, e ativa quando necessário os procedimentos de transferência de mensagem para a camada de sessão ou para o usuário. Cada análise realizada se refere a uma possível recepção de uma mensagem proveniente da camada de sessão ou do usuário.

O Sistema de Manipulação de Mensagem pode ser dividido em 3 módulos, que são:

A - O módulo UAE que contém funções ativadas pelo usuário que correspondem aos Elementos de Serviços necessários para interagir com o Sistema de Transferência de Mensagem;

B - O módulo MTAE que fornece os meios pelos quais os agentes usuários (UAs) podem trocar mensagens através das redes de comunicação de dados;

C - O módulo RTS que é a parte da entidade de aplicação (AE) responsável pela criação e manutenção de associações entre os pares de AEs.



MODELO FUNCIONAL DO X.400

Em seguida, será detalhado apenas o módulo RTS já que apenas sua especificação será apresentada neste relatório.

II.2 - O MÓDULO RTS

O módulo RTS (Serviço de Transferência Confiável) é a parte da entidade de aplicação (AE) responsável pela criação e manutenção de associações entre os pares de AE e pela transferência confiável das unidades de dados do protocolo de aplicação (APDU) entre estes pares. O RTS realiza todas as funções necessárias para controlar e completar a transferência das unidades de dados do protocolo de mensagem (MPDU); porém, caso o RTS não possa transferir a MPDU, uma indicação de EXCEPTION é emitida.

O gerenciamento de associações, detalhado em seguida, é composto pelos procedimentos de:

- Estabelecimento e término de associações
- Transferência de MPDU's
- Gerenciamento de vez

II.2.1 - ESTABELECIMENTO E TÉRMINO DE ASSOCIAÇÕES

Associações entre duas MTAE's são criadas de acordo com entendimento bilateral que define os seguintes pontos:

(1) O número máximo de associações que podem existir simultaneamente;

(2) Se as associações usadas são monólogos ou diálogos;

(3) Qual MTAE tem a responsabilidade de estabelecer as associações;

(4) Se as associações são permanentemente estabelecidas ou estabelecidas e terminadas segundo a necessidade.

Se associações podem ser estabelecidas segundo a necessidade, o seu estabelecimento é uma decisão local baseada, por exemplo, no tempo, no número e na prioridade das MPDU's a espera de transferência. O critério específico usado refletirá a qualidade do serviço para transferência de mensagem (por exemplo, o tempo médio de entrega de uma mensagem). Se uma associação pode ser terminada quando não é mais necessária (não há mais nenhuma MPDU a transferir), ela não precisa ser terminada imediatamente; ao invés disto, ela pode ser deixada inativa por um período de tempo quando outras MPDUs podem chegar.

OBSERVAÇÕES QUANTO A SELEÇÃO DAS OPÇÕES DO RTS SEGUNDO A NOSSA PROPOSTA:

1) O número máximo de associações que podem existir simultaneamente vai depender da capacidade do sistema. Caso o máximo seja atingido, o RTS não deverá iniciar ou aceitar novas associações;

2) Associações são estabelecidas pelo MTA (agente de transferência de mensagem) que tem uma mensagem a ser transferida;

3) Associações são liberadas quando não necessárias. Elas podem também ser terminadas prematuramente devido a problemas internos do RTS;

4) Quando uma associação do RTS é estabelecida, as seguintes regras se aplicam ao uso de parâmetros:

- Modo de diálogo. O modo diálogo alternado deve ser sempre suportado. Nesta versão, o modo diálogo monólogo não é usado;

- Vez Inicial. A decisão fica a cargo do iniciador da associação. No caso do modo monólogo o iniciador mantém a vez e não há passagem de "Token";

- Protocolo de Aplicação. O identificador de protocolo P1 será suportado. Seu valor é igual a 1;

- Dados do Usuário. Está sempre presente.

5) Não é obrigatório que uma implementação do RTS inicializador contenha os procedimentos de recuperação no caso de sessões abortadas. Da mesma forma, não é obrigatório que uma implementação do RTS respondedor contenha os procedimentos de tratamento de recuperação no caso de sessões abortadas. Caso os procedimentos de recuperação sejam implementados, como é o nosso, a iniciativa para recuperação de uma sessão abortada é sempre do RTS iniciador;

6) O mecanismo de prioridade e o limite de tempo de transferência são considerados decisões do implementador. No nosso caso, não é estipulado tempo de transferência. Apenas as mensagens mais prioritárias são transmitidas antes daquelas de menor prioridade. Já as de mesma prioridade obedecem o algoritmo FIFO (FIRST-IN-FIRST-OUT).

II.2.1.1 - DESCRIÇÃO DA PRIMITIVA OPEN

Um usuário do RTS transmite a primitiva OPEN.Request para estabelecer ou abrir uma associação com um outro usuário do RTS.

A associação pode comportar algumas conexões de sessão em sequência. Sempre que uma conexão de sessão falhar, o RTS abre uma outra nova.

. OPEN.REQUEST
Parâmetros:

(1) Endereço do Respondedor
O Endereço de Sessão do RTS associado ao usuário com o qual se pretende estabelecer a nova associação.

(2) Modo de Diálogo
O tipo de associação a ser aberta, monólogo ou diálogo alternado, de acordo com o entendimento bilateral.

(3) Vez Inicial
O usuário do RTS que terá a vez inicial de comunicação, o iniciador ou o respondedor. A vez será do iniciador, se e somente se ele tiver MPDU's para transmitir.

(4) Protocolo de Aplicação
Designa o protocolo de aplicação que governará a comunicação sobre a associação. No nosso caso, P1.

(5) Dado do Usuário
Dado do usuário associado com a abertura da associação.

```
CHOICE {
   NULL, se nenhuma validação é requerida
   (1) IMPLICIT SET {
          nome MTA (0) IMPLICIT IA5 string
          password (1) any)
       }
```

OPEN.INDICATION
Parâmetros:
(1) Endereço do Iniciador
O Endereço de Sessão do RTS associado ao usuário do RTS que transmitiu a primitiva OPEN.Request.

(2) Modo de Diálogo
O tipo de associação sendo aberta: monólogo ou diálogo alternado.

Este parâmetro transporta o mesmo valor como na primitiva OPEN.Request.

(3) Vez Inicial
O usuário do RTS que terá a vez inicial de comunicação, o iniciador ou o respondedor.

Este parâmetro transporta o mesmo valor como na primitiva OPEN.Request.

(4) Protocolo de Aplicação
Designa o protocolo de aplicação que governará a comunicação sobre a associação.

Este parâmetro transporta o mesmo valor como na primitiva OPEN.Request.

(5) Dado do Usuário
Dado do usuário associado com a abertura da associação.

Este parâmetro transporta o mesmo valor como na primitiva OPEN.Request.

A validação dos parâmetros é realizada, e um OPEN.RESPONSE é transmitido:

. OPEN.RESPONSE
Parâmetros:
(1) Resultado
O resultado ao pedido da associação, aceito ou recusado.

(2) Dado do Usuário
Dado do usuário associado com a abertura da associação.

Este parâmetro somente é indicado se a associação for aceita e sua estrutura é a mesma da descrita anteriormente.

(3) Razão da Recusa
A razão para a recusa da associação.

Este parâmetro somente é indicado se a associação for recusada. Valores definidos são:

. Modo de diálogo inaceitável
. Falha na validação
. Ocupado

. OPEN.CONFIRMATION
Possui os mesmos parâmetros e com os mesmos valores como aqueles da primitiva OPEN.RESPONSE.

II.2.1.2 - DESCRIÇÃO DA PRIMITIVA CLOSE

As associações são liberadas invocando a primitiva CLOSE.REQUEST. O usuário do RTS somente pode fazer isto se tiver a vez de transmissão. Nenhuma das primitivas de CLOSE (CLOSE.REQUEST, CLOSE.INDICATION, CLOSE.RESPONSE e CLOSE.CONFIRMATION) possui parâmetros.

II.2.2 - TRANSFERÊNCIA DE MPDU's

Um usuário do RTS transmite a primitiva TRANSFER.Request para pedir a transferência confiável de uma unidade de dados através da associação. Ele somente pode fazer isso quando o RTS estiver apto a transferir dados na atual conexão de sessão.

II.2.2.1 - DESCRIÇÃO DA PRIMITIVA TRANSFER

. TRANSFER.REQUEST

Parâmetros:

(1) APDU

A MPDU a ser transferida.

(2) Transfer-Time

O período de tempo dentro do qual o RTS deve transferir com sucesso a APDU para o outro usuário do RTS. Determinado por regra local.

A seguir estão as regras para associar MPDU's às associações:

(1) Cada MPDU está associada a uma prioridade baseada no seu tipo, de acordo com a tabela.

Tabela de Prioridade da MPDU

Prioridade	Tipo da MPDU	
1	SMPDU ou UMPDU (*)	Urgente
2	UMPDU	Normal
3	UMPDU	Não-Urgente

* SMPDU - MPDU de Serviço
  UMPDU - MPDU do Usuário

(2) MPDU's são associadas às associações de acordo com suas prioridades.

(3) Algumas associações podem ser usadas para cuidar de MPDU's de mesma prioridade.

(4) Em qualquer associação, as MPDU's de maior prioridade são transmitidas primeiro.

(5) Em qualquer associação, as mensagens de mesma prioridade são transmitidas em FIFO (First-In First-Out).

. TRANSFER.INDICATION
Parâmetros:
(1) APDU
A APDU a ser transferida.
Este parâmetro transporta o mesmo valor como na primitiva TRANSFER.Request.

II.2.2.2 - DESCRIÇÃO DA PRIMITIVA EXCEPTION

Indicação de Fracasso na Transferência.
O RTS transmite a primitiva EXCEPTION.Indication se ele não conseguir completar a transferência de uma APDU como foi pedido.

. EXCEPTION.INDICATION
Parâmetros:
(1) APDU
A APDU que não pode ser transferida dentro do tempo estipulado.

II.2.3 - GERENCIANDO A VEZ

Se a associação é para diálogo, isto é, dois caminhos alternados, o Despachante de Mensagem sem a vez pode enviar uma primitiva TURN-PLEASE.Request cujo parâmetro de prioridade reflete a prioridade mais alta da MPDU a espera de transferência. Quando o Despachante de Mensagem requisita a vez, de maneira que ele possa enviar uma primitiva CLOSE.Request, o parâmetro de prioridade deve refletir a menor prioridade da MPDU que o MTA está preparado para receber antes de terminar a associação.

O Despachante de Mensagem que tem a vez enviará uma primitiva TURN-GIVE.Request em resposta a uma TURN-PLEASE.Indication quando ele não tiver mais nenhuma MPDU para transferir cuja prioridade seja igual ou mais alta que a indicada na TURN-PLEASE.Indication.

II.2.3.1 - DESCRIÇÃO DA PRIMITIVA TURN-PLEASE

Um usuário do RTS transmite a primitiva TURN-PLEASE.Request para pedir a vez. Isto somente é feito se ele não tiver ainda a vez. A vez é pedida ou para transferir dado ou para liberar a associação. O pedido transporta a prioridade da ação a ser tomada de forma que o outro usuário do RTS possa decidir quando ele pode efetivamente ceder a vez.

. TURN-PLEASE.REQUEST

Parâmetros:

(1) Prioridade

A prioridade de ação, governado pela vez, que o usuário do RTS requisitante deseja cumprir.

A prioridade é atribuída a cada usuário do RTS. Às ações de liberar uma associação é de transferir as diversas APDUs serão atribuídas prioridades. A gama de prioridades válidas é uma propriedade do protocolo de aplicação em uso.

. TURN-PLEASE.INDICATION

Possui o mesmo parâmetro, com o mesmo valor da primitiva TURN-PLEASE.Request.

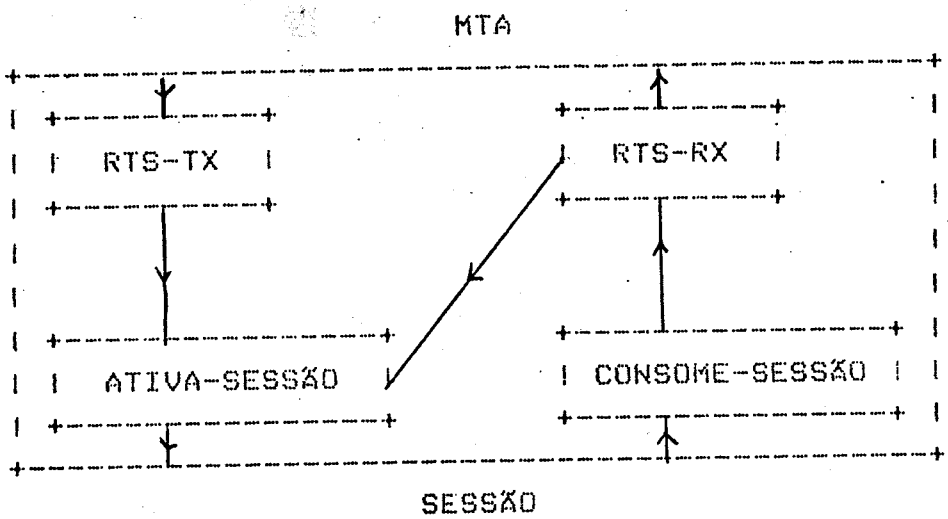
II.2.3.2 - DESCRIÇÃO DA PRIMITIVA TURN-GIVE

Um usuário do RTS transmite a primitiva TURN-GIVE.Request para liberar a vez ao seu par. Ele somente pode fazer isto se tiver a posse da vez.

Estas primitivas TURN-GIVE.Request e TURN-GIVE.Indication não possuem parâmetros.

III. O FUNCIONAMENTO DO RTS

Os serviços do módulo RTS são oferecidos através de funções que são selecionadas pelo módulo MTA. Essas funções executam o mapeamento dos serviços pedidos pelo módulo MTA em serviços oferecidos pela camada de sessão.



MODELO FUNCIONAL DO RTS

O modelo funcional do RTS consiste de quatro funções principais que são:

1. RTS-TX

Função responsável pelo processamento da mensagem proveniente do módulo MTA e por ativar quando solicitado, a função ATIVA-SESSÃO para transmitir uma mensagem à camada de sessão.

Este processamento consiste em mapear solicitações de serviços pelo MTA em solicitações de serviços do RTS para a camada de sessão.

2. ATIVA-SESSÃO

Responsável pelo processamento de mensagens a serem transmitidas para a camada de sessão.

3. RTS-RX

Através dessa função a mensagem recebida pela função CONSOME-SESSÃO é processada e repassada ao módulo MTA, quando necessário. Por outro lado, é ativada, quando solicitada a função ATIVA-SESSÃO para transmitir uma mensagem à camada de sessão.

Este procedimento consiste em mapear os serviços oferecidos pela camada de sessão em serviços oferecidos pelo RTS para o módulo MTA.

4. Se SDTIND
   - monta TRANSFIND

5. Se STPLIND
   - monta TPLEASEIND
   - estado=W-RECEPTOR

6. Se SRELCNF
   - monta CLOSECNF
   - estado=RTS-INATIVO

7. Se STGVIND
   - monta TGIVEIND
   - estado=TRANSMISSOR

8. Monta SCONREQ informando se tem o direito a vez
   - estado=W-SCONCNF
   - ativa-sessão

9. Se estado é W-SRELCNF
   - monta CLOSECNF
   - estado=RTS-INATIVO

   Senão
   - estado=RTS-ABORTO
   - informa que não existe mensagem a ser enviada para o MTA

10.
   - monta SUABREQ
   - ativa sessão
   - monta SCONREQ informando se tem o direito a vez
   - estado=W-SCONCNF
   - ativa sessão

11.
   - monta CLOSECNF
   - estado=RTS-INATIVO

12.
   Se STPLIND
      - monta TPLEASE IND
      - estado=W-SRELCNF

	RTS INATIVO	TRANS-MISSOR	RECEPTOR	W-TRANSMISSOR	W-SCONCNF	W-SRELCNF	W-RECEPTOR	RTS-ABORTO
SCONIND	1	-	-	-	13	-	-	14
SCONCNF	-	-	-	-	2	-	-	-
SRELIND	-	10	3	3	-	11	10	-
SRELCNF	-	-	-	-	-	6	-	-
STGVIND	-	10	7	7	-	11	10	-
STPLIND	-	5	10	10	-	12	-	-
SDTIND	-	10	4	4	-	11	10	-
SPABIND	-	8	8	8	8	11	8	-
SUABIND	-	9	9	9	9	9	9	-

Procedimentos:

1. Se SCONIND
   - monta OPENIND
   - estado=W-OPENRSP

2. Se SCONCNF e resultado é aceito
   - monta OPENRSP com sucesso
   - se tem a vez
   - estado=TRANSMISSOR

   senão
   - estado=RECEPTOR

   senão
   - monta OPENRSP com falha
   - estado=RTS-INATIVO

3. Se SRELIND
   - monta CLOSEIND
   - estado=W-CLOSERSP

EVENTO	DESCRIÇÃO
TGIVEREQ	Liberação da vez para transmissão de dados (TURN-GIVE.REQUEST)
TPLEASEREQ	Pedido de troca de vez para a transmissão de mensagens (TURN-PLEASE.REQUEST)
TRANSFREQ	Pedido de transferência de unidade de dados (TRANSF.REQUEST)
SCONIND	Indicação de conexão de sessão com o par remoto
SCONCNF	Confirmação de conexão de sessão com o par remoto
SRELIND	Indicação de liberação de conexão com o par remoto
SRELCNF	Confirmação de liberação de conexão com o par remoto
STGVIND	Indicação de liberação de vez para a transmissão de dados
STPLIND	Indicação de pedido de troca de vez para transmissão de dados
SDTIND	Indicação de transferência de dados
SPABIND	Indicação de aborto pelo provedor do serviço de sessão
SUABIND	Indicação de aborto pelo usuário (RTS)
SUABREQ	Pedido de aborto de associação com o par remoto

ESTADO	NOME	DESCRIÇÃO
W-CLOSERSP	Estado de Resposta do CLOSE	O RTS espera receber do MTA uma resposta da sua indicação de associação
W-SCONCNF	Estado de Confirmação do S-CONNECT	O RTS local espera receber do remoto uma confirmação do pedido de estabelecimento de conexão
W-SRELCNF	Estado de Espera de Confirmação do S-RELEASE	O RTS local espera receber do remoto uma confirmação do pedido de término de conexão de sessão
RTS-ABORTO	Estado de Aborto	Estado de reestabelecimento de associação

EVENTOS:

Os possíveis eventos do protocolo são gerados por primitivas de serviço ora acionadas pelo MTA ora acionadas pela sessão.

EVENTO	DESCRIÇÃO
OPENREQ	Pedido de estabelecimento de associação com o MTA remoto (OPEN.REQUEST)
OPENRSP	Resposta de estabelecimento de associação com o MTA remoto (OPEN.INDICATION)
CLOSE.REQ	Pedido de término de associação com o MTA remoto (CLOSE.REQUEST)
CLOSE.RSP	Resposta de término de associação com o MTA remoto (CLOSE.RESPONSE)

4. CONSOME-SESSÃO

é responsável em detectar a recepção de mensagens provenientes da camada de sessão.

IV. TABELA DE ESTADOS

Neste tópico é descrito o funcionamento do RTS em termos de tabela de estados. A tabela de estados mostra o estado da associação entre um par de AE's, os eventos que ocorrem, as ações a serem tomadas e o estado resultante.

A seguir, é apresentada a notação usada na tabela de estados, descrevendo os estados e os eventos.

ESTADO	NOME	DESCRIÇÃO
RTS-INATIVO	Estado RTS Inativo	Não existe associação entre o par de AE (Entidade de Aplicação)
TRANSMISSOR	Estado de Transmissão	Tem a vez para transmitir
RECEPTOR	Estado de Recepção	Não tem a vez para transmitir
W-TRANSMISSOR	Estado de Espera de Transmissão	Não tem a vez de transmitir mas, a AE remota solicitou este direito
W-RECEPTOR	Estado de Transição para Recepção	Tem a vez de transmitir, mas a AE remota solicitou este direito
W-OPENRSP	Estado de Resposta do OPEN	O RTS espera receber do MTA uma resposta da sua indicação de estabelecimento de associação

13. Se SCONIND
    - monta SCONRSP com resultado igual a não aceito
    - ativa-sessão

14. Se SCONIND
    - monta SCONRSP com informação sobre o direito a vez
    - ativa sessão

TABELA DE ESTADOS DE TRANSMISSÃO DO RTS

	RTS INATIVO	TRANSMISSOR	RECEPTOR	W-OPENRSP	W-CLOSERSP	W-RECEPTOR	W-TRANSMISSOR
OPENREQ	1	-	-	-	-	-	-
OPENRSP	-	-	-	2	-	-	-
CLOSEREQ	-	5	9	-	-	5	9
CLOSERSP	-	-	-	-	3	-	-
TGIVEREQ	-	7	9	-	-	7	9
TPLEASEREQ	-	9	4	-	8	9	-
TRANSFREQ	-	6	9	-	-	6	9

PROCEDIMENTOS:

1. Se é OPENREQ
    - monta SCONREQ
    - estado=W-SCONCNF
    - ativa sessão

2. Se é OPENRSP e estado é aceito
   - monta SCONRSP sem erro
   - estado=RECEPTOR
   senão
   - monta SCONRSP com erro
   - estado=RTS-INATIVO

3. Se é CLOSERSP
   - monta SRELRSP
   - estado=RTS-INATIVO
   - ativa sessão

4. Se é TPLEASEREQ
   - monta SPLEASEREQ
   - estado=W-TRANSMISSOR
   - ativa sessão

5. Se é CLOSEREQ
   - monta SRELREQ
   - estado=W-SRELCNF
   - ativa sessão

6. Se é TRANSF.REQ
   - monta SDTREQ
   - ativa sessão

7. Se é TGIVEREQ
   - monta SGIVEREQ
   - estado=RECEPTOR
   - ativa sessão

8. Se é TPLEASEREQ
   - monta SPLEASEREQ
   - estado=W-CLOSERSP

9.
   - monta SUABREQ
   - ativa sessão
   - monta SCONREQ informando se tem o direito a vez
   - estado=W-SCONCNF
   - ativa sessão

V - ESTRUTURA DE DADOS

V.1 - ESTRUTURA DE DADOS UTILIZADA INTERNAMENTE

Do ponto de vista interno, o módulo RTS opera sobre uma área de dados e uma entrada na tabela de associações, onde se encontram informações sobre determinada associação.

A área de dados é composta da área da mensagem recebida da caixa postal (mailbox) ou de uma função do MTA e de uma outra área, global a todas as outras camadas. A área global é dinamicamente alocada/dealocada (usando o algoritmo "first fit") a partir de uma região pré-definida de memória, e é utilizada apenas nos campos "DADOS" das primitivas. Esta forma de comunicação entre as camadas é muito conveniente pois, além da flexibilidade quanto ao tamanho dos dados passados, o processo é muito rápido (pois é todo executado em memória).

A tabela de informação denominada de TABRTS consta de 6 campos por associação entre um par de AE. As diversas áreas de informações são alocadas, consecutivamente, em endereços crescentes de memória, e são constituídas pelos seguintes campos:

. SSAP
Identificador do ponto de acesso do serviço de sessão (SSAP) chamado.

. PREF
Referência para o RTS.

. SREF
Referência para a SESSÃO.

. ESTADO
Indica o estado em que se encontra uma determinada associação entre um par de AE's.

. VEZ
Informa se tem o direito a vez para transmitir dados.

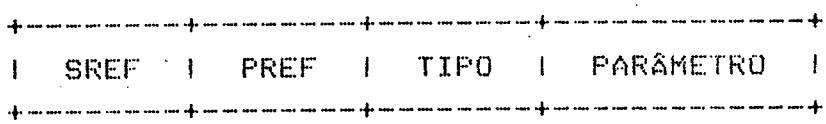
. REINIC
Informa se uma conexão de sessão deve ser restabelecida devido a um aborto.

V.2 - INTERFACES:

V.2.1 - INTERFACE SESSÃO COM O MÓDULO RTS

As informações trocadas entre o módulo RTS e a camada de sessão podem ser implementadas através de dispositivos virtuais de I/O denominado caixa postal (mailbox).

A mensagem que contém as informações consumidas ou produzidas do/para o mailbox apresenta o seguinte formato:



onde:

SREF	-	Referência para a Sessão: identifica a conexão para a Sessão.
PREF	-	Referência para o RTS: identifica a associação para o RTS.
TIPO	-	Identifica a primitiva da camada de sessão..
PARÂMETRO	-	Este campo contém os parâmetros da primitiva de sessão.

Para esta camada, será necessário dois (2) mailboxes para a troca de mensagens com o nível inferior, sessão. Esses mailboxes são denominados de:

(1) MBX-A-IN - é o mailbox responsável pelo envio de informações do módulo RTS para a camada de sessão;

(2) MBX-A-OUT - é o mailbox responsável pelo envio de informações da camada de sessão para o módulo RTS.

No caso da troca de mensagens entre a camada de sessão e o módulo RTS, o campo TIPO da mensagem pode assumir os seguintes valores:

(1) Pedido de conexão de sessão (SCONREQ);
(2) Indicação de conexão de sessão (SCONIND);
(3) Resposta de conexão de sessão (SCONRSP);
(4) Confirmação da conexão de sessão (SCONCNF);
(5) Pedido de transmissão de dados (SDTREQ);
(6) Indicação de transmissão de dados (STDIND);
(7) Pedido de liberação de conexão de sessão (SRELREQ);
(8) Indicação de liberação de conexão de sessão (SRELIND);

(9) Resposta de liberação de conexão de sessão (SRELRSP);
(10) Confirmação de liberação de conexão de sessão (SRELCNF);
(11) Pedido do usuário de aborto de sessão (SUABREQ);
(12) Indicação de aborto de sessão originado do usuário (SUABIND);
(13) Indicação de aborto de sessão originado do provedor (SPABIND).
(14) Pedido ao acesso do "Token" (SPLREQ);
(15) Indicação ao acesso do "Token" (SPLIND);
(16) Acesso ao "Token" concedido (REQUEST) - SGVREQ;
(17) Acesso ao "Token" concedido (INDICATION) - SGVIND.

Quando o campo TIPO da mensagem for igual a (1) ou (2), o campo PARÂMETRO da mensagem apresenta os seguintes subcampos:

IDENT.DA CONEXÃO							
USER CHAMADOR	REF COMUM	REF ADIC.	SSAP CHAMADOR	SSAP CHAMADO	QOS	REQUISITOS	DADOS (*)

No caso do tipo ser igual a (3) ou (4), o campo PARÂMETRO da mensagem apresenta os seguintes subcampos:

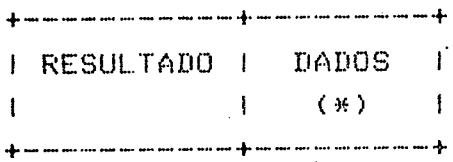
IDENT.DA CONEXÃO							
USER CHAMADOR	REF COMUM	REF ADIC.	SSAP CHAMADO	RESULTADO	QOS	REQUISITOS	DADOS (*)

No caso do tipo ser igual a (5) ou (6), o campo PARÂMETRO da mensagem apresenta os seguintes subcampos:

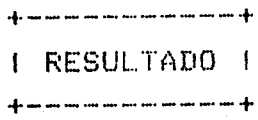
DADOS

No caso do tipo ser igual a (7), (8), (11), (12) o campo PARÂMETRO da mensagem apresenta os seguintes subcampos:

Quando o tipo for igual a (9) ou (10), o campo PARÂMETRO da mensagem apresenta os seguintes subcampos:

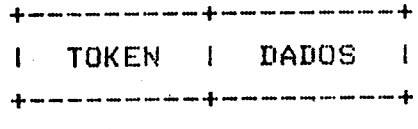


Quando o tipo for igual a (13), o campo PARÂMETRO da mensagem apresenta os seguintes subcampos:



Quando o tipo for igual a (14) ou (15), o campo PARÂMETRO da mensagem apresenta os seguintes subcampos:

No caso do tipo ser igual a (16) ou (17), o campo PARÂMETRO da mensagem apresenta os seguintes subcampos:



Notação:
(*) - Opcional.

. Descrição dos Parâmetros das Primitivas de Sessão

a) Identificador de Conexão

Possibilita aos usuários do Serviço de Sessão identificar a conexão de sessão. Esse identificador é transparente para o provedor do S.S. Os subcampos desse parâmetro são:

a.1) User Chamador/Chamado

Identifica o usuário do serviço de sessão que iniciou/recebeu a conexão. Tamanho máximo: 24 octetos.

a.2) Referência Comum

Tamanho máximo: 14 octetos.

a.3) Referência Adicional

Tamanho máximo: 2 octetos.

b) Resultado da Conexão

Identifica o sucesso ou falha no estabelecimento de uma conexão. Pode ser um dos seguintes valores:

(0) Razão não especificada;

Rejeitado pelo usuário S.S. chamado, quando a razão para a falha neste parâmetro é um dos:

(1) Razão não especificada;
(2) Rejeição pelo usuário S.S. chamado devido a congestão temporária;
(3) Rejeição pelo usuário S.S. chamado. O campo de dados do usuário pode ser usado para prover mais informações.

Rejeitado pelo provedor S.S. quando a razão da falha neste parâmetro é um dos:

(128) Razão não especificada;
(129) Endereço SSAP chamado desconhecido;
(130) Usuário S.S. chamado não conectado ao SSAP;

(131) Congestão no provedor do S.S.;
(132) Versão do protocolo proposto não é suportada.

Somente os valores (0), (1), (2) ou (3) podem estar presentes na resposta. Qualquer um dos valores podem estar presentes na confirmação. Tamanho do campo: 1 octeto.

c) Qualidade de Serviço

O termo qualidade de serviço refere-se a certas características observadas em uma conexão de sessão. Tamanho: 1 octeto.

d) Requisitos de Sessão

é uma lista de unidades funcionais proposta pelo S.S. chamado ou chamador. As unidades funcionais existentes e seus correspondentes bits são:

BIT	UNIDADE FUNCIONAL
1	Half duplex
2	Duplex
3	Dados urgentes
4	Sincronismo secundário
5	Sincronismo principal
6	Resincronização
7	Gerenciamento de atividades
8	Liberação negociada
9	Dados transparentes especiais
10	Exceção
11	Dados transparentes

Quando um determinado bit é igual a zero a unidade funcional não é proposta. No caso de ser igual a 1 (hum) a unidade funcional é proposta. Tamanho do campo: 1 octeto.

e) Dado do Usuário

Contém informações do usuário. Quando este parâmetro faz parte da primitiva S-DATA, seu conteúdo é um número inteiro representando o deslocamento da área, onde se encontra armazenada uma SSDU, dentro da estrutura de alocação. O tamanho da SSDU é um número inteiro de octetos maior que zero.

Esse parâmetro quando utilizado pelas primitivas S-CONNECT, S-RELEASE ou S-TOKEN-PLEASE tem seu tamanho variando de 1 (hum) a 512 (quinhentos e doze) octetos. No caso deste parâmetro ser utilizado pela primitiva U-ABORT seu tamanho varia somente de 1 (hum) a 9 (nove) octetos.

Veja o campo de dados da interface sessão/transporte para mais detalhes de como este campo é utilizado. Tamanho do campo: 8 octetos (2 inteiros).

f) Resultado da Liberação

Esse parâmetro indica se a liberação da sessão é ou não permitida. Seu valor pode ser:

a) Afirmativo;
b) Negativo.

O último valor pode ser utilizado somente se a ficha de liberação está disponível. Tamanho: 1 octeto.

g) Razão do Aborto

Indica a razão do aborto. Seu valor pode ser:

a) Transporte desconectado;
b) Erro de protocolo;
c) Indefinido.

Tamanho do campo: 1 octeto.

h) Endereço do SSAP Chamador

Contém o endereço do SSAP da qual partiu a solicitação da conexão de sessão. Tamanho máximo: 16 octetos.

i) Endereço do SSAP Chamado

Contém o SSAP para o qual a conexão de sessão deve ser estabelecida. Tamanho máximo: 16 octetos.

j) TOKEN

é uma lista de Tokens cujo acesso está sendo requisitado ou cedido para o outro usuário. O valor é a combinação dos tokens de dados, sincronismo principal, sincronismo secundário e de liberação.

. USO DOS SERVIÇOS DE APRESENTAÇÃO E SESSÃO

É descrito neste item como o Serviço de Transferência Confiável (RTS) é suportado pelos Serviços de Sessão e Apresentação do modelo de referência OSI/ISO. O RTS tem o mínimo de procedimentos do nível de apresentação mas, utiliza bastante os serviços de sessão.

O subconjunto dos Serviços de Sessão utilizado na implementação do NCE para esta primeira versão é composto pelas seguintes unidades funcionais:

KERNEL	- Conexão de sessão - Transferência de dados normais - Liberação ordenada - Aborto iniciado pelo usuário - Aborto iniciado pelo provedor
HALF DUPLEX	- Give Tokens - Please Tokens

1. FASE DE ESTABELECIMENTO DE SESSÃO

O RTS inicializa o estabelecimento de uma conexão em resposta ou a um OPEN REQUEST emitido pelo usuário do RTS ou na tentativa de recuperar uma conexão de sessão que foi abortada.

Os dois casos são diferenciados por meio dos dados oriundos do parâmetro DADO DO USUÁRIO.

---) Observações quando a utilização dos parâmetros do S-CONNECT

i. Identificador da Conexão de Sessão

O RTS inicializador fornecerá o identificador da conexão de sessão que identificará unicamente a conexão.

```
Identificador da conexão de sessão:= SEQUENCE {
    Endereço-RTS INICIALIZADOR [0] IMPLICIT T61 string
    [1] IMPLICIT UTC TIME,
    Information Adicional      [2] IMPLICIT T61 string optional}
```

O endereço do RTS inicializador é equivalente ao endereço SSAP chamador que, por sua vez é equivalente ao endereço de transporte.

2. O MHS não utiliza o endereçamento de sessão, isto é, o endereço de sessão não será passado na SPDU de CONNECT do nível de sessão.

O SSAP são passados para o (recebidos do) Nível de Transporte. Eles tem uma correspondência com o endereçamento de transporte.

3. O RTS receptor pode ACEITAR ou REJEITAR o S-CONNECT indication. O parâmetro resultado indicará o que ocorreu. Algumas qualificações da razão de rejeição pode ser colocada no parâmetro DADO DO USUÁRIO. O provedor de sessão pode rejeitar o pedido de conexão em certas circunstâncias.

4. Os parâmetros "Extended Control" e "Optimize Dialogue Transfer" são atribuídos como não desejado. Os parâmetros restantes são atribuídos com os valores "default".

5. Este parâmetro especifica as unidades funcionais utilizadas.

6. Setado com zero.

7. O RTS originador da conexão sempre pedirá a disponibilidade da Ficha (Token) de dados para obter o tipo de associação; o monólogo ou o diálogo alternado.

No caso do OPEN, o RTS originador da conexão especificará qual RTS inicialmente terá a Ficha (Token) para transmissão dos dados de acordo com o parâmetro VEZ fornecida por meio da primitiva OPEN REQUEST.

O RTS originador da conexão deve atribuir todas fichas (Tokens) para o mesmo RTS. A conexão deve ser rejeitada se esta regra for violada. Em qualquer instante, o que retém a Ficha (Tokens) é referenciado com o RTS transmissor e o outro como RTS Receptor.

8. No caso do S-CONNECT request este parâmetro conterá um PCONNECT DATA (APRESENTAÇÃO). No S-CONNECT response, conterá um PACCEPT ou PREFUSE dependendo do parâmetro resultado.

```
PCONNECT::=SEQUENCE {
     [0] Implicit Data Transfer Syntax,
     [1] Implicit pUser-Data Set {
         Checkpoint Size [0] Implicit Integer Default 0
         Window Size [1] Implicit Integer Default 3
         Dialogue Mode [2] Implicit Integer {
            monologue (0), twa (1)} Default Monologue,
     [3] Connection Data,
     application Protocol [4] Implicit Integer {
        p1(1), p3(3)) Default p1))

PACCEPT::=SET {
     [0] Implicit Data Transfer Syntax,
     [1] Implicit pUser Data Set {
         Checkpoint Size [0] Implicit Integer Default 0.
         Window Size [1] Implicit Integer Default 3.
         [2] Connection Data ))

PREFUSE::=SET { [0] Implicit Integer (X409(0))}
```

O único valor definido para o parâmetro Data Transfer Syntax refere a sintaxe de transferência de apresentação especificada na recomendação X409 [4].

```
Connection Data::=CHOICE {
     open[0] any, .. RTS user data
     recover[1] Implicit Session Connection Identifier }

Refuse Reason::=Integer {
     rts Busy (0), Cannot Recover (1), Validation Failure (2),
     unacceptable Dialogue Mode (3) }
```

O parâmetro identificador de conexão de sessão é usado para especificar a conexão de sessão que foi abortada. Com esse parâmetro podemos relacionar a nova sessão com a associação RTS existente.

O parâmetro Window Size permite negociar o número de pontos de sincronismo secundário antes da transferência de dados puder ser suspensa. O RTS receptor deve fornecer na resposta ou confirmação um valor menor que ou igual ao

valor pedido. O valor negociado é utilizado por ambas as direções da transferência.

O parâmetro checkpoint Size permite negociar o tamanho máximo de dados (em unidades de 1024 octetos) que pode ser enviado entre dois pontos de sincronismo principais. O valor zero indica que nenhuma verificação será feita. O RTS receptor fornecerá um valor na resposta ou confirmação que é menor ou igual ao valor pedido. Excepcionalmente no caso que o zero for pedido pelo RTS Transmissor ele pode fornecer qualquer valor. O valor fornecido pelo RTS receptor será o valor máximo e governará ambas as direções da transferência.

2. FASE DE TRANSFERÊNCIA DE DADOS

Quando o usuário do RTS transmissor emite um TRANSFER.REQUEST, o RTS transmissor emite um S-DATA.REQUEST encapsulando a APDU.

Um S-U-ABORT REQUEST é emitido pelo RTS receptor quando um problema é detectado durante a recepção da APDU.

---) Requisitando o controle da conexão da sessão.

Quando o usuário do RTS transmissor emite um TURN-PLEASE REQUEST, o RTS emite um S-TOKEN-PLEASE REQUEST. Na recepção do S-TOKEN-PLEASE INDICATION, o RTS transmissor emite um TURN-PLEASE INDICATION para o usuário do RTS.

OBSERVAÇÕES QUANTO A UTILIZAÇÃO DOS PARÂMETROS DO S-TOKEN-PLEASE

1. Tokens - O RTS receptor somente solicita o token de dados. No MHS, o RTS transmissor sempre fornecerá todos os outros tokens disponíveis quando emitir um S-GIVE REQUEST.

2. No campo SS-USER-DATA é armazenado o parâmetro prioridade da primitiva TURN-PLEASE REQUEST.

OBSERVAÇÕES QUANTO A UTILIZAÇÃO DOS PARÂMETROS DO S-U-ABORT REQUEST

1. O parâmetro USERDATA contém informações do aborto.

```
INFORMAÇÃO ABORTO::= SET(
   [0] IMPLICIT RAZÃOABORTO OPTIONAL,
   PARÂMETROREFLETIDO [1] IMPLICIT BIT STRING OPTIONAL)

RAZÃO ABORTO::= INTEGER(
   PROBLEMALOCAL (0),
   PARÂMETROINVÁLIDO (1), .. PARÂMETRO REFLETIDO FORNECIDO
   ATIVIDADENÃORECONHECIDA (2),
   PROBLEMATEMPORÁRIO (3), .. O RTS NÃO PODE ACEITAR A SESSÃO POR UM
                              PERÍODO DE TEMPO
   ERRODEPROTOCOLO (4), .. ERRO DE PROTOCOLO DO RTS ....)
```

--) Passando o controle da sessão para o RTS receptor

Quando o usuário do RTS emite um TURN-GIVE-REQUEST, o RTS transmissor dará o controle da sessão para o outro RTS usando a primitiva S-GIVE REQUEST.

3. FASE DE LIBERAÇÃO DA CONEXÃO DE SESSÃO

1. Liberação ordenada

O RTS transmissor emite um S-RELEASE REQUEST em resposta a CLOSE.REQUEST. Na recepção de um S-RELEASE indication, o RTS emite um CLOSE.indication e um S-RELEASE.RESPONSE.

2. Aborto iniciado pelo usuário

Se um RTS detecta um problema sério, é emitido um S-U-ABORT request. O parâmetro razão deve ser fornecido para propósito de diagnósticos e, se o aborto foi provocado pela recepção de um parâmetro inválido do MHS, este parâmetro deve ser fornecido.

PERFIL PARA O PROTOCOLO RTS

1. Uma associação de RTS corresponde a uma ou mais conexões de sessão consecutivas. A primeira é aberta com Dados Conexão do tipo abertura; as subsequentes são abertas com Dados Conexão do tipo Recuperação;

2. Caso os procedimentos de recuperação sejam implementados, como é o nosso caso, a iniciativa para recuperação de uma sessão abortada é sempre do RTS iniciador;

3. Não são suportadas sincronização e gerenciamento de atividade na nossa implementação.

V.2.2 - INTERFACE MTA COM O MÓDULO RTS

As informações trocadas entre o módulo RTS e o MTA são passadas através dos parâmetros de funções.

Estes parâmetros conterão o endereço da mensagem propriamente dita. A mensagem apresenta o seguinte formato:

. EVENTORTS - Informa o tipo da primitiva
. SSAP - Identificador do ponto de acesso do serviço de sessão
. PRIMITIVA - Este campo contém os parâmetros da primitiva do RTS

No caso de troca de informação entre o módulo MTA e o módulo RTS, o campo EVENTORTS da mensagem pode assumir os seguintes valores:

(01) Pedido de estabelecimento de associação (OPENREQ);
(02) Indicação de estabelecimento de associação (OPENIND);
(03) Resposta de estabelecimento de associação (OPENRSP);
(04) Confirmação de estabelecimento de associação (OPENCNF);
(05) Pedido de liberação de associação (CLOSEREQ);
(06) Indicação de liberação de associação (CLOSEIND);
(07) Resposta de liberação de associação (CLOSE.RSP);
(08) Confirmação de liberação de associação (CLOSECNF);
(09) Pedido de troca de vez (TPLEASEREQ);

(10) Indicação de troca de vez (TPLEASEIND);
(11) Pedido para dar a vez (TGIVEREQ);
(12) Indicação para dar a vez (TGIVEIND);
(13) Pedido de transferência de dados (TRANSFREQ);
(14) Indicação de transferência de dados (TRANSFIND);
(15) Indicação de fracasso da transferência (EXCEPTIND).

Quando o campo EVENTORTS for igual a (1) ou (2), o campo PRIMITIVA da mensagem apresenta os seguintes subcampos:

ENDSESSÃO	DIALOGO	INIC TURN	APL	USER DATA

No caso do EVENTORTS for igual a (3) ou (4), o campo PRIMITIVA da mensagem apresenta os seguintes subcampos:

ESTADO	USER DATA	RAZÃO

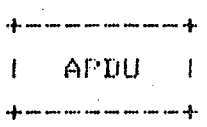
Quando o EVENTORTS for igual a (9) ou (10), o campo PRIMITIVA da mensagem apresenta os seguintes subcampos:

PRIORIDADE

Quando o EVENTORTS for igual a (13), o campo PRIMITIVA da mensagem apresenta os seguintes subcampos:

APDU	TEMPO TRANSF

No caso do EVENTORTS for igual a (14) ou (15) o campo PRIMITIVA da mensagem apresenta os seguintes subcampos:



A descrição dos parâmetros das primitivas encontra-se no ítem II.2.

VI - CONCLUSÃO:

A proposta de especificação e implementação do RTS pelo NCE pode ser implementada em diversos sistemas operacionais, visto que as soluções adotadas não são particulares ao sistema operacional utilizado em nossa instalação, o VMS.

Apesar disso, existem problemas que, devido a sua natureza, somente admitem soluções dependentes do sistema operacional, como é o caso da comunicação entre processos. Nestes casos, as soluções são postas em um módulo separado com uma interface bem definida de modo a facilitar as modificações em caso de transferência de sistema operacional.

Agrupando-se a descrição funcional do MHS às especificações dos módulos RTS, MTAE e UAE obtem-se a descrição completa do Sistema de Manipulação de Mensagens (MHS) proposto pelo NCE para ser adotado no projeto REDE-RIO.

BIBLIOGRAFIA

1. ISO 7498
Information Processing Systems - Open Systems Interconnection - Basic Reference Model.

2. CCITT X400-X430
Message Hondling Systems.

3. HSY, C.Y.
"Estudo da Série de Recomendações X400"
Relatório Técnico NCE 0187

4. GIOZZA, E. et all
"Redes Locais de Computadores
Protocolos de Alto Nível e Avaliação de Desempenho"
Mac-Graw-Hill 1986

5. PIRMEZ, L. et Mendes, S.
"Uma Visão do Funcionamento do Sistema de Mensagem (X400) do NCE/UFRJ"
Relatório Técnico NCE 05/89

6. MENDES, S. et PIRMEZ, L.
"Análise e Especificação do Agente Usuário (UA) no Sistema de Manipulação de Mensagens (MHS)"
Relatório Técnico NCE a ser publicado

7. MENDES, S. et PIRMEZ, S.
"Uma Proposta de Implementação da Entidade de Agente Usuário (UAE) no Contexto de Sistemas de Manipulação de Mensagens (MHS)"
Relatório Técnico NCE a ser publicado

8. PIRMEZ, L. et MENDES, S.
"Análise e Especificação do Agente de Transferência de Mensagem (MTA) no Sistema de Manipulação de Mensagens (MHS)"
Relatório Técnico NCE 04/90

9. PIRMEZ, L. et MENDES, S.
"Uma Proposta de Implementação do Agente de Transferência de Mensagem (MTA) no Contexto de Sistemas de Manipulação de Mensagens (MHS)"
Relatório Técnico NCE a ser publicado

10. PIZZORNO, J.; MENDES, S. et PIRMEZ, L.
"Implementação de uma Interface Homem-Máquina para o Sistema de Manipulação de Mensagem (MHS) no Ambiente VAX/VMS"
Relatório Técnico NCE a ser publicado