

EXPERIÊNCIA DA IMPLEMENTAÇÃO DO PROTOCOLO DE TRANSPORTE ISO,
CLASSE 2 , EM MICROPROCESSADORES*

AUTORES: STEFANIA STIUBIENER

ESCOLA POLITÉCNICA DA UNIVERSIDADE DE
SÃO PAULO

WAGNER LUIZ ZUCCHI

ESCOLA POLITÉCNICA DA UNIVERSIDADE DE
SÃO PAULO

BRISA - SOCIEDADE BRASILEIRA PARA IN
TERCONEXÃO DE SISTEMAS ABERTOS

* Este trabalho foi desenvolvido dentro do Programa conjunto de atividades firmado entre o Centro Científico Rio da IBM Brasil e o Laboratório de Sistemas Digitais do Departamento de Engenharia de Eletricidade da Escola Politécnica da Universidade de São Paulo.

INTRODUÇÃO

O atual desenvolvimento de aplicações distribuídas, funcionando ou não em tempo real, impõe cada vez mais a utilização de redes de computadores como solução de implementação destes sistemas.

O projeto de uma rede implica na escolha de soluções de "hardware" especialmente no que diz respeito a implementação das funções de comunicação de dados e em soluções de "software" para a organização da troca de informação, especialmente no que diz respeito às funções orientadas ao processamento de dados. O cumprimento das funções de comunicação de dados baseia-se na implementação de protocolos de redes.

A literatura especializada em assuntos de redes, artigos publicados em periódicos ou apresentados em congressos abordam aspectos de modelos de arquiteturas de protocolos, padronização, métodos formais de especificação e verificação, testes de protocolos etc, porém, em poucas ocasiões foi debatido o aspecto de implementação de protocolos. Nesta perspectiva, o pre

sente artigo se propõe a divulgar a experiência adquirida pelo grupo de redes de computadores do LSD-EPUSP nos últimos dez anos, nesta área.

O propósito deste artigo é apresentar a anatomia da implementação de um protocolo estabelecendo as fases desta atividade e adotando como objeto de trabalho a implementação do protocolo de transporte classe 2 elaborado pelo ISO [1,2], num microcomputador compatível com a linha PC-XT.

O artigo se baseia nos conceitos de "especificação de implementação" anteriormente divulgados [3]. Inicialmente será feita uma breve exposição de motivos da escolha do microcomputador como ambiente de implementação experimental do protocolo. Em seguida, será apresentada a especificação de implementação, o projeto de "software", os testes e finalmente as conclusões.

A convicção dos autores deste artigo é de que a implementação de protocolos constitui um problema extremamente importante e para o qual as técnicas tradicionais de implementação de algoritmos são insuficientes. Entre a leitura do protocolo a ser implementado e o seu efetivo funcionamento devem ser inseridas uma série de etapas que garantem a qualidade do produto final. Neste artigo apresenta-se uma metodologia para preenchimento destas lacunas.

USO DE MICROCOMPUTADORES PARA A IMPLEMENTAÇÃO DE PROTOCOLOS.

A estrutura dos sistemas de teleprocessamento e redes de computadores representa a interligação de equipamentos de processamento de dados de várias portes: computadores de grande porte, minicomputadores e microcomputadores.

O microcomputador torna-se cada vez mais a figura presente no sistema de rede, desempenhando o papel de terminal inteligente que permite a descentralização das funções de processamento, o papel de unidade controladora ou de estação de trabalho. Dentro deste contexto a implementação de protocolos em microcomputadores torna-se uma necessidade contínua e conseqüentemente aumenta o interesse do estudo da problemática das limitações desses equipamentos de pequeno porte para este fim.

De fato, os microcomputadores não oferecem os recursos necessários à implementação de protocolos dentro do contexto OSI, tais como, a possibilidade de comunicação entre processos, gerenciamento de memória, temporizadores de tempo real, etc. A convivência com tais limitações exige que no desenvolvimento de protocolos sejam elaboradas soluções que venham a suprimi-lâmas que nem sempre são as mais convenientes do ponto de vista de transportabilidade de programa, da escolha do sistema operacional e da linguagem de programação.

Além destas considerações, o custo do desenvolvimento em microcomputadores e a própria abertura que estes sistemas oferecem ao desenvolvimento de "software" básico facilita a realização do trabalho sobre estes equipamentos.

2. ESPECIFICAÇÃO DE IMPLEMENTAÇÃO

2.1 Escolha do Perfil

A especificação de implementação do protocolo, de acordo com a metodologia descrita em [3] se inicia com a escolha do perfil a ser utilizado. Os critérios de escolha que devem ser considerados são as aplicações potencialmente mais importantes para o sistema e o tipo de equipamento físico onde o protocolo será implementado.

No caso do protocolo de transporte esta escolha significa a definição da classe a ser utilizada, das opções dentro da classe, os campos opcionais a serem utilizados e o tamanho dos campos de dados.

A classe 2 foi escolhida pela sua relativa simplicidade e por permitir a multiplexação de conexões de transporte sobre uma conexão de rede. Esta multiplexação é importante para permitir o uso de múltiplas aplicações independentes quando se possui uma única conexão de rede. (É o caso do acesso à sub-rede através do protocolo X.28).

As opções foram escolhidas considerando a limitação de capacidade das máquinas-objeto e as possíveis aplicações a serem desenvolvidas nos níveis superiores.

Optou-se por manter na entidade a ser implementada todos os recursos oferecidos pelo protocolo que não impossibilita sem seu desenvolvimento nas máquinas de pequeno porte. Dessa forma consegue-se atender a uma ampla gama de aplicações potenciais.

As opções escolhidas para a operação da classe 2 estão mostradas na tabela 1.

	1	2
concatenação e separação		x
dados expeditos	o	o
controle de fluxo explícito	x	x
formato normal	x	x
formato estendido		
controle de fluxo implícito		
não uso de dados expeditos	o	x

Tabela 1. - Opções Escolhidas para Operação

LEGENDA:

Coluna 1 - Uma marca "o" nesse coluna indica que o usuário do nível de transporte precisa solicitar a opção para ela ser utilizada numa conexão de transporte.

Uma marca "x" nessa coluna indica que a opção é utilizada em todas as conexões de transporte originadas por essa entidade.

Um espaço nessa coluna indica que a opção nunca é utilizada nas conexões de transporte originadas por essa entidade.

Coluna 2 - Uma marca "o" nessa coluna indica que conexões de transporte remotamente solicitadas

são aceitas quando o usuário local da camada de transporte aceita operar com aquela opção.

Uma marca "x" nessa coluna indica que conexões de transporte remotamente solicitadas podem sempre ser aceitas, sem necessidade de confirmação de utilização pelo usuário local.

Um espaço nessa coluna indica que conexões de transporte remotamente solicitadas e utilizando a opção correspondente são sempre recusadas.

As escolhas realizadas, justificam-se da seguinte maneira:

- a) A concatenação e a respectiva separação de TPDU's dentro de um único NSDU foi considerada complexa para microcomputadores uma vez que exige a acumulação de TPDU's originados de diversas fontes com a posterior montagem de NSDU. Além disto, a otimização que poderia ser obtida no uso do protocolo de acesso à rede é irrelevante, pois se prevê um pequeno número de conexões.

Na recepção este recurso é sempre aceito por estar incluído na classe 2.

- b) O serviço de dados expedidos deve ser implementado e sua utilização deixada a critério dos usuários por ser utilizado em alguns protocolos de aplicação, como o FTAM, por exemplo.
- c) A opção de controle de fluxo explícito é importante para permitir a comunicação de micros com máquinas de maior porte, e portanto de maior velocidade, e também para permitir a interrupção temporária do fluxo de dados a fim de que o micro possa realizar

outras atividades pendentes.

- d) A numeração das TPDU's em formato normal justifi
ca-se pela previsão de utilização deste perfil a
través de enlaces terrestres com atrasos relativa
mente pequenos. De fato num acesso a 1200 bps as
síncrono e transmitindo TPDU's de 128 octetos o
tempo necessário para a rotação da janela é:

$$128 \times \frac{128 \times 11}{1200} \cong 150 \text{ segundos}$$

(sem levar em conta a sobrecarga produzida pelas men
sagens de controle e pelos níveis inferiores).

No que diz respeito aos campos opcionais das TPNU's a entidade de transporte pode receber TPDU's com todos os parâme
tros opcionais previstos no protocolo, sem interpretar esses
campos como erros de protocolo. Todavia, na implementação a ser realizada, apenas um sub-conjunto desses parâmetros é considera
do.

Nas TPDU's transmitidas também apenas um sub-conjunto dos parâmetros opcionais é utilizado.

A seguir, são apresentados os sub-conjuntos de parâme
tros considerados na recepção e transmitidos com os TPDU's en
viados (tabela 2.).

a) CR TPDU e CC TPDU		
	1	2
Identificador do TSAP origem	x	x
Identificador do TSAP destino	x	x
Tamanho do TPDU		o(2)
Número de versão		
Parâmetros de proteção		
Checksum		
Seleção de opção adicional	o(1)	o
Classes alternativas		
Tempo de reconhecimento		
Throughput		
Taxa de erro residual		
Prioridade		
Atraso de trânsito		
Tempo de reatribuição		
b) DR TPDU e DC TPDU		
Informação adicional		
Checksum		
c) DT TPDU		
Checksum		
d) ED TPDU		
Checksum		
e) AK TPDU		
Checksum		
Número de subsequência		
Confirmação de controle de fluxo		
f) EA TPDU		
Checksum		
g) ER TPDU		
TPDU inválida	x	o
Checksum		

Tabela 2. - Subconjunto de Parâmetros Considerados na Recepção e Transmitidos com as TPDU's

LEGENDA:

Coluna 1 - Um "x" nessa coluna indica que o parâmetro é sempre transmitido com a TPDU.

Um "o" nessa coluna indica que o parâmetro é utilizado apenas quando o usuário da entidade de transporte solicitou um serviço opcional correspondente.

Um espaço nessa coluna indica que o parâmetro nunca é transmitido.

Coluna 2 - Um "x" nessa coluna indica que o parâmetro sempre é considerado quando a TPDU é recebida e sua ausência implica no encerramento da conexão de transporte.

Um "o" nessa coluna indica que a presença do parâmetro é opcional, porém, quando presente seu valor será considerado para efeito dos procedimentos adotados na camada de transporte.

Um espaço nessa coluna indica que o parâmetro não é considerado nas TPDU's recebidas.

Notas:

1. Esse parâmetro será utilizado nas CR TPDU transmitidas apenas se a não utilização de dados expeditos for solicitada pelo usuário da entidade de transporte.
2. Não será aceito o estabelecimento de conexões de transporte com entidades parceiras que solicitarem um tamanho de TPDU diferente de 128 octetos. Caso se pode verificar o formato das TPDU's foi simplificado tanto quanto possível, objetivando facilitar a sua montagem. A transmissão dos identifi

cadadores de TSAP origem e destino, utilizados como seletores de transporte visa facilitar a interconectividade e o reconhecimento mútuo das aplicações.

Finalmente, no que diz respeito ao uso de campo de dados nas TPDU's especifica-se que as TPDU's dos tipos CR, CC, DR e DT podem conter dados até os limites indicado a seguir. A superação dos limites indicados deve ser considerado um erro de protocolo.

Nenhum procedimento executado pela entidade de transporte deve depender do conteúdo desse campo de dados.

Os limites máximos das áreas de dados são:

CR TPDU	:	até 32 octetos
CC TPDU	:	até 32 octetos
DR TPDU	:	até 64 octetos
DT TPDU	:	até 123 octetos
ED TPDU	:	até 16 octetos

Os tamanhos das áreas de dados são definidas dentro dos limites da recomendação 8073 e de forma a originar um pacote de nível de rede de tamanho máximo igual a 128 octetos.

Com as hipóteses realizadas e opções assumidas a presente especificação de implementação pretende-se conforme ao protocolo ISO 8073 na operação classe 2, transmitindo e recebendo CR TPDU's e com tamanho máximo de TPDU de 128 octetos.

2.2 Descrição dos Procedimentos da Entidade de Transporte

A segunda etapa da elaboração de uma especificação de implementação consiste em descrever os procedimentos da entidade que implementará o protocolo, particularizando-os para as opções que foram selecionadas, completando a descrição da norma quando necessário e servindo com base para o projeto de "software".

No caso do protocolo de transporte esta etapa consis

te a nosso ver, no detalhamento das tabelas de transição de es
tados fornecidos pela recomendação ISO 8073 e em abstrair daque
las tabelas os procedimentos relativos à operação classe 2. Ao
final do processo deve-se obter uma máquina de estados finitos
estendida, representada por uma tabela de transição de estados,
suficientemente detalhada para permitir a implementação indepen
dentemente de consulta à norma e omitindo detalhes irrelevantes
para o perfil escolhido.

A fim de reduzir as dimensões desta máquina de esta
dos e facilitar sua compreensão, ela foi dividida, de acordo com
um critério funcional, em três outras máquinas, hierarquicamen
te comunicantes, que serão chamadas de componentes da entidade
de transporte (esta terminologia é utilizada em algumas padroni
zações da ISO que utilizam esta técnica de divisão de máquinas
de estados).

Cada componente é responsável pelo tratamento de um
subconjunto dos eventos reconhecíveis pela entidade de transpor
te. Uma hierarquia é estabelecida entre as componentes, no sen
tido de que, a componente hierarquicamente superior passa os
eventos para a componente inferior, desde que esta componente
esteja habilitada por aquela.

A divisão em componentes deve ser vista como uma téc
nica de descrição de máquinas de estado e não como um regimento
para implementação. A mesma técnica de sub-divisão é utilizada
nas linguagens de descrição formal, particularmente em ESTELLE
[4] sem que se pretenda criar vínculos para o implementador .
É óbvio que em alguns casos particulares pode ser conveniente
utilizar na implementação a mesma sub-divisão, porém fica a cri
tério do programador decidir em qual elemento de implementação
as componentes serão mapeadas.

A divisão da entidade de transporte em componentes é
mostrada pela figura 1. A função de cada componente é:

- a) Componente de estação: responsável pelo tratamento
de eventos relativos à interface de nível inferior;
- b) Componente de conexão: responsável pelos eventos re
lativos ao estabelecimento, manutenção e encerramen

to das conexões de transporte;

- c) Componente de controle de fluxo: responsável pela troca de dados dentro de uma conexão de transporte, pela sincronização dos dados trocados e pelo gerenciamento de memória na entidade de transporte.



FIGURA 1. - COMPONENTES DA ENTIDADE DE TRANSPORTE E SUA HIERARQUIA

A descrição completa das componente transcende em muito o espaço deste trabalho. A título de ilustração é apresentada a seguir a descrição da componente estação que é relativamente mais simples. A descrição completa pode ser encontrada em [5].

2.2.1. Componente de Estação

A componente de estação é responsável pelo gerenciamento das conexões de rede e eventos unicamente associados a essa interface.

Embora possam haver diversas conexões de rede numa única máquina, a componente de estação é sempre única e seu comportamento é independente da conexão de rede a qual um evento é associado.

Entretanto, admite-se, para fins de especificação, que

os eventos associados à interface da camada de rede são vinculados a uma conexão de rede por um identificador terminal de conexão de rede que permite à estação de transporte reconhecer, quando necessário, qual (ou quais) a conexão de rede afetada pelo evento.

O diagrama de estados da componente de estação é apresentado na figura 2.

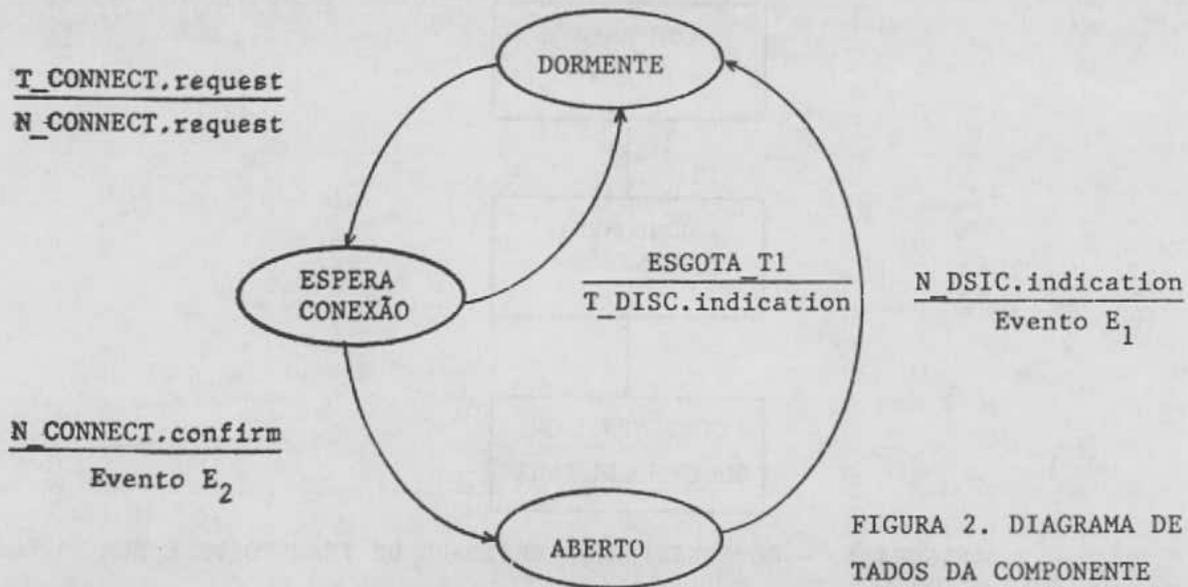


FIGURA 2. DIAGRAMA DE ESTADOS DA COMPONENTE DE ESTAÇÃO.

A tabela de transição para o diagrama de estados da componente de estação é apresentada a seguir (tabela 3.).

ESTADO ATUAL	EVENTO	AÇÃO	PRÓXIMO ESTADO
DORMENTE	T-CONNECT.request	. Envia N-CONNECT.request . Liga T ₁	ESPERA CONEXÃO
	N-DISCONNECT	. Gera evento E ₁ para a Componente Conexão	DORMENTE
	Outro Evento	. Envia para Componente de Conexão	DORMENTE
ESPERA CONEXÃO	T-CONNECT.request	. Descarta	ESPERA CONEXÃO
	T-DISCONNECT.request	. Envia N-DISCONNECT.request <causa: conexão abortada> . Desliga T ₁	DORMENTE

TABELA 3 (PARTE 1 DE 2) - TRANSCRIÇÃO DE ESTADOS DA COMPONENTE DE ESTAÇÃO

ESTADO ATUAL	EVENTO	AÇÃO	PRÓXIMO ESTADO
ESPERA CONEXÃO	N-DISCONNECT. Indication <falha de conexão>	. Envia T-DISCONNECT.Indica <u>tion</u> <causa:conexão de rede não disponível > . Desliga T_1	DORMENTE
	N-CONNECT.confirm	. Gera evento E_2 para a Comp <u>o</u> nente de Conexão . Desliga T_1	ABERTO
	N-CONNECT.confirm < colisão >	. Envia T-DISCONNECT.indica <u>tion</u> <causa: colisão > . Desliga T_1	DORMENTE
	Esgota T_1	. Envia T-DISCONNECT.Indica <u>tion</u> . Envia N-DISCONNECT.request	DORMENTE
	Outro Evento	. Envia para a componente Conexão	ESPERA CONEXÃO
ABERTO	N-DISCONNECT. Indication	. Gera evento E_1 para a com <u>o</u> nente Conexão . Envia N-DISCONNECT.request	DORMENTE
	Esgota T_2	. Gera evento E_1 para a com <u>o</u> nente Conexão . Envia N-DISCONNECT.request	DORMENTE
	Outro evento	. Envia para a componente Conexão	ABERTO
	T-CONNECT.request < de outra Conexão de transporte >	. Gera evento E_2 para a com <u>o</u> nente Conexão	ABERTO

Tabela 3(parte 2 de 2) - Transição de Estados da Componente de Estação

As temporizações e os eventos internos da tabela 3 são explicados a seguir:

a) Temporizações da Componente de Estação:

- T_1 : temporizador que limita o tempo em que componente de estação espera pela abertura de uma conexão de rede. Esse temporizador é iniciado quando a componente entra no estado "espera conexão" e desligado quando a componente deixa esse estado.
- T_2 : temporizador de manutenção de conexões de rede. Esse temporizador encerra as conexões de rede quando não há conexões de transporte a elas associados por um tempo maior do que a sua duração. O relógio é iniciado pela componente de conexão e é desligado por essa mesma componente se uma conexão de transporte vier a ser estabelecida sobre a conexão de rede.

b) Eventos internos da componente de estação:

- E_1 : suspende todas as conexões de transporte associadas a uma dada conexão de rede. A decisão de encerrar as conexões de transporte ou reiniciá-las sobre outra conexão de rede é tomada pela componente de conexão.
- E_2 : inicia uma conexão de transporte. Os parâmetros dessa conexão são fornecidos por uma primitiva T-CONNECT.request que foi tratada pela componente de estação.

2.3 Aspectos Problemáticos da Especificação de Implementação

Durante a elaboração desta especificação de implementação alguns aspectos do protocolo foram detalhados como demasiadamente vagos para permitir sua especificação mais formal. Cabe ressaltar que tais problemas não afetam a interoperabilidade de implementações.

a) Controle de Fluxo

O protocolo especifica os mecanismos de controle de fluxo a serem utilizados pela entidade de transporte na comunicação com a sua entidade parceira. Todavia, o instante em que estes mecanismos são utilizados e os seus reflexos no serviço oferecido ao usuário de transporte não são definidos.

Dizer que estes são aspectos de implementação que não cabe definir na especificação do protocolo parece-nos uma simplificação apressada. Com efeito, a impossibilidade de transferir dados numa conexão de transporte redundante, necessariamente, na momentânea impossibilidade de prestação de serviço que deve ser comunicada aos usuários de transporte.

Por outro lado a decisão de interromper a troca de dados na camada de transporte tem que ser condicionada pela impossibilidade do usuário de transporte em receber aqueles dados.

Esta troca de informação, que nos parece absolutamente necessária para a descrição dos procedimentos da entidade de transporte, implica na definição de primitivas de serviço que a conduzam entre o usuário de transporte e a respectiva entidade.

Observa-se que parece não existir um consenso na ISO quando à definição de primitivos de controle de fluxo, pois se por um lado elas não aparecem nas descrições de serviços, por outro lado são definidas em alguns protocolos, com o LLC [6], por exemplo. Da mesma forma, um recente estudo sobre protocolos de transporte publicado na literatura especializada [7] ponderou a respeito deste problema que: "é necessário que seja dada alguma orientação aos elementos que executam a implementação e os elementos que especificam os protocolos farão um bom trabalho se pensarem um pouco no papel dos aspectos de armazenamento (buffering) nas arquiteturas elaboradas".

Por estas razões, primitivas de controle de fluxo foram definidas para a interface de transporte e utilizadas na descrição dos procedimentos da componente de controle de fluxo.

Estas primitivas são:

- a) T_FLOW_CONTROL.request: é utilizada pelo usuário de transporte para indicar sua disponibilidade, ou não, para receber TSDU's. Um parâmetro desta primitiva indica se o TS-user está apto ou não a receber dados.
- b) T_FLOW_CONTROL.Indication: é utilizada pela entidade de transporte para indicar ao TS-user se um TSDU por ele enviado pode ou não ser transmitido em função do estado do controle de fluxo da entidade de transporte. Esta primitiva é enviada pela entidade de transporte cada vez que o TS-user solicita a transmissão de um TSDU e um seu parâmetro indica se o TSDU foi aceito ou não para transmissão.

Observa-se o caráter estritamente lógico destas primitivas que apenas usam definir as informações a serem trocadas, sem entrar no mérito de como as informações ou as próprias primitivas são efetivamente representadas.

b) Endereçamento

A fim de identificar os usuários do serviço de transporte, entre os quais a entidade de transporte deve oferecer a troca de dados, as seguintes hipóteses foram admitidas.

Durante a fase de estabelecimento da conexão de transporte, o processo que iniciou a conexão deve fornecer ao nível de transporte o endereço completo da entidade remota. Para suportar as conexões, o nível de transporte oferece para o nível superior uma série de pontos de acesso ao serviço do nível de transporte (TSAPs). Um processo pode acessar o seu nível de transporte através de um ou mais TSAPs.

Assim, uma conexão é definida por um par de TSAPs.

Se um processo P_1 desejar estabelecer uma conexão com um processo remoto P_2 , ele saberá o nº do TSAP remoto ao qual P_2 está associado.

Um TSAP é globalmente endereçável por um endereço de

transporte e pode ser usado por múltiplas conexões. Uma terminação de conexão de transporte (TCEPs) identifica uma conexão em particular dentro de um TSAP. O formato de um endereço de transporte é mostrado na figura 3.

TRANSPORT ADDRESS		TA = NA + TS
NETWORK ADDRESS	TS	Transport Address = Network + Transport Selector

Tamanho do TS = 1 byte (Capacidade para endereçar 256 TSAPs)

Tamanho do Network Address = 8 bytes

Figura 3. - FORMATO DE UM ENDEREÇO DE TRANSPORTE

Uma vez que o projeto da entidade de transporte prevê a sua utilização exclusiva sobre a RENPAC, o endereço de rede foi assumido idêntico ao SNPA ("sub-network point of attachment") daquela rede. Esta hipótese é bastante restritiva e deve ser substituída tão logo um esquema de endereçamento compatível com o proposto pela ISO para a camada de rede esteja disponível.

c) Parâmetros de Qualidade de Serviço

Decidiu-se não utilizar os parâmetros de qualidade de serviço nesta especificação de implementação uma vez que o seu uso efetivo é condicionado pelo desempenho do serviço oferecido nas camadas inferiores. Uma vez que a qualidade oferecida por aquelas camadas não é conhecida, torna-se inviável quantificar o serviço oferecido pela camada de transporte.

Considera-se, na verdade, que este assunto deve ser separadamente estudado, definindo-se um perfil vertical de utilização (isto é, um perfil para todas as camadas) a partir das características elétricas do meio físico utilizado.

3. Aspectos de Implementação do Protocolo de Transporte Classe

2

Uma vez completada a especificação de implementação,

a tarefa de implementação propriamente dita pode ser iniciada. Dentro da metodologia proposta, esta tarefa deve ser dividida nas seguintes etapas:

- a) definição do ambiente de programação
- b) definição das interfaces concretas entre os processos
- c) especificação dos algoritmos de implementação

Destaca-se em seguida os principais problemas e soluções adotadas em cada uma destas etapas

3.1 Definição do ambiente de programação

A implementação da camada de transporte requer a utilização de processos, assincronamente ativados, que possam realizar concorrentemente as tarefas do protocolo. Naturalmente estes processos devem ter a habilidade de comunicarem-se entre si e devem ter à sua disposição temporizadores que sejam capazes de controlar o acionamento dos processos em instantes adequadas.

Naturalmente é possível implementar todas estas características diretamente sobre o "hardware" do micro, todavia tal solução seria fortemente dependente da máquina e exigiria uma profunda interação com o sistema operacional nativo. Desta forma a maior parte do esforço de implementação seria dedicada à solução de problemas de "software" básico, ao invés de se otimizar a entidade que implementa o protocolo propriamente dita.

Por estas razões decidiu-se utilizar um núcleo multi-tarefa já disponível que fornecesse o ambiente de programação adequado para o desenvolvimento da entidade de transporte. O sistema utilizado e não obstante, tal solução apresenta alguns inconvenientes:

- ela redundava numa baixa transportabilidade de programa desenvolvido;
- não permite a coexistência de outras aplicações no mesmo microcomputador;
- impõe a linguagem de programação, no caso PLM (8)

Estas dificuldades são porém contornáveis considerando que os objetivos de conformidade e de interoperabilidade do trabalho descrito.

3.2 Definição dos Interfaces Concretos

Esta etapa consiste em adotar um formato concreto para as primitivas de interface entre os processos. Apesar de não se utilizar nas TPDUs todos os parâmetros possíveis, preferiu-se manter uma ampla gama de parâmetros na interface a fim de permitir a utilização da mesma interface em outros perfis.

Assim, por exemplo, a figura 4 apresenta o formato adotado para as primitivas T_CONNECT.request e T_CONNECT.Indications.

C A M P O S	TAMANHO(BYTE)	T I P O	V A L O R	
Identificador da Primitiva (CODE)	1	Inteiro positivo	00	ID Primitiva
Identif. de Conexão	1	Inteiro positivo	-	
TSAP REMOTO (TS)	1	Inteiro positivo	User	Endereço chamado
Network Address Remoto	8	String 8 caracteres	User	
TSAP LOCAL (TS)	1	Inteiro positivo	User	Endereço chamador
Dados Expedidos	1	Booleano	0-não utilizar (FALSE) ≠0-utilizar(TRUE)	Opção de dados expressos
Tempo de Conexão	1	Inteiro (unidades de 1s)	User	Parâmetros de QOS
Tempo de Desconexão	1	Inteiro (unidades de 1s)	User	
Controle dos Parâmetros	1	Inteiro positivo	Vide Nota 1	
Parte Opcional			Vide Nota 2	
Tamanho dos Dados	1	Inteiro positivo	User	Dados do Usuário
Dados do Usuário	Até 32	String até 32 car	User	

Figura 4 - Mapeamento das Primitivas T_CONNECT.request e T_CONNECT.Indication

Observa-se que o formato depende do tipo de perfil utilizado. Assim, por exemplo, o campo "Dados Expedidos" é necessário para que o usuário informe se deseja ou não a utilização daquele serviço. Os campos destinados aos parâmetros de qualidade de serviço são opcionais uma vez que a representação destes parâmetros é complexa e não são utilizados dentro do atual perfil.

3.3 Especificação dos Algoritmos de Implementação

Finalmente os procedimentos do protocolo, já descritos na tabela de transição de estados, são detalhados a nível de diagramas estruturadas. Observa-se que deve existir uma estreita relação entre as tabelas e os diagramas de modo a reduzir o número de erros de implementação. A figura 5 apresenta o diagrama estruturado do estado ESPERA CONEXÃO correspondente ao segundo estado da tabela 3. Comparando a tabela com o diagrama vê-se a correspondência entre os eventos, sendo que os eventos E_R foi expandido por simplicidade de implementação.

Espera Conexão						
Case evento of						
T_CONNECT. REQUEST	T_DISCONNECT REQUEST	N_CONNECT.CONFIRMATION	N_DISCONNECT INDICATION	T1 Estourou	Outros eventos	
Descartado	Envia N_DISC.RE QUEST	Aceitação da conexão		Envia T_DISCONNECT INDICATION	Envia T_DISCONNECT INDICATION	CALL CONEX (evento)
	Desliga T1	OK	Envia T_DISCONNECT INDICATION	(causa: cone- xão de rede não disponí- vel)	Envia N_DISCONNECT. REQUEST	
	EST_DA_ESTA- ÇÃO= repouso	CALL CONEX. (evento E2)	Desliga T1	Desliga T1	Desliga T1	EST_DA_ESTA- ÇÃO=repouso
		Desliga T1	Desliga T1	Desliga T1	Desliga T1	Desliga T1
		EST_DA_ESTA- ÇÃO = aberto	EST_DA_ESTA- ÇÃO=repouso	EST_DA_ESTA- ÇÃO=repouso	EST_DA_ESTA- ÇÃO=repouso	

Figura 5 - Diagrama de implementação para o estado ESPERA CONEXÃO da componente de estação

CONCLUSÕES

A metodologia utilizada permitiu a implementação do protocolo de transporte de forma relativamente rápida e através de refinamentos sucessivos, ou seja a estruturação da especificação de implementação em componentes, que garante que o programa obtido se aproxima da descrição informal do protocolo da qual se parte. A garantia da coincidência integral entre o protocolo e sua implementação transcende as limitações desta metodologia.

As principais vantagens que a metodologia apresenta correspondem a possibilidade de verificação da consistência das suas diversas etapas com as etapas anteriores, a produção da documentação do projeto durante a sua execução, a discussão das opções que a especificação do protocolo oferece de forma independente do ambiente onde o protocolo é alojado.

Comparando a metodologia proposta com outras técnicas utilizadas, observa-se que ela permite uma mistura em doses convenientes das técnicas de programação de máquinas de estados finais, receptora de eventos, com metodologias utilizadas na programação de sistemas distribuídos.

A metodologia apresenta ainda um forte interesse didático na medida em que permite o aprendizado da realização de uma especificação de implementação de forma completa por si só, e que conduz à conclusão da importância da utilização de técnicas de descrição formal na implementação de protocolos.

No que diz respeito à implementação propriamente dita pôde-se atingir o objetivo de compatibilidade de uso de microcomputadores sem necessidade de alterar o seu "hardware". A utilização de um núcleo de sistema operacional multitarefa revelou-se necessária e a sua incompatibilidade com o sistema DOS e com seus aplicativos específicos deve ser considerado como uma restrição do núcleo multitarefa utilizado neste caso e não com uma restrição da metodologia de desenvolvimento proposta. Desta forma, pode-se concluir que para atingir a meta da transportabilidade do programa obtido não é necessária revisão da metodologia de implementação, mas um investimento na área de "software" básico.

Ressalva-se que a restrição imposta pela falta de recursos do sistema operacional não afeta um bom desempenho num teste de conformidade e/ou de interoperabilidade do produto obtido.

A continuação deste trabalho reside na automatização da atividade de implementação e de testes dos protocolos. No que diz respeito à implementação é interessante introduzir o uso de técnicas de especificação formal e ferramentas próprias.

Quanto aos testes impõe-se a necessidade da utilização de uma metodologia de geração de sequências de testes (de preferência a partir da especificação em linguagem formal) e a elaboração de um sistema testador.

BIBLIOGRAFIA

- [1] ISO, 8073 "Information Processing Systems - Open Systems Interconnection-Connection Oriented Transport Protocol Specification", 1986.
- [2] ISO, 8072 "Information Processing Systems - Open Systems Interconnection-Transport Service Definition", 1986.
- [3] P.LONERGAN; S. STIUBIENER; S.B. DA SILVA; W.L. ZUCCHI; W.V. RUGGIERO - "Proposta de uma Metodologia de Especificação para Protocolos de Comunicação" - Anais do 7º SBRC, Porto Alegre, 1989.
- [4] ISO/TC97 SC21 N1575, DP9074 - "Information Processing Systems - Estelle a Formal Technique Based on an Extended State Transition Model" - 1986.
- [5] "Estudo de Implementação do Protocolo de Transporte ISO 8074 Classe 2 em Microcomputadores" - Relatório do Programa de Trabalho para o Desenvolvimento Conjunto de Pesquisa em Comunicação de Dados Existente entre Centro Científico Rio , IBM Brasil e L.S.D do Departamento de Engenharia de Eletricidade, EPUSP.
- [6] "Information Processing Systems Local Area Network, Logical Line Control", ISO 8802-2, 1987.
- [7] DAVID, D.; CHARLE ET TALL - "An Analysis of TCP Processing Overhead" - IEEE Communications Magazine, June 1989, Vol 27, nº 6.
- [8] Manual de Usuário do SC (Versão SC 86) - FDTE, EPUSP, 1986.