MODELO GLOBAL

Versão 5.02

LIVRO III

CAPÍTULO C
PROTOCOLO TRANSFERÊNCIAS DE FICHEIROS

C.1 INTRODUÇÃO

C.2 ALGUNS CONCEITOS GENÉRICOS USADOS NESTE PROTOCOLO

C.3 ESTRUTURAÇÃO E PARALELISMO COM O MODELO OSI

C.4 *SESSION LEVEL* (Abertura e Fecho)

C.5 SESSÃO, REGIMES E OPERAÇÕES EFECTUADAS DENTRO DE UMA SESSÃO

C.6 PROCEDIMENTOS PARA ENVIO E RECEPÇÃO DE FICHEIROS E CONSULTAS

C.7 FORMATO DAS MENSAGENS PARA O NÍVEL *FILE TRANSFER*

C.8 LIGAÇÕES ASSÍNCRONAS *(Assyncronous Communications Interface)*

C.9 COMPACTAÇÃO DE DADOS

C.10 APLICAÇÃO *FILE TRANSFER* EM JAVA – MFT97

© Setembro 2005 SIBS, S.A.

A informação seguinte é proprietária, não podendo ser duplicada, publicada ou fornecida total ou parcialmente a terceiros sem o prévio consentimento da Sociedade Interbancária de Serviços, S.A.

C.1 INTRODUÇÃO

Pretende-se nas páginas seguintes definir as regras que governam o diálogo entre dois sistemas residentes em computadores diferentes, para que estes possam transferir do primeiro para o segundo conjuntos de informação normalmente designados por ficheiros (sequências de registos), usando unicamente como meio de comunicação um circuito lógico, suportado por linhas/redes de comunicação de dados.

A leitura e escrita em cada um dos computadores, bem como a operacionalidade interna, não são objecto deste documento, pois prendem-se naturalmente com as características de cada máquina. Recomenda-se no entanto o maior grau de automatismo possível, evitando nas operações de rotina normal a sempre falível intervenção humana.

Este sistema tem por objectivo satisfazer as necessidades de transferência constante de grandes quantidades de informação de âmbito bancário, centralizadas pela SIBS com o seu Sistema Multibanco.

Seguindo a mesma orientação de anteriores protocolos e a mesma integração na Arquitectura de Interligação de Sistemas própria do ambiente Multibanco, recolhemos alguns conceitos e funções do modelo OSI, limitando-nos no entanto aos aspectos estritamente necessários a fim de tornar simples, performante e de rápida implementação o protocolo a desenhar.

O *standard* OSI-FTAM foi tomado em consideração, retirando-se daí alguns conceitos e funções, aplicáveis ao âmbito restrito que se pretende abranger.

Nota:
No presente documento são frequentemente usados nomes de comandos, parâmetros e outros em nomenclatura inglesa. Pensamos que ao usar esta nomenclatura, é mais fácil estabelecer paralelismo, semelhanças e diferenças com documentação similar tradicionalmente escrita em inglês.

Seguinte

C.2 ALGUNS CONCEITOS GENÉRICOS USADOS NESTE PROTOCOLO

Começamos por definir alguns conceitos, que de alguma forma são usados em *standards* internacionais e que por certo se aplicam ao ambiente Multibanco.

Ficheiro:
É uma colecção de informação estruturada e manipulada como um todo, à qual se aplicam atributos comuns. Um ficheiro é produzido por uma entidade e destina-se a outra entidade. É enviado de uma máquina para outra, podendo eventualmente ser enviado desta para uma terceira e assim sucessivamente.

Atributos de um ficheiro:
Identificação (nome, origem, destino, data e hora de criação), características e tamanho.

Entidade:
É uma aplicação, departamento, empresa ou banco, que produz e emite ficheiros e/ou recebe ficheiros.
Uma entidade pode usar mais do que uma máquina para enviar ficheiros (um banco com vários PCs) e várias entidades podem usar a mesma máquina para enviar/receber ficheiros (vários departamentos de um banco usando o computador central).

Endereço de máquina:
'File Transfer Machine' é um sistema capaz de efectuar file transfer, usando circuitos X.25 ou ligações assíncronas. Pode ser única no hardware em que está instalada (PCs por exemplo) ou coexistir com outras (por exemplo máquina de testes e máquina de produção). Cada máquina tem pois um endereço único, 4 bytes, cujo formato e conteúdo deve ser acordado bilateralmente entre os sistemas que vão dialogar.

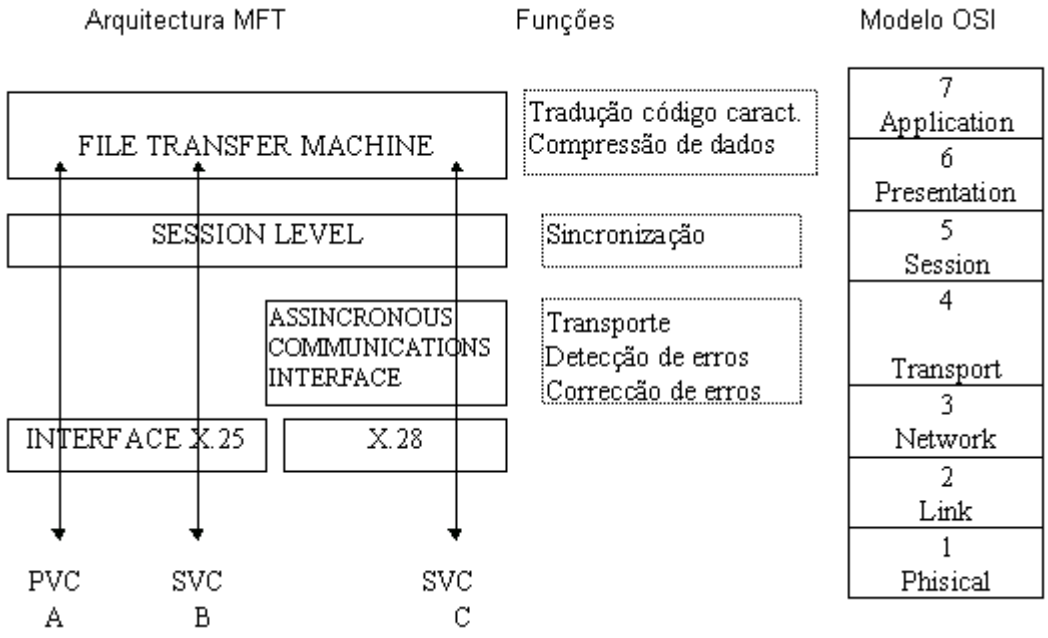
Primitivas:
São os comandos enviados pela entidade "F.T." ao nível inferior, para requerer qualquer serviço deste (abertura de sessão, fecho, abort, etc.).

Window:
Este conceito estabelece um valor limite para o envio de mensagens, ao fim do qual a máquina que está a recebê-las deve enviar uma resposta (*acknowledgement*) e a máquina que está a enviar deve parar o envio até receber aquela resposta, para que possa recomeçá-lo de novo. O conceito de *Window* pode ser usado independentemente a vários níveis da comunicação, e tem normalmente funções de '*pacing*'. Usá-lo-emos aqui a nível de sessão, nas mensagens de dados, que abrangem todo o dialogo F.T., exactamente com este objectivo.

Anterior/Seguinte

C.3 ESTRUTURAÇÃO E PARALELISMO COM O MODELO OSI

Como já se disse, o modelo OSI esteve presente como base orientadora deste sistema, pelo que a figura seguinte pretende estabelecer um certo paralelismo entre a estrutura a desenhar e os *layers* daquele modelo.



A figura pretende mostrar também os três cenários admitidos para implantação do *File Transfer*, no sistema central da SIBS.

A implementação a nível *File Transfer* pode no entanto suportar apenas um dos cenários (caso dos PCs).

Quanto a especificações dos níveis inferiores ao 4, é desnecessário qualquer comentário, dado serem níveis bem definidos em qualquer dos cenários apontados e normalmente implementados pelos construtores. Apontamos apenas os parâmetros passíveis de variação para os circuitos X.25 :

- Level 3 Window size = 2.
- Packet size = 128 bytes (pode ser acordado bilateralmente outro valor, quando usada linha ou rede privada).
- Facility Field no CALL Request NÃO usado.
- User Data Field nos pacotes de CALL, a acordar bilateralmente.
- Delivery Bit (D) = 0.
- Qualifier Bit (Q) = 0.

Cenário A :

Este cenário é implementado em PVCs X.25, quer usando uma linha alugada, quer a rede pública de comunicação de dados.

No cenário A, dado que as duas máquinas estão ligadas permanentemente entre si, os procedimentos do nível sessão limitam-se essencialmente a sincronizar os dois interlocutores, podendo eventualmente alterar parâmetros de sessão prefixados.

(Está praticamente abandonada a sua utilização, em benefício dos SVCs).

Cenário B :

Este cenário é implementado em SVCs, quer via rede pública de comunicação de dados, quer em linha ou rede privada.

Nesta situação, dado não existir uma ligação permanente entre as máquinas, torna-se necessário identificar os interlocutores para estabelecer parâmetros de sessão.

Especialmente neste cenário, são importantes as funções do nível sessão tais como estabelecimento e fecho, identificação e autenticação do interlocutor que estabeleceu a chamada X.25.

Parâmetros de sessão, tais como código de caracteres usado, devem ser previamente acordados entre ambos os interlocutores, pelo que são assignados consoante a identificação do interlocutor.

Este é o cenário preferencial do presente protocolo, por ser aquele em que se torna mais simples e performante a implementação. O circuito pode manter-se aberto permanentemente (quando usada rede privada) ou ser aberto no início de cada transferência e fechado no final.

Cenário C :

O cenário C, como se vê na figura, é usado para ligações assíncronas, normalmente em PCs que se ligam a um "PAD".

Neste cenário, tal como no B, tornam-se necessários os mesmos procedimentos do nível sessão. Mas além disso, em virtude da comunicação assíncrona, da baixa velocidade e dos possíveis erros de transmissão, torna-se necessário implementar algumas funções do nível transporte, que garantam a qualidade da informação e a sincronização entre as máquinas. Estes pormenores serão descritos em parágrafo próprio.

Para este cenário fica definido que é o PC que tem sempre a iniciativa de estabelecer a chamada X.25 e iniciar a sessão.

Anterior/Seguinte

C.4 *SESSION LEVEL* (Abertura e Fecho)

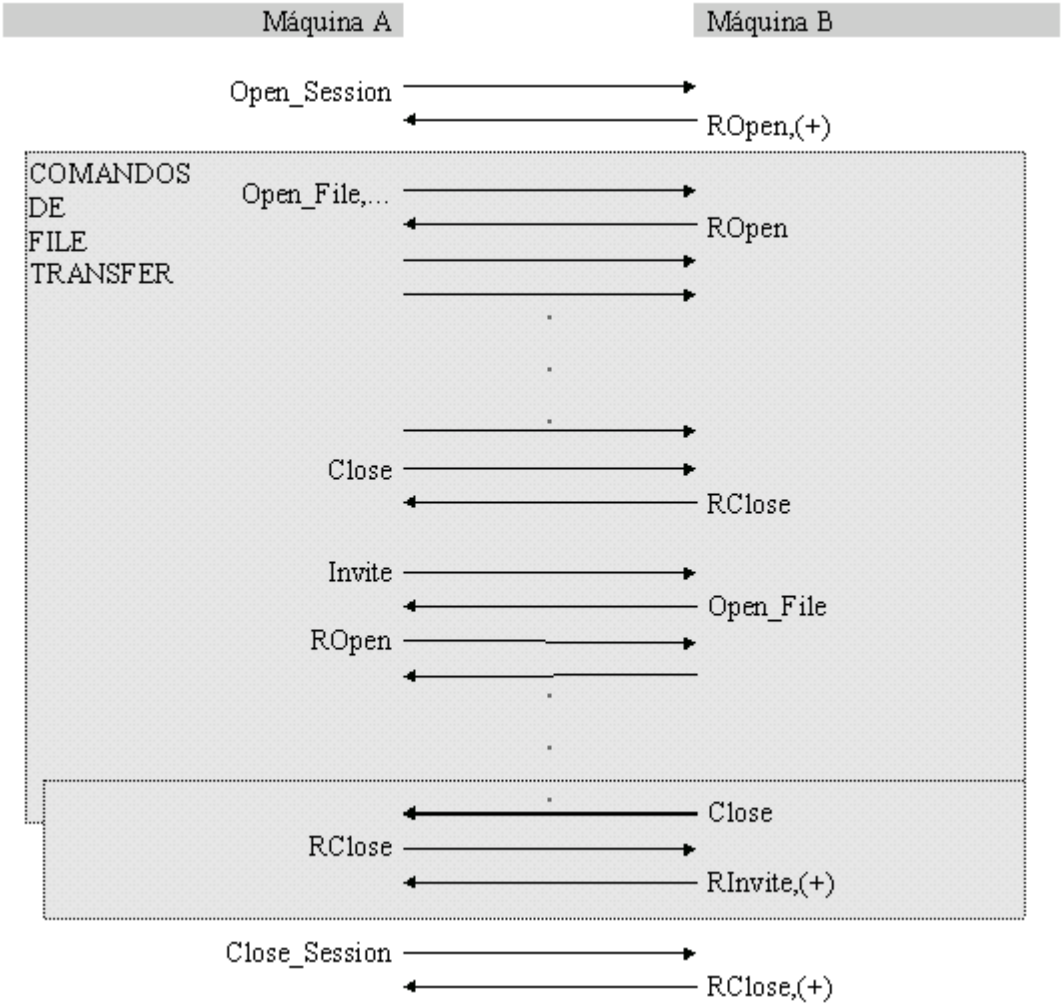
Este nível tem por função identificar e validar os interlocutores e estabelecer parâmetros de transmissão.

Ele estabelece comandos próprios para serem trocados no início e no fim de qualquer diálogo, e ainda um comando *Abort* que termina abruptamente uma sessão.

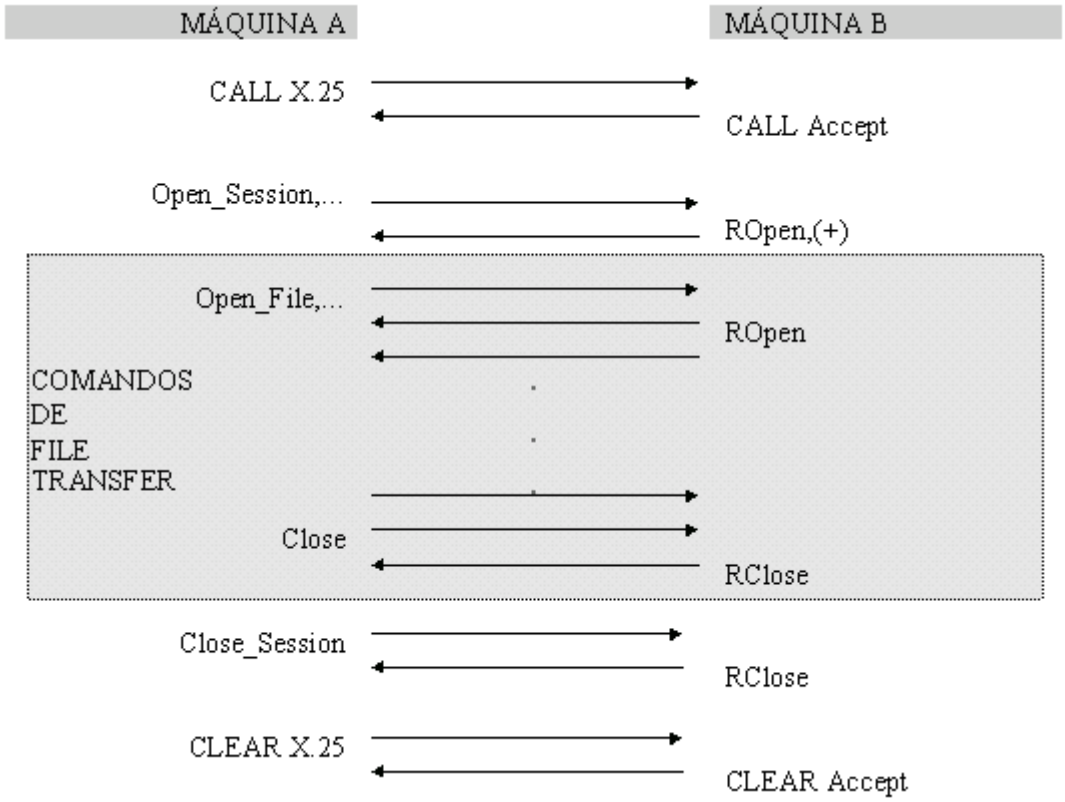
Os dados de utilizador são trocados através de comandos '*Data*', entre a abertura e o fecho ou *Abort* da sessão.

C.4.1 ESTABELECIMENTO E FECHO DE SESSÃO NAS LIGAÇÕES SÍNCRONAS

Cenário A :



Cenário B :



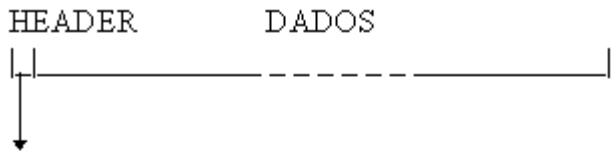
De notar que se houver um *Abort*, não há fecho de sessão, mas para restabelecer o diálogo deve ser feita nova abertura.

Parâmetros de sessão a acordar bilateralmente:

- Endereços de máquina a usar.
- Tamanho máximo das mensagens.
- Código de caracteres usado.
- Iniciativa de abertura da sessão.

C.4.2 *HEADER* NOS COMANDOS DE SESSÃO

O *header* aplica-se a todas as mensagens trocadas a este nível, ocupa a primeira posição da mensagem e tem tamanho fixo de 1 *byte*, com o formato a seguir definido :



Código do comando de Sessão.

- formato : alfanumérico
- valores : '1' - Open_Session
  '2' - ROpen
  '3' - Data_Message
  '4' - Data_Ack
  '5' - Close_Session
  '6' - RClose
  '7' - Abort
  '8' - Rabort

C.4.3 CODIFICAÇÃO DOS COMANDOS DE SESSÃO

Os parâmetros dos comandos de sessão são codificados em formato *display*, usando o código de caracteres que ambos os interlocutores acordarem previamente como parâmetro fixo (ASCII ou EBCDIC).

São os seguintes os comandos e respectivos parâmetros :

Comandos	Parâmetros	Bytes	Formato, valores e regras de preenchimento
OPEN	Source_ Address	8	Alfanumérico. Identificação do sistema chamador (endereço máquina). Normalmente só usados os primeiros 4 bytes.
	Destination_Address	8	Alfanumérico. Identificação do sistema chamado (endereço máquina). Normalmente são usados só os primeiros 4 bytes.
	Session_id	2	Alfanumérico. Identificação da Sessão ou da aplicação (poderá ser usado para distinguir diferentes Sessões FT).
	Window_ size	2	Numérico. Indica o número de Data_Messages sucessivamente enviadas, ao fim do qual tem que ser recebido um Data_Ack.
	Command_ seq_number	2	Numérico. Começa com o valor 1 e deve ser incrementado sempre que o comando é repetido.
	Security_ Level	1	0 - Nulo (não há segurança) 1 - Usa-se o Pin block (de um cartão) 2 - Usa-se Mac da mensagem 3 - Pin block + Mac 4 - Password.
	Autentication	var	Informação de segurança (a definir posteriormente em apêndice, de acordo com os níveis de segurança).
ROPEN	Resp_Code	1	0 - Ok. 1 - Erro de sequência ou de iniciativa do comando. 2 - Erro nos endereços indicados. 3 - Endereço destino indisponível de momento. 4 - Nível de segurança não aceite. 5 - Autenticação errada. 9 - Outros motivos.
	Window_ size_Resp.	2	Numérico. Deve ser igual ou menor que o proposto no Open.
	Command_ seq_number	2	Numérico, deve ter o mesmo valor que no correspondente comando Open.
	Autentication	var	Informação de segurança (a definir posteriormente em apêndice).
DATA_ MESSAGE	---	var	Dados de utilizador (nível F.T.).
DATA_ ACK	---	---	NÃO tem parâmetros ou dados.
CLOSE		---	NÃO tem parâmetros.
RCLOSE	Resp_Code	1	0 - Ok. 1 - Erro de sequência ou de iniciativa do comando. 9 - Outros motivos.

			(Com resposta negativa a sessão considera-se abortada).
ABORT	Diagnostic_ Code	1	0 - Sem informação adicional. 1 - Erro de Protocolo. 2 - Recebido comando ou resposta impróprio no presente regime ou estado da sessão. 3 - Erro interno. 4 - Erro no número de sequência do comando. 5 - Erro na validação de segurança. 6 - Ultrapassado o tamanho de ficheiro indicado no Open. 7 - Abort por comando de operador. 8 - Erro nos parâmetros do comando ou resposta recebido/a. 9 - Timeout na recepção de uma resposta.
RABORT	---	---	NÃO tem parâmetros ou dados.

C.4.4 FUNCIONAMENTO DA JANELA

O Window_size é o número de Data_Messages que qualquer máquina pode enviar seguidamente sem ter recebido nenhuma.

Este valor é negociado na abertura de sessão, sendo proposto um valor no comando *Open*, e respondido um valor = ou < no *ROpen*. Fica fixado para a sessão o valor do *ROpen*.

Assim uma máquina pode enviar de seguida um número máximo de Data_Messages = ao window_size, após o que deve parar o envio e esperar, até receber um Data_Ack ou outra Data_Message. A recepção de uma Data_Message antes de atingido o window_size, leva a contagem de Data_Messages enviadas a 0.

A máquina que recebe Data_Messages, ao receber um número seguido igual ao window_size, deve enviar um Data_Ack, a fim de poder continuar a receber novas Data_Messages.

O Data_Ack quando atingido o window_size é opcional, se em seguida houver uma Data_Message para enviar.

Significa isto que a janela é "aberta" de novo, tanto por um Data_Ack como por qualquer Data_Message.

Em resumo, qualquer máquina deve incrementar o contador de mensagens enviadas, sempre que envia uma msg-Data e fazer-lhe *reset* sempre que receber uma msg-Data ou Data_Ack. Por outro lado, deve incrementar o contador de recebidas, sempre que recebe uma msg-Data e fazer-lhe *reset* sempre que envia um Data ou Data_Ack.

Procedimentos nas situações mais típicas de recepção de mensagens impróprias ou outros erros :

	Recebido						
Enviado	Open	ROpen	Close	RClose	Data	Abort	RAbort
Open	ROpen(-)	Ok	Abort	Abort	Abort	Abort	Abort
Close	Abort	Abort	Abort	Ok	Abort	Abort	Abort
Data	Abort	Abort	RClose *	Abort	Ok	RAbort	Abort
Abort	---	---	---	---	---	---	---

* Assume-se neste caso que estão terminadas as operações de nível superior. Caso contrário, o próprio nível F.T. envia o *Abort*.

Timeout: O comando *Open* deve ser repetido, até um máximo de três vezes, após o que é aconselhável enviar um comando *Abort*. No *Close* não é aconselhável a repetição mas sim o *Abort*, logo após o primeiro *timeout*.

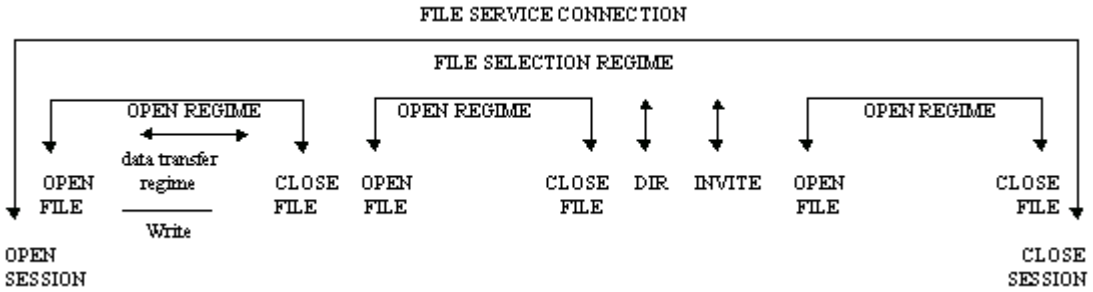
Anterior/Seguinte

C.5 SESSÃO, REGIMES E OPERAÇÕES EFECTUADAS DENTRO DE UMA SESSÃO

Apresenta-se aqui mais um conceito OSI, o de regime:

Regime é o período durante o qual uma parte dos comandos ou troca de informação é válida. Os regimes tornam-se mais restritos à medida que se limita o tipo de comandos/informação admitida. Pelo inverso, cada comando tem um regime em que é válido.

Também aqui simplificamos as definições do modelo OSI, tanto mais que não se pretende abranger as áreas de '*File Access*' e '*File Management*'.



A máquina que abre a sessão tem o seu controle, não podendo a outra enviar qualquer comando por sua iniciativa.

A máquina que abre a sessão pode enviar ficheiros e/ou pedir à outra que lhe envie ficheiros. O pedido de recepção é feito com um comando '*invite*' e este comando pode pedir de uma só vez que a segunda máquina envie todos os ficheiros que tem para enviar. É apenas neste caso que a segunda máquina fica com iniciativa de enviar ficheiros, só a perdendo quando terminar aquela operação.

Compete à máquina que abriu a sessão, fechá-la.

Numa só sessão podem ser efectuadas todas as operações que o protocolo prevê, podendo cada uma ser repetida várias vezes.

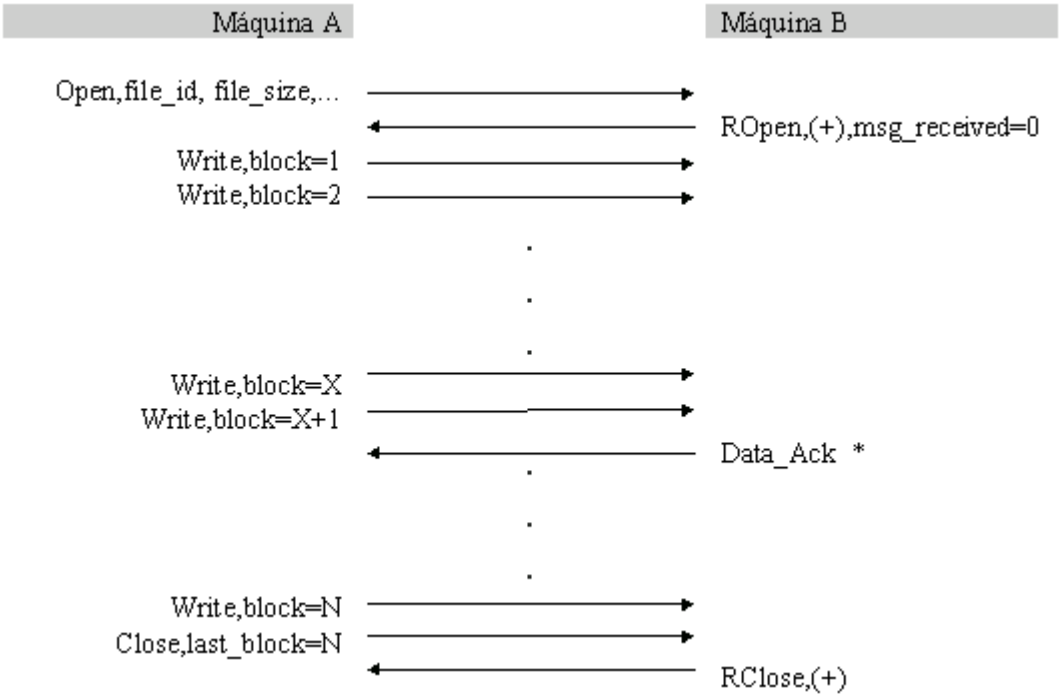
Anterior/Seguinte

C.6 PROCEDIMENTOS PARA ENVIO E RECEPÇÃO DE FICHEIROS E CONSULTAS

Passamos agora a definir as sequências de operações e comandos para o envio de um ou mais ficheiros, recepções e consultas ao directório, tentando abranger todas as situações típicas.

C.6.1 ENVIO POR INICIATIVA DO EMISSOR

Envio normal de um ficheiro, da máquina "A" para a máquina "B", num cenário em que a máquina "A" abriu a sessão:



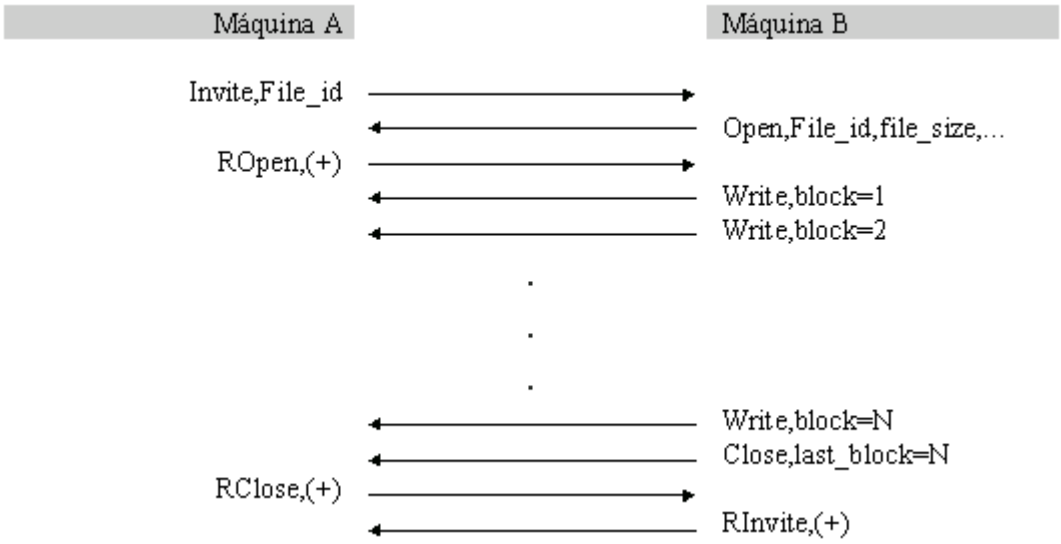
A resposta ao *Open* indica, neste caso, que o ficheiro é aberto no início, número de mensagens já recebidas = 0 . Este parâmetro vai distinguir as situações de início normal, das situações de restart de um ficheiro já enviado parcialmente.

No comando *Write*, entenda-se bloco = mensagem, independentemente de ela transportar apenas um ou mais registos.

* O comando Data_Ack pertence ao nível de sessão e é transparente para o nível FT. A sua apresentação aqui destina-se apenas a ilustrar o funcionamento deste mecanismo durante a sessão.

C.6.2 ENVIO POR INICIATIVA DO RECEPTOR (Pedido de recepção)

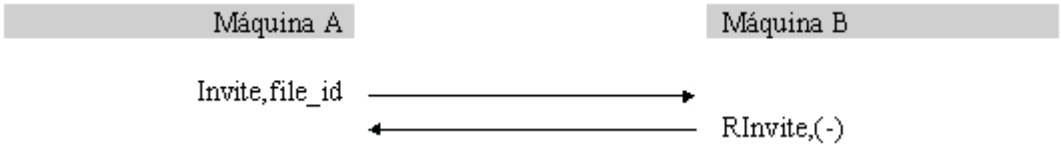
A máquina "A" pede à máquina "B" que lhe envie um ficheiro específico:



Dizemos neste caso que o "*Invite*" é específico, pelo que só dá origem à emissão do ficheiro especificado, caso ele exista.

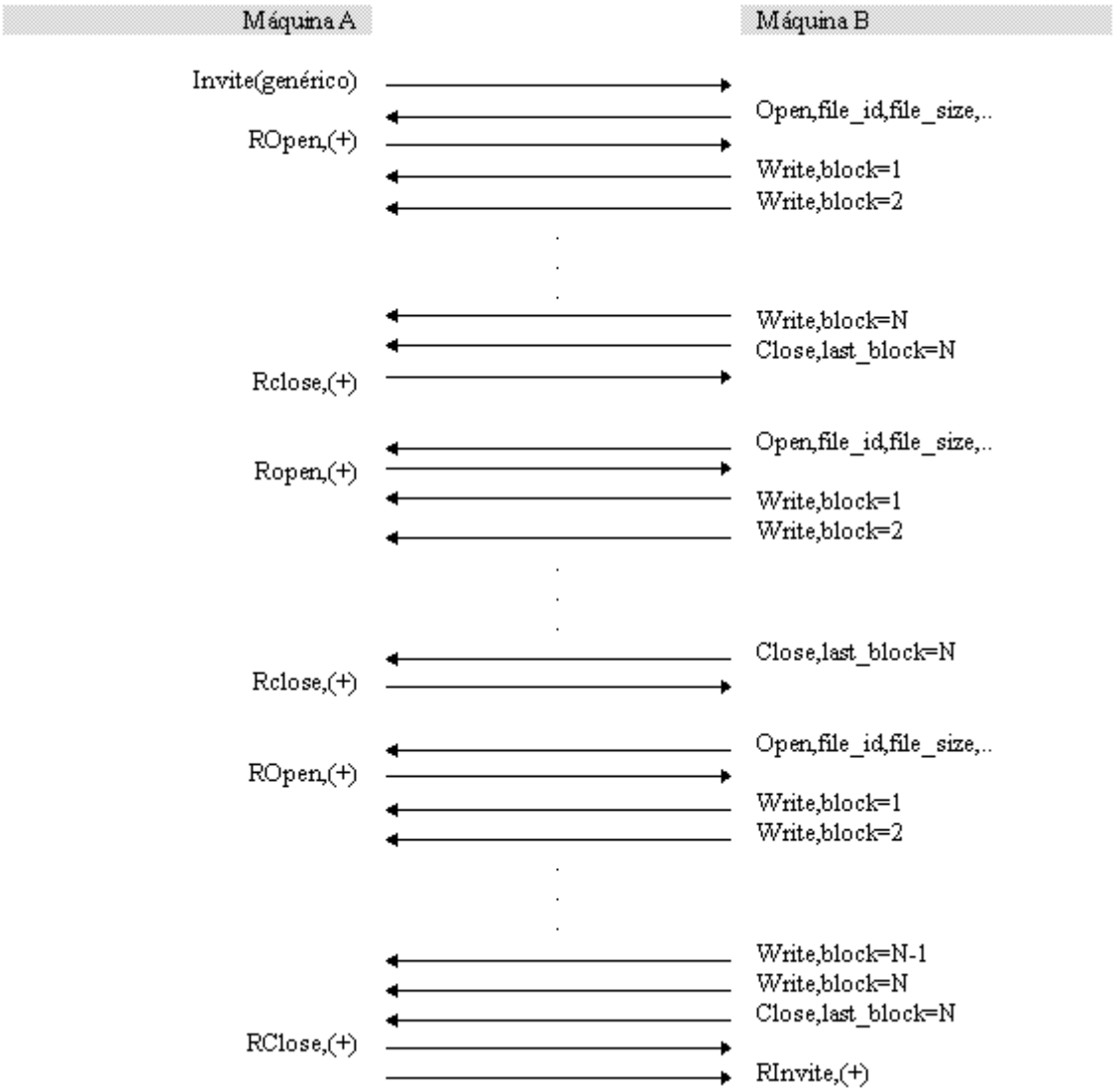
Adiante veremos a situação de "*Invite*" genérico, em que a máquina "A" pode pedir a recepção de todos os ficheiros que a máquina B tem para lhe enviar.

Apresenta-se também a situação em que a máquina "B" não tem o ficheiro pretendido:



A máquina "A" pode em seguida efectuar qualquer outra operação.

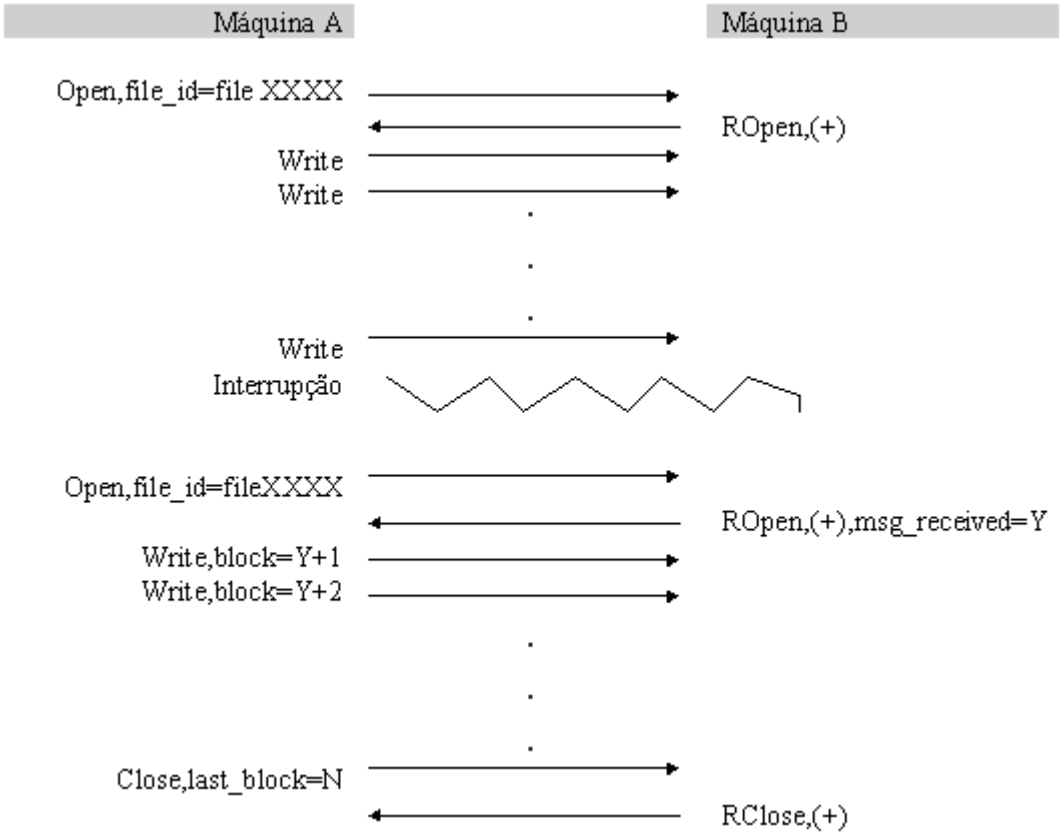
A máquina "A" pede à máquina "B" que lhe envie todos os ficheiros que tiver disponíveis para aquela:



Depois de enviados todos os ficheiros, a máquina "B" conclui a operação com a resposta ao "*Invite*", devolvendo assim o controlo à máquina "A".

C.6.3 REINÍCIO DE UMA TRANSFERÊNCIA

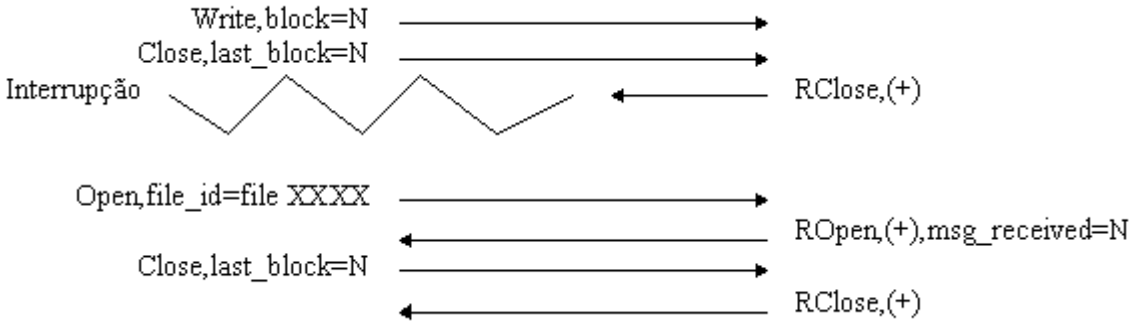
Interrupção e *Restart* numa transferência da máquina "A" para a "B":



Caso a transferência tivesse começado com um "*Invite*" da máquina "B", então o *restart* seria iniciado também com um "*Invite*", da mesma máquina, seguido do *Open* da máquina "A".

A interrupção pode acontecer por *timeout* na recepção de uma mensagem esperada, em qualquer dos interlocutores, por envio de um *Abort* de uma das máquinas ou por problema no circuito que provoque o *Abort* de ambos os lados.

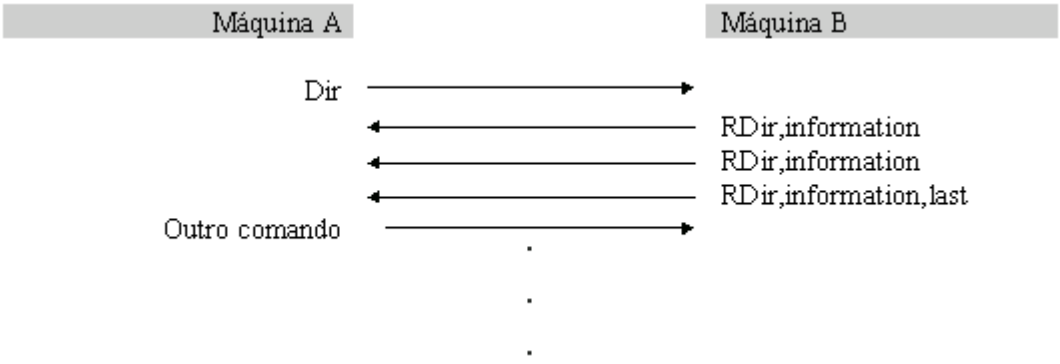
Apresenta-se ainda a situação em que a interrupção acontece na fase de *Close*.



C.6.4 CONSULTA AO DIRECTÓRIO

Esta operação é opcional, pelo que pode ser omitida a sua implementação, devendo neste caso ser devolvida uma resposta negativa, uma vez que o comando seja recebido.

O comando DIR destina-se a consultar o directório de ficheiros da máquina "B", a partir da máquina "A".



A informação contida em cada RDir, são várias ocorrências de File_Information (File_id + File_size + File_record_size) até ao limite de 20 por mensagem.

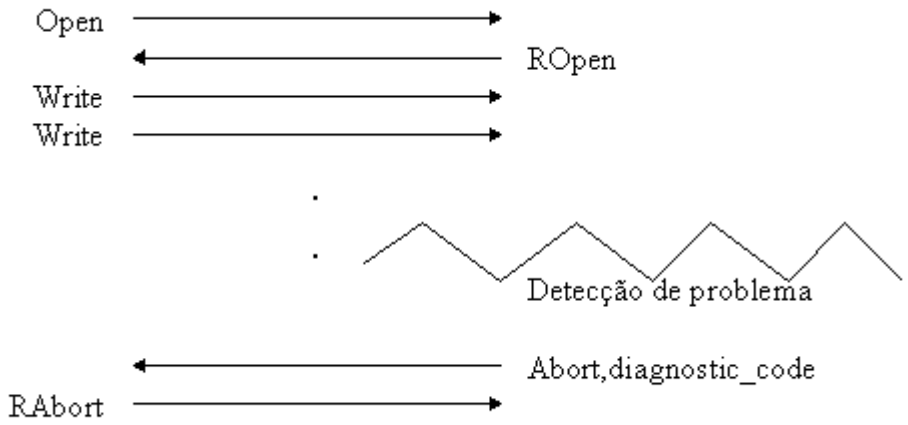
A resposta ao comando DIR pode ser constituída por várias mensagens, tantas quantas as necessárias para enviar toda a informação pedida. A última leva essa indicação explícita.

Este comando pode ser enviado pela máquina que tem o controlo da sessão, durante o '*file selection regime*'.

C.6.5 COMANDO ABORT

O *Abort* é um comando de sessão que pode ser enviado por qualquer das máquinas que detecte um problema, quer de transmissão, quer interno.

Provoca o *Abort* da operação em curso e da sessão, e no caso dos cenários B e C, o *Clear* do circuito X.25.



Depois de um *Abort*, e logo que resolvido o problema que o causou, a sessão tem que ser novamente aberta para iniciar ou reiniciar qualquer operação.

C.6.6 PROCEDIMENTOS NAS SITUAÇÕES MAIS TÍPICAS DE ERRO NAS RESPOSTAS ESPERADAS

Apresentam-se no quadro seguinte as situações possíveis de recepção dos diversos comandos, após o envio de um outro, apontando-se a acção a executar em cada uma das hipóteses.

	Recebido					
Enviado	Open	Write	Close	Invite	Dir	Abort
Open	Abort	Abort	Abort	Abort	Abort	RAbort
Write	Abort	Abort	Abort	Abort	Abort	RAbort
Close	Abort	Abort	Abort	Abort	Abort	RAbort
Invite	Abort	Abort	Abort	Abort	Abort	RAbort
Dir	Abort	Abort	Abort	Abort	Abort	RAbort
Abort	---	---	---	---	---	---

	Recebido					
Enviado	ROpen	RClose	RInvite	RDir	RAbort	Timeout
Open	Ok	Abort	Abort	Abort	---	RAbort
Write	Abort	Abort	Abort	Abort	---	RAbort
Close	Abort	Ok	Abort	Abort	---	RAbort
Invite	Abort	Abort	Ok	Abort	---	RAbort
Dir	Abort	Abort	Abort	Ok	---	RAbort
Abort	---	---	---	---	---	---

A acção traduz-se no envio do comando ou resposta indicado, ou simples paragem do processo quando não há comando a enviar.

Ok - significa que a resposta está correcta e que se pode prosseguir a operação.

Timeout aconselhados para os vários comandos :

Open - 60 segundos
Write - 40 "
Close - 60 "
Invite - 120 "
Dir - 120 "
Abort - 60 " (para efeito de *clear* do circuito nos cenários B e C)

Anterior/Seguinte

C.7 FORMATO DAS MENSAGENS PARA O NÍVEL *FILE-TRANSFER*

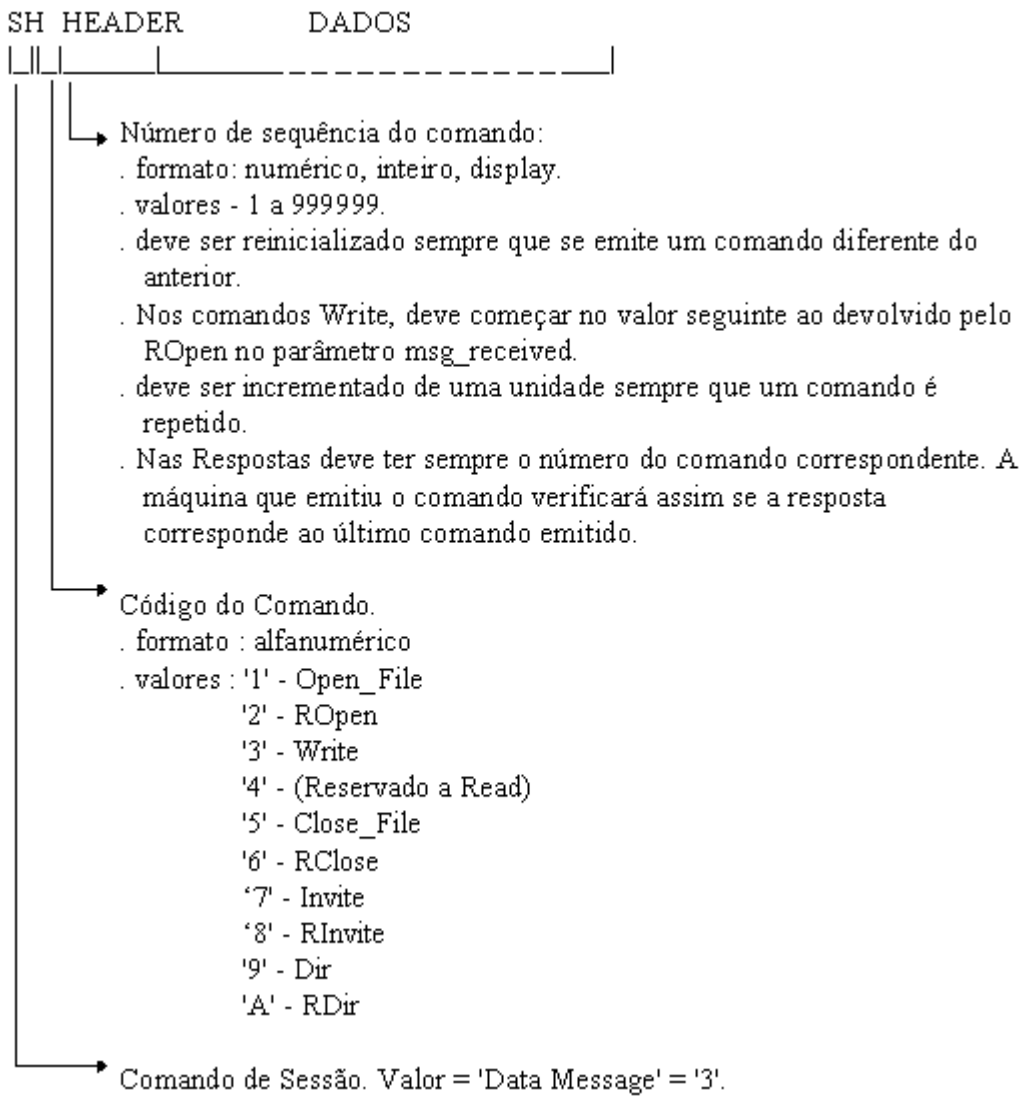
C.7.1 CODIFICAÇÃO DO *HEADER*

Todos os comandos F.T. são 'Data Messages' para o nível Sessão.

Assim, o *header* de sessão é seguido do *header* F.T., sendo este constituído por dois campos, com 7 *bytes* no total e tamanho fixo em todos os comandos.

- O primeiro é o código do comando e ocupa 1 *byte*.
- O segundo é o número de sequência do comando e ocupa 6 *bytes*.

Mensagens F.T. :



C.7.2 CODIFICAÇÃO DOS COMANDOS

Todos os parâmetros dos comandos são transmitidos em formato *display*, usando o código (ASCII/EBCDIC) que for acordado bilateralmente entre os interlocutores.

Note-se que a sessão identifica os interlocutores e assim automaticamente é conhecido o código usado no diálogo.

Comando	Parâmetros	Bytes	Formato, valores e regras de preenchimento
OPEN	File_id (File_name (File_origin File_destinat. File_code) File_date File_seq. num./time)	 8+ 8+ 8+ 6+ 4=34	Alfanumérico. Identificação completa do ficheiro. Nome, composto por 3 subcampos seguintes : 1. nome da entidade origem. 2. nome da entidade destino. 3. código ou nome do ficheiro. numérico, data de criação (AAMMDD). numérico, numeração sequencial ou hora de criação (NNNN ou HHMM).
	File_size	8	Numérico. Indica o número de registos do ficheiro. Nos ficheiros não estruturados em registos (ficheiros flat) indica o tamanho do ficheiro em bytes.
	File_record_ size	4	Numérico. Indica o tamanho do registo. Nos ficheiros não estruturados em registos terá o valor 1. Nos ficheiros de registo variável, terá o valor zero, situação que serve exactamente para caracterizar este tipo de ficheiros.
	Block_size	4	Numérico, tamanho do bloco/msg em transferência (número de bytes de dados em cada mensagem). Nos ficheiros de registo variável, este é o tamanho máximo do bloco.
	File_ compress	1	Indicador de compactação : 'Y' = Yes 'N' = No (No para ficheiros Binary).
	File_ caracter_ code	1	Código de caracteres usado no conteúdo do ficheiro : A = ASCII E = EBCDIC B = BINARY
	Block_ number	6	Numérico. Indica o número de blocos (mensagens) que são enviadas/recebidas, durante a transmissão.
ROPEN	Resp_code	1	0 - Ok. 1 - Erro de sequência ou de iniciativa do comando. 2 - Parâmetros errados ou valores não suportados. 3 - Ficheiro já recebido. 4 - Falta de espaço para o tamanho do ficheiro. 5 - A máquina que abriu a Sessão não está autorizada a enviar ficheiros desta Entidade. 9 - Outros motivos.
	Msg_ received	6	Numérico. = 0 se o ficheiro está a ser aberto pela primeira vez. = número de mensagens já recebidas quando se retoma uma recepção anteriormente interrompida.
WRITE	Data		Um registo ou um bloco. Valor limite 4084 bytes.
CLOSE_ FILE	Last_block	6	Numérico, deve conter o número de sequência do último Write enviado (ou seja o número de comandos Write efectuados/blocos enviados).
RCLOSE	Resp_Code	1	0 - Ok. 1 - Erro de sequência ou de iniciativa do comando.

			2 - Last_block não coincide com o valor do último Write recebido. 3 - Erro interno no Close. 9 - Outros motivos.
INVITE	File_id File_name File_date File_seq._ number	 24 6 4	(formato indicado no Open) O File_date e o File_seq_number devem ser preenchidos a zeros quando se deseja receber o ficheiro mais antigo ainda não transmitido, ou sempre que o Invite é genérico. (O file_code não é obrigatório e o file_origin também não. Podem ser preenchidos a espaços) (Se o File_code estiver a espaços, o Invite diz-se Genérico. Neste caso o Receptor pede ao Emissor que lhe envie todos os ficheiros que tiver para ele. Se o file_origin também estiver a espaços, o Invite refere-se a todos os ficheiros que o destino especificado tem para receber, independentemente da origem) (O Invite é especifico sempre que o File_code esteja preenchido. Neste caso apenas um ficheiro é enviado)
RINVITE	Resp_Code	1	0 - Resposta positiva. (a resposta positiva é enviada apenas depois de feito o envio do ficheiro ou ficheiros pedidos. Ou seja, a resposta positiva indica que está completa a operação solicitada). 1 - Erro de sequência ou de iniciativa do comando. 2 - Parâmetros errados ou valores não suportados. 3 - NÃO existe o ficheiro pedido ou não existe nenhum ficheiro para enviar. 5 - A máquina que abriu a Sessão não está autorizada a receber ficheiros para esta Entidade. 9 - Outras razões.
DIR	Inquiry_type	1	Situação dos ficheiros a consultar: A - ALL. T - Transmitidos (só). N - NÃO transmitidos (só).
	File_ destination	8	Nome da entidade destino. (Como descrito no Open_file)
	Initial_Key	6	File_date início de consulta.
RDIR	Resp.Code	1	0 - Ok. 1 - Erro de sequência ou de iniciativa do comando. 2 - Comando não suportado ou formato dos parâmetros errado. 3 - Directório vazio. 4 - NÃO há mais ficheiros para além da data indicada. 9 - outros motivos.
	End_ indicator	1	0 - NÃO é a última mensagem da resposta. 1 - É a última mensagem da resposta.
	Number of File_inf. in message	2	Numérico. Número de referências incluídas na mensagem. Valores de 1 a 20.
	File_Info. File_id + File_size + Record_size	46= (34+ 8+ 4)	 descrito anteriormente (no comando Open) " " " "

Comando 'Open':

File_id										
File_name										
ORIGIN	DESTIN.	CODE	DATE	SEQ.	FILESIZE	Rsize	Bsize			B.num
########	########	########	######	####	########	####	####	Y N	A E B	####

Comando 'ROpen' :

Resp_code	Msg_rec
0 1	######

Comando 'Dir' :

Type	File_dest	Initial_Key
A T N	########	###### 000000

Comando 'RDir':

			File_information		
Resp_ Code	End_ Indic	Num_ Files	File_identification	File_size	Rec_ size
0 ... 9	0 1	01 ... 20	XXXXXXXXXXXXXXXXXXXXX XXXNNNNNNNNNN	NNNNNNNN	NNNN

Comando 'Write':

Ficheiros com registo de tamanho fixo :

Data				
Record 1	Record 2	Record 3	---	Record N

Ficheiros de registo variável :

Nos ficheiros de registo variável, cada registo é precedido de 4 *bytes* que indicam o tamanho deste. Estes 4 *bytes* são representados no mesmo código de caracteres que os dados do ficheiro (ASCII ou EBCDIC). Se o ficheiro for 'Binary', então eles estão no mesmo código que os parâmetros do Open_File.

Data							
LLLL	Record 1	LLLL	Record 2	LLLL	---	LLLL	Record N

Anterior/Seguinte

C.8 LIGAÇÕES ASSÍNCRONAS (*Assyncronous Communications Interface*)

As comunicações assíncronas revestem-se de algumas particularidades que implicam o uso adicional de um protocolo de envelopagem e validação das mensagens, que evite a introdução de eventuais erros de transmissão nos dados transmitidos.

Vamos assim descrever os procedimentos adicionais que se enquadram no **cenário C**, atrás referido.

Em ambiente assíncrono pressupõe-se sempre a ligação entre um computador pessoal e um *Host*, usando normalmente a rede pública de comunicação de dados, em que apenas o primeiro tem iniciativa de estabelecer a chamada X.25 e abrir a sessão.

O acesso do PC à rede faz-se através de X.28 (PAD), podendo ser usada linha dedicada para este ou a rede telefónica.

O nível ACI tem funções complementares do nível de rede (sinalização do estabelecimento e fecho de chamada, e detecção e correcção de erros de transmissão). Para tal usa os seguintes mecanismos:

No envio de mensagens:

- Envelopagem das mensagens com dois caracteres no início e outros 2 no fim ; DLE STX e DLE ETX.
- No caso de se encontrarem no conteúdo da mensagem caracteres DLE, cada um deles é duplicado, por forma a evitar a ocorrência de uma configuração de fim de mensagem (DLE ETX) no interior da mesma.
- Geração um caracter LRC, usando para o seu cálculo toda a mensagem mais o DLE ETX, sendo colocado a seguir a este.

Na recepção de mensagens:

- Validação da mensagem recebida, através de idêntico cálculo do caracter LRC e comparação com o recebido. A procura do caracter LRC recebido deve ter em conta, a possível existência de pares de caracteres DLE, que não contam para a configuração de fim de mensagem.
- Resposta a todas as mensagens recebidas, com um caracter ACK, caso a validação do LRC tenha sucesso.
- No caso de detecção de erros, devolução de um caracter NACK, devendo neste caso o outro interlocutor repetir a mensagem.
- A repetição deve fazer-se até um máximo de três vezes, após o que deve ser abortada a sessão.
- Eliminação de um caracter DLE em cada 2 consecutivos encontrados na mensagem.
- Retirada das configurações de início e fim (DLE STX e DLE ETX), passando a mensagem limpa ao nível superior.

Formato das mensagens:

DLE	**STX**	MENSAGEM	**DLE**	**ETX**	**LRC**

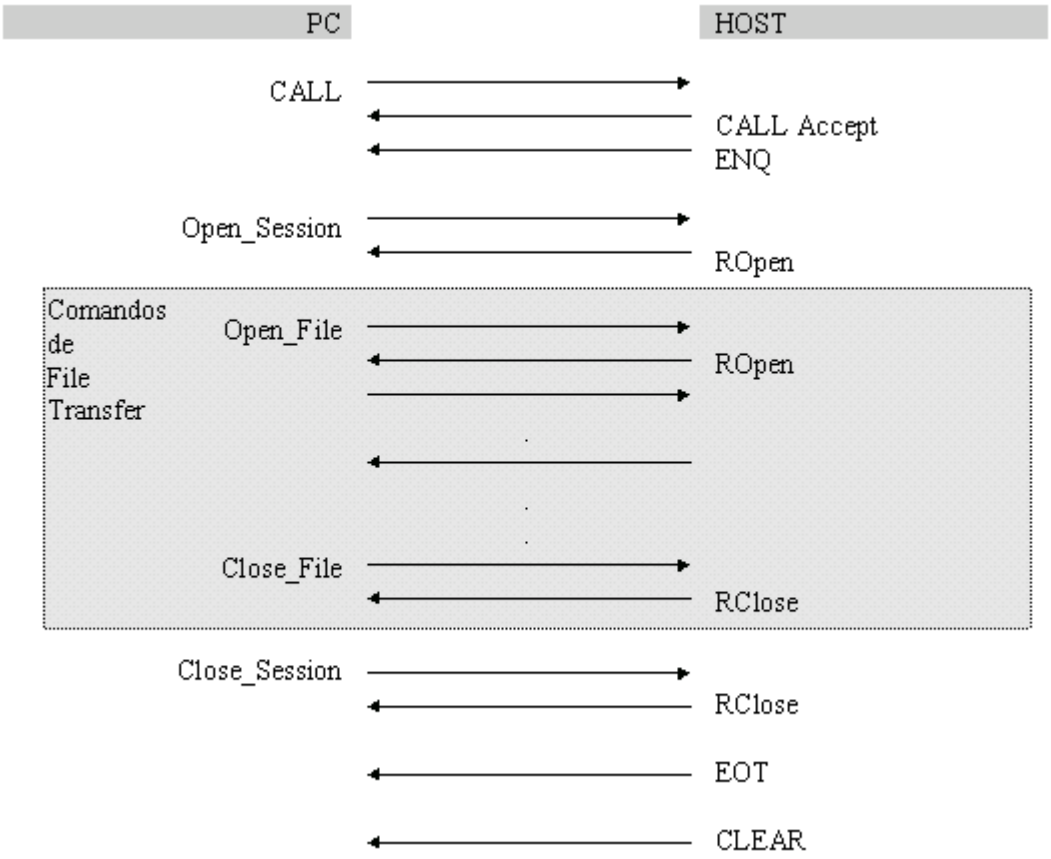
Estabelecimento da sessão:

Cabe ainda a este nível uma operação de envelopagem da sessão. Assim o PC, após feita a chamada para o *Host*, deve aguardar um caracter ENQ, que este envia para indicação de que o circuito está estabelecido.
(A ligação ao PAD deve funcionar em perfil 6)

Só depois de recebido este caracter, pode o PC enviar o comando de abertura de sessão.

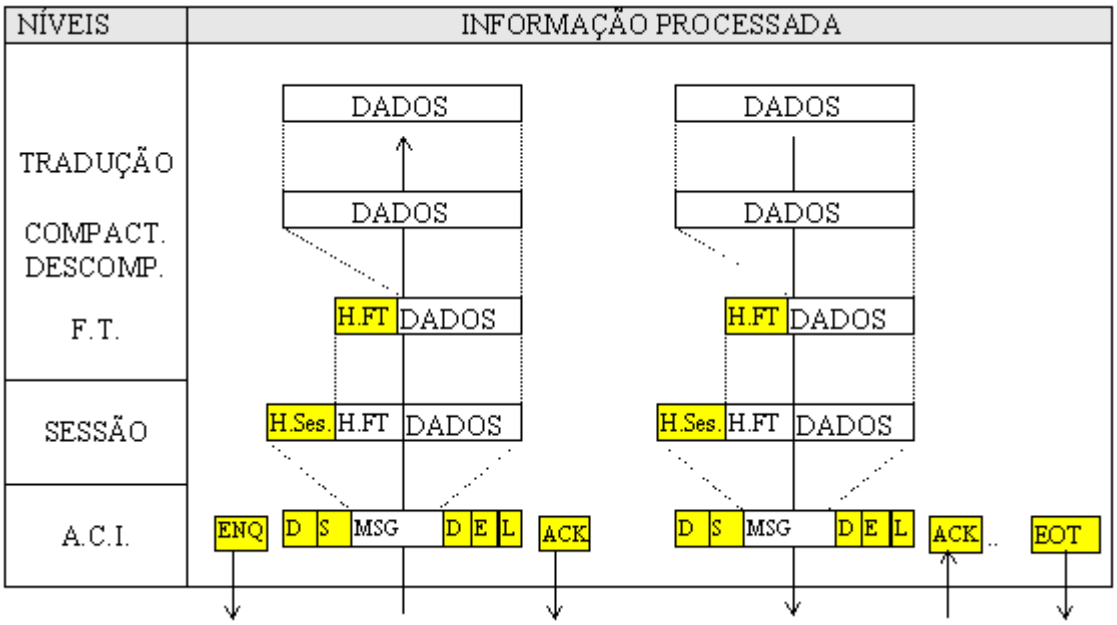
Compete também ao *Host*, logo após o fecho da sessão, enviar um caracter EOT, seguido do fecho do circuito. Este caracter tem também por função indicar ao PC que o circuito vai ser desligado.

Vejamos assim a sequência de intervenções deste nível:



Frisamos pois que há 3 comandos que são sempre da iniciativa do *Host*: ENQ , EOT e CLEAR ao circuito.

A figura seguinte mostra o posicionamento das camadas e respectiva intervenção no processamento da informação. Repare-se que os detalhes relativos a comunicações assíncronas estão concentrados num único nível (A.C.I.), pelo que, retirando esse nível, temos a restante parte que corresponde à versão usada em comunicações síncronas.



Anterior/Seguinte

C.9 COMPACTAÇÃO DE DADOS

A compactação destina-se a tornar mais rápida a transmissão, sempre que com um algoritmo apropriado se pode codificar a informação diminuindo o número de *bytes* a transmitir.

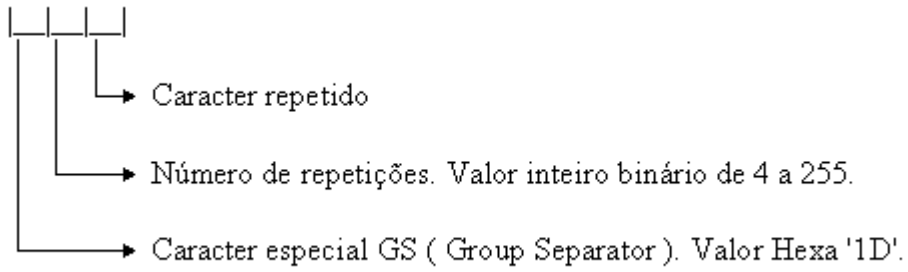
A fim de usar algoritmos simples, fáceis de implementar e performantes, aplica-se a compactação apenas em ficheiros *display* (ASCII ou EBCDIC).

A compactação faz-se ao nível do ficheiro, isto é, apenas aos dados e nunca aos headers nem aos parâmetros dos comandos. A unidade de informação a submeter de cada vez a este processo é um bloco (os dados que transporta uma mensagem).

Estabelecem-se dois algoritmos, um específico para compactar caracteres repetidos e o outro para compactar caracteres numéricos mais "A, B, C, D, E, F". Durante a compactação o primeiro será preferencialmente usado pois permite na maioria dos casos maior redução de caracteres.

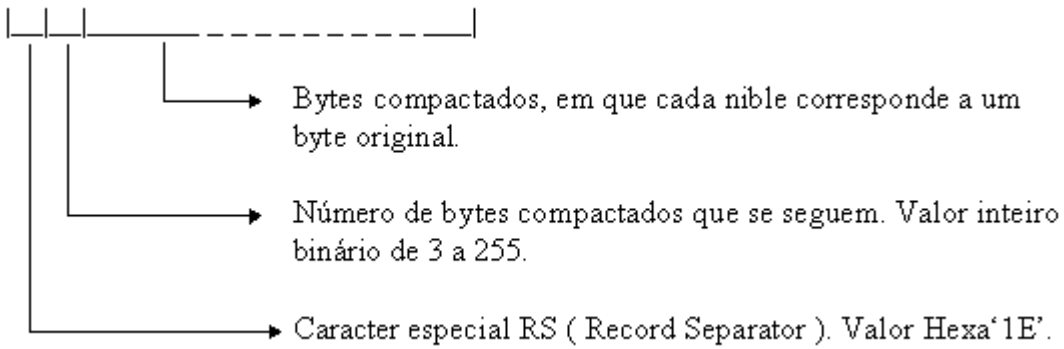
Caracteres repetidos:

Um *string* de caracteres repetidos é substituído por 3 caracteres com o seguinte formato:



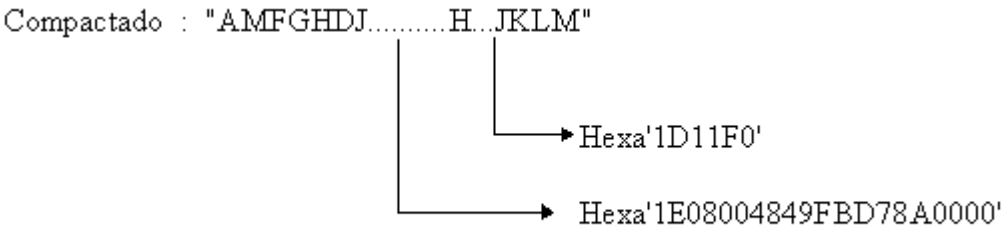
Caracteres numéricos + 'A B C D E F' :

Um *string* que contenha apenas aqueles caracteres é substituído por outro *string* com o seguinte formato:



Exemplo:

Não compact : "AMFGHDJ004849FBD78A0000H00000000000000000JKLM"



Anterior/Seguinte

C.10 APLICAÇÃO *FILE-TRANFER* EM JAVA - MFT97

A SIBS desenvolveu uma aplicação de *File Transfer* em Java, denominada MFT97, que tem a particularidade de poder ser instalada em múltiplas plataformas de *hardware*/sistema operativo, que possuam instalada uma máquina virtual de java.

Esta aplicação pode ter dois cenários de execução:

1. Para entidades que pretendam transferir um elevado volume de informação, sugere-se a instalação da aplicação no equipamento do cliente, sendo executada como uma aplicação *stand-alone*.

Actualmente é possível assegurar a boa execução da aplicação nas seguintes plataformas:

Windows/NT
IBM RISC 6000
HP UX
IBM AS/400

Comunicações

Em termos de comunicações, o acesso à SIBS é efectuado em TCP/IP através da utilização de um equipamento de comunicações (*router*) a instalar pelo fornecedor de comunicações.

2. Para entidades que pretendam transferir um baixo volume de informação, mesmo que numa base de utilização diária, caso das entidades de Pagamento de Serviços/Compras, aconselha-se a utilização da aplicação sobre a forma de um *applet*, disponível na Intranet da SIBS, e acedida mediante a utilização de um *browser*, Microsoft Internet Explorer 5.0 ou superior.

Neste cenário de utilização, a Empresa deve possuir um *pinpad* e é-lhe facturada uma taxa mensal de utilização da VPDN SIBS (que começa a ser facturada no mês em que os elementos para acesso à aplicação são disponibilizados, independentemente do acesso ser, ou não, utilizado).

Comunicações

Em termos de comunicações, o acesso à Intranet da SIBS pode ser efectuado numa das seguintes modalidades:

1. Ligação simples de um único terminal

   - Linha comutada analógica, com modem da entidade
   - Instalação da linha, aluguer mensal e tráfego negociados directamente entre a entidade e o operador de comunicações.

2. Ligação em rede (múltiplos terminais)

   - Instalação de um *mini-router* pelo fornecedor de comunicações da Empresa, que permite ligar uma rede local tipo Ethernet, com protocolo TCP/IP à rede da SIBS, quer por acesso RDIS, quer por circuito dedicado *Frame-Relay*. O processo pode ser evolutivo, isto é, ser iniciado em RDIS até que o circuito *Frame-Relay* fique instalado.

Anterior