COMUNICAÇÃO DE DADOS
ENTRE INSTITUIÇÕES BANCÁRIAS:
O CASO DO PROTOCOLO PDD

Fernando Nunes Brito e Abreu
Luís António das Neves Abreu
Vitor Manuel Teles Rodrigues
Vitor Manuel Paulino Vargas
(IST/INESC)

A instalação e subsequente operação da rede TELEPAC no país veio proporcionar às Instituições Bancárias um novo meio para a comunicação de dados entre os programas de aplicação correndo nos seus computadores. Confinando-se, todavia, a TELEPAC ao serviço de transporte de pacotes formatados conforme à Recomendação X.25 do CCITT, importa desenvolver, sobre este serviço, alguns protocolos "de alto nível"; estando já especificados os protocolos PDD e FTP, a presente comunicação apresenta o primeiro deles, após o que discute a arquitectura de "software" e a metodologia de desenvolvimento seguida recentemente pelos autores na sua codificação concreta.
1. INTRODUÇÃO

A introdução e subsequente entrada em operação da rede pública de comunicação de dados TELEPAC no nosso país veio proporcionar, particularmente às Instituições Bancárias, um novo meio para a comunicação de dados entre os programas de aplicação correndo nos seus computadores.

No contexto do modelo OSI/CCITT, tal rede Telepac, bem como aliás as demais redes públicas de comunicação de dados em outros países, confina-se aos seus três primeiros níveis: físico, trama e pacote/rede.

Impõe-se, por conseguinte, complementar o serviço oferecido pela rede TELEPAC com um conjunto de protocolos, ditos "de alto nível", correndo nos computadores daquelas Instituições Bancárias.

Um de tais protocolos denomina-se de Protocolo De Diálogo (PDD); preconizado pela Sociedade Interbancária de Serviços (SIBS), destina-se a viabilizar as comunicações entre o Centro de Processamento de Dados da SIBS e os centros informáticos dos Bancos associados.

Pretendendo viabilizar no futuro qualquer tipo de aplicação, o objectivo imediato do PDD é de suportar as necessidades de comunicação da aplicação ATM, ao serviço da rede Multibanco ou de outra rede internacional similar – e é em torno dele, e da metodologia seguida na sua realização pelos autores deste texto, que este se organiza.
2. O PROTOCOLO PDD

De raiz, o protocolo PDD privilegia o uso da Rede Pública de Comunicação de Dados (com acesso conforme à recomendação X.25 do CCITT) como a forma tecnicamente mais adequada ao estabelecimento de uma rede de comunicação ponto a ponto, que abranja não só parceiros nacionais como outras redes internacionais (cfr fig 1).

![Diagrama do protocolo PDD](image)
Com essa ressalva, o modelo de comunicação subjacente ao PDD faz apelo a três "níveis" maiores:

- O nível inferior (nível de rede), a cargo da interface com o "software" X.25 - por sua vez dedicado à transferência de pacotes entre os computadores das Instituições Bancárias envolvidas na comunicação;
- O nível intermédio (nível de diálogo) - PDD -, centrado nas funções de transporte e sessão aplicacional;
- O nível superior (nível aplicacional), enfim, responsável pela construção e interpretação dos dados aplicacionais.

A fig.2 ilustra a modelação da comunicação, em níveis, e o tipo de informação trocada.
Por ai se infere que, confronando com o Modelo de Interligação de Sistemas Abertos (OSI - Open Systems Interconnection) da ISO, o protocolo PDD assume as funções que, nesse Modelo, são relegadas para os níveis transporte e sessão - ficando os demais níveis superiores (apresentação e aplicação) a cargo do programa de aplicação.

3. OS NIVEIS DO MODELO PDD

3.1 O nível Rede:

A máquina sobre que os autores desenvolveram o protocolo PDD é um VAX/VMS. Desde logo, no tocante ao nível de rede, e uma vez que as comunicações são efectuadas através da rede pública de comutação de pacotes (TELEPAC), que utiliza o protocolo X.25, basta recorrer aos serviços postos à disposição pelo "software" VAX/PSI. Este permite uma invocação dos serviços do nível rede, nomeadamente: estabelecimento de chamadas e sua finalização, sub-endereçamento, controlo de congestão...

3.2 O nível Transporte do PDD:

O protocolo de diálogo, conforme às especificações da SIBS (PDD R.3 - 87/10/23), não se encontra explicitamente estruturado por níveis.

Uma afinada percepção das suas tarefas veio todavia a tornar mandatória a sua estrutura em dois níveis, a saber: de transporte e sessão, como segue:

O nível de transporte utiliza os serviços do nível rede para garantir um serviço ponto-a-ponto entre os computadores em comunicação com as características desejáveis para os níveis superiores. Tal é conseguido através da informação que é veiculada nos cabeçalhos (inseridos na emissão e extraídos na recepção) e que diz respeito a:
• **Multiplexagem de mensagens**, através do uso de ILTs (Indicadores Lógicos de Terminal). O ILT 0 é exclusivamente usado na troca de mensagens de controlo, enquanto que os outros ILTs são unicamente para dados.

• **Controlo de Fluxo**, através do recurso a uma janela de comprimento negociano (a dimensão da janela corresponde ao maior número de “pedidos” que podem ser enviados sem ser recebida a subsequente resposta). A dimensão da janela corresponde ao número máximo de ILTs a utilizar.

• **Controlo da sequencialidade das mensagens de dados** (destinados à aplicação), utilizando um número de sequência de mensagem dentro de cada ILT. Quanto às mensagens de controlo (destinadas ao nível sessão), e embora não se pretenda a validação da sequencialidade, o número de sequência da mensagem é utilizado para “carimbar” um dado pedido e, assim, identificar univocamente a sua resposta.

• **Segmentação de mensagens em blocos** através da atribuição de um número de bloco (embora as especificações do protocolo e o “software” desenhado prevejam esta possibilidade, de momento a SIBS está usando apenas mensagens monoblocos).

• **Detecção de erros** - São detectadas, entre outras, as seguintes situações:
  - número de ILT incorrecto;
  - número de sequência de mensagem incorrecto;
  - número de bloco incorrecto;
  - chegada de uma mensagem num ILT com uma mensagem ainda não tratada;
  - mensagens fora do contexto, (por exemplo, mensagens de recuperação sem se estar em estado de recuperação, ou mensagens de resposta a pedidos que não foram emitidos);
  - tipo de mensagem de controlo de dados incorrecto.

Em qualquer dos casos, as mensagens são devolvidas ao computador interlocutor, com a indicação do motivo da sua rejeição.
3.3 O Nível Sessão do PDD

O nível de sessão permite que as duas entidades em comunicação organizem e sincronizem o seu diálogo, por forma a permitir a troca de dados. Este protocolo destina-se, numa primeira fase, a suportar o serviço de atendimento automático nas caixas Multibanco. Os pedidos são centralizados pelo computador da SIBS, que por sua vez os distribui a cada um dos bancos envolvidos, com quem mantém aberta uma sessão de diálogo através deste protocolo. No que se segue, e para individualizar as diferentes partes em comunicação, recorrer-se-á às designações de computador “emissor de pedidos” (SIBS) e de computador “receptor de pedidos” (banco).

Os mecanismos básicos envolvidos numa sessão são: a sua abertura, suspensão controlada do tráfego de dados e fecho de sessão; todos eles são desencadeados pelo “receptor de pedidos”. Em certos casos (respostas não entregues à aplicação que as esperava, por exemplo), qualquer um dos interlocutores poderá desencadear o mecanismo de recuperação da sessão. A realização a que este artigo se refere é a do “receptor de pedidos”.

A activação dos mecanismos atrás referidos depende principalmente da iniciativa de um operador. Este tem ao seu dispor os seguintes procedimentos (cfr fig.3):

- **Procedimento de Abertura de Sessão**: Em estado de sessão inactiva (estado 0) o operador poderá invocar o comando "INIT WINDOW<dimensão da janela>"; é enviado para o outro interlocutor a mensagem de pedido de abertura de sessão. A sessão passa para o estado 1, em que aguarda a resposta ao pedido de abertura. Caso a resposta não seja afirmativa, ou seja ultrapassado um período de tempo pré-fixado (temporizador programável) sem resposta, a sessão evolui novamente para o estado de sessão inactiva (estado 0).
Caso a resposta seja afirmativa, a mensagem correspondente conterá a informação da aceitação da dimensão da janela proposta ou um valor inferior ao proposto, que será o doravante aceite pelas duas partes. A sessão evoluirá para o estado de sessão activa (estado 2) em que o programa de aplicação dialogará com o seu correspondente correndo no interlocutor remoto.
- **Procedimento de Suspensão de Sessão:** A suspensão da sessão é solicitada com o fim de assegurar a conclusão do tráfego de dados. Esta operação permite que seguidamente se possa pedir o fecho de sessão, ou então uma interrupção temporária para permitir algum processamento de manutenção/inspeção. Neste último caso, o operador deverá pedir a reinicialização da sessão. Em estado de sessão ativa (estado 2), o operador poderá invocar o comando "SUSPEND", sendo enviado para o outro interlocutor a mensagem de pedido de suspensão de sessão. A sessão passa então para o estado 5 em que aguarda a resposta ao pedido de suspensão, continuando contudo a processarem-se normalmente as respostas aos pedidos (mensagens de dados) recebidos. O outro interlocutor ("emissor de pedidos"), é após receber o pedido de suspensão, cessa o envio de pedidos, mas apenas responde após receber as respostas a todos os pedidos enviados. Caso isso não aconteça, ele provocará uma recuperação da sessão.

O estado de sessão suspensa (estado 6) poderá ser também atingido sempre que haja uma recuperação, se o operador emitir o comando "SUSPEND AFTER RECUP". Por defeito (ou emissão do comando "ACTIVATE AFTER RECUP"), após uma recuperação a sessão evolui para o estado de sessão ativa (estado 2).

- **Procedimento de Reactivação da Sessão:** Em estado de sessão suspensa (estado 6), o operador poderá reactivar a sessão, se digitar o comando "REINIT"; é enviada para o outro interlocutor a mensagem de controlo correspondente. Aqui, não é necessária a indicação da dimensão da janela, uma vez que permanece em uso aquela já negociada no início. Quando o "emissor de pedidos" envia a resposta, passa-se de novo a sessão ativa (estado 2), retomando-se o tráfego de dados entre os programas de aplicação dos dois interlocutores.
• Procedimento de fecho da sessão: Em estado de sessão suspensa (estado 6), pode o "receptor de pedidos" decidir a conclusão definitiva do tráfego de dados. Para isso, bastará ao operador digital o comando "EXIT". Também neste caso se aguardará a resposta ao pedido de fecho (estado 8), após a chegada da qual se passará ao estado inicial de sessão inactiva (estado 0).

• Procedimento de Recuperação: Este procedimento poderá ser, como já se disse, desencadeado por qualquer um dos interlocutores, após situações várias como sejam a queda da ligação X.25, falha de energia, chegada de mensagens fora do contexto, ou, inclusivamente, por intervenção do operador. Para evitar casos em que um interlocutor não dê pela falha do outro, um pedido de recuperação será sempre respondido afirmativamente, seguindo-se o processamento correspondente. O processamento de recuperação envolve um re-envio, por parte do "emissor de pedidos", de todas as mensagens de dados (devidamente identificadas como sendo de recuperação) que tinha na sua janela e que ainda não haviam sido respondidas. Por seu lado, o "receptor de pedidos", ao receber as mensagens de dados e vendo que são de recuperação, verifica se já tinham sido processadas (pela aplicação) antes do procedimento da recuperação. Em caso afirmativo, devolve a mesma resposta que anteriormente, sem interpelar a aplicação. Caso contrário, os pedidos são submetidos normalmente à aplicação. O "emissor de pedidos" provoca o fim de recuperação quando obtiver respostas a todas as mensagens pendentes.

3.4 O nível Aplicação

O desenvolvimento das aplicações reais a instalar, suportando-se nos serviços fornecidos pelos níveis atrás referidos, não foi contemplado no trabalho aqui em referência.
Para efeitos de simulação e teste da interface com os níveis inferiores, todavia, foi construída uma aplicação interactiva elementar, usando primitivas simples do tipo "ENVIA MENSAGENS DE DADOS" e " RECEBE MENSAGEM DE DADOS". A recepção é feita de uma fila de espera com estrutura FIFO ("FIRST IN FIRST OUT") onde são depositadas as mensagens procedentes dos níveis inferiores.

4. SIMULAÇÃO DO PROTOCOLO

Dado que um protocolo se estabelece entre duas entidades; e que, no caso vertente, o diálogo entre ambos os lados (computador da SIBS e computador de banco associado) é assimétrico, surgiu, para efeitos de teste do protocolo, a necessidade de dispor de uma realização do "emissor de pedidos", doravante referenciado como "nó remoto" (em contraposição com a do "receptor de pedidos" dito de "nó local").

Com efeito, uma outra opção a seguir, a de efectuar os testes apenas na fase final do desenho, directamente com a versão da SIBS, apresentava vários inconvenientes: 1) não permitiria o teste modular, sempre mais acessível; 2) ficaria dependente da disponibilidade técnica e humana da parte da SIBS; 3) bem como acarretaria custos e atrasos (tempos de estabelecimentos de chamadas, tempos de resposta) pela utilização directa da TELEPAC.

Optou-se assim por uma hipótese alternativa que evitou os problemas supracitados e que passou por:
- **Simulação do “nó remoto”** (procedimentos do lado do “emissor de pedidos”) como um processo associado a um utilizador diferente do que desencadeia o “nó local”.

- **Simulação da comunicação via rede**, por um mecanismo de partilha de caixas do correio entre utilizadores (permitido pelo sistema operativo).

- **Simulação do nível aplicacional**, faseada da seguinte forma (cfr fig 5):

1°) Para testar a comunicação no sentido “nó local” – “nó remoto”, foi simulada uma sequência de mensagens aplicacionais num ficheiro INAPL.DEBUG, resumindo-se a aplicação a enviar essas mensagens para o “nó remoto” utilizando os serviços dos níveis inferiores. Essa mesma sequência foi comparada com a recebida no “nó remoto” (ficheiro OUTPSI.LOG).
2º) Para testar a comunicação no sentido nó remoto - nó local, foi simulada uma sequência de mensagens aplicacionais no ficheiro TESTPSI.DEBUG. A simulação da aplicação do lado remoto envia sequencialmente essas mensagens para o nó local, onde são "recolhidas" no ficheiro OUTAPL.LOG.

3º) Finalmente, para testar a comunicação simultânea nos dois sentidos, simulou-se a aplicação do lado do nó remoto como sendo um "curto-circuito lógico" isto é, todas as mensagens recebidas são reenviadas. Assim, comparou-se, repetindo o procedimento descrito no parágrafo 1º, o conteúdo dos ficheiros INAPL.DEBUG e OUTAPL.LOG. (Da mesma forma se fez o teste do outro lado, isto é fazendo o curto-circuito do lado do nó local e comparando os conteúdos de TESTPSI.DEBUG e OUTPSI.LOG).

Foram ainda utilizados outros ficheiros auxiliares, como se pode ver na figura: para poder analisar a evolução de uma sessão (guardando as codificações dos estados e das condições de transição em STATUS_PDD.LOG), para guardar as mensagens rejeitadas de um e outro lados (ANORMAL_PDD.LOG e ORRPSI.LOG) e para a análise dos cabeçalhos inseridos e retirados pelo protocolo PDD em um e outro lado (SEND_PSI.LOG, RECEIVE_PSI.LOG e DAT_PDD.LOG).

5. O "SOFTWARE" DO SISTEMA PDD

5.1. A Arquitetura de Processos

A realização do protocolo de diálogo (PDD) já descrito anteriormente foi feita com base em três processos (cfr fig 5):
- O processo PDD, que virtualiza os níveis transporte e sessão acima descritos, garantindo a comunicação quer com o nível imediatamente mais abaixo (nível de rede), quer com o nível acima (aplicação), quer ainda com o operador.

- O processo API, que suportará a aplicação a desenvolver conforme as necessidades da Instituição Bancária. Presentemente, encontra-se configurada uma versão que simula estes níveis, por forma a permitir os procedimentos de teste já descritos.

- O processo OPER, que permite a intervenção de um operador quer na monitorização, quer na evolução da sessão de diálogo. A razão de ser de este processo existir como tal prende-se com o facto de este poder ser activado ou desactivado sempre que desejado, por forma a não consumir recursos (tempo do processador) desnecessariamente.

Diagrama:

```
<table>
<thead>
<tr>
<th>STATUS_MACHINE</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDD_FUNCTIONS</td>
</tr>
<tr>
<td>TRANSPORT</td>
</tr>
<tr>
<td>COMMUN_MOD</td>
</tr>
<tr>
<td>COMMUN_MOD</td>
</tr>
<tr>
<td>COMMUN_MOD</td>
</tr>
<tr>
<td>PARSER_FUNCTIONS</td>
</tr>
<tr>
<td>PARSER_OPER</td>
</tr>
<tr>
<td>OPER_MAIN</td>
</tr>
<tr>
<td>Nível Rede</td>
</tr>
<tr>
<td>PSI</td>
</tr>
<tr>
<td>APL_PRIMITIV</td>
</tr>
<tr>
<td>Aplicação</td>
</tr>
</tbody>
</table>

Processo PDD  Nível Sessão
             Nível Transporte

Processo APL

Processo OPER

Operador
```
A comunicação entre estes processos é feita através de canais unidireccionais associados a caixas de correio.
Para garantir uma transferência sem problemas de sincronização e adaptação de ritmos, foram utilizadas na recepção, em qualquer dos casos, filas de espera com estrutura FIFO.

A comunicação entre processos foi modularizada, tendo-se desenvolvido um conjunto de funções básicas que foram partilhadas pelos quatro sentidos de comunicação.

Resta referir a comunicação com o nível da rede, virtualizada pelo PSI (Packet Switching Interface) do VAX. Esta comunicação efectua-se através de um canal bidireccional associado à caixa do correio reservada "_.NWAO".
Foram desenvolvidas duas rotinas básicas para utilizar este canal que são do tipo "ENVIA PACOTE" e " RECEBE PACOTE".

5.2 Estrutura do "Software"

O sistema está funcionalmente modularizado em blocos que:

- reflectem o nível de protocolo que realizam;
- centralizam funções de processamento comuns;
- isolam funções dependentes da máquina e do sistema operativo

A figura 6 representa esquematicamente os seus blocos constituintes, dependências, processos a que estão assignados e nível de protocolo que realizam.
Resumidamente,

- Módulo COMMUN_MOD

Comum a todos os processos intervenientes, o módulo COMMUN_MOD centraliza:

as funções de criação e gestão das estruturas de comunicação entre processos (caixas de correio, filas de espera associadas);

as funções básicas de transferência assíncrona de informação entre processos.
• Módulo SCI

As funções dependentes do sistema operativo estão a cargo do módulo SCI.

• Módulos associados ao processo PDD

O nível de rede é realizado no módulo PSI, encarregue da interface com o X.25.

O nível de transporte é realizado no módulo TRANSPORT.

Incumbido de invocar as funções adequadas com os parâmetros adequados do módulo TRANSPORT, o módulo PDD_FUNCTIONS é por sua vez controlado por STATUS_MACHINE – que evolui conforme ao diagrama de estados do Protocolo De Diálogo:

controlando e validando as condições de transição entre estados;

invocando a função que realiza a acção associada a cada condição de transição;

transitando para o novo estado;

• Módulos associados ao processo OPER:

O módulo OPER_MAIN encarrega-se da interface com o operador, permitindo a monitorização e o controlo do sistema.

Os comandos de controlo digitados são analisados léxica e sintáticamente pelo módulo PARSER_OPER, encarregue de invocar a função associada de PARSER_FUNCTIONS.

Este último módulo encaminha a mensagem de controlo adequada para o processo adequado.

• Módulos associados ao processo APL

O módulo APL_PRIMITIV possui os primitivas de comunicação que suportarão a interface entre o Protocolo de Diálogo e os programas de aplicação a desenvolver ulteriormente.
6. SUMARIO E PERSPECTIVAS FUTURAS

A utilização da Telepac como meio de comunicação de dados entre computadores de Instituições Bancárias convoca à especificação e desenvolvimento de protocolos de "alto nível". Entre estes, sobressai o protocolo PDD, de que foi alvo a presente comunicação.

O "software" desenhado pelos autores foi já testado em ligação directa (de um computador VAX do banco BIC) com a versão SIBS, via TELEPAC, tendo o teste intensivo decorrido com pleno sucesso:

Mercê da prévia simulação do protocolo, as pequenas alterações a efectuar confinaram-se apenas aos problemas de funcionamento em tempo real (adaptação aos tempos reais de estabelecimento de chamadas, aos atrasos dos tempos de resposta, etc).

Tendo sido realizadas as primitivas básicas de suporte à aplicação, bastará agora a definição pela Instituição Bancária das aplicações específicas pretendidas para a sua imediata instalação sobre o Protocolo De Diálogo.

Ademais, pela modularização conferida ao "software", torna-se algo trivial a sua cooperação para outros protocolos de alto nível: em particular, tem sido bastamente utilizado para o desenho do protocolo PTF (Protocolo de Transferência de Ficheiros) recomendado pela SIBS, em fase de acabamento pelos mesmos autores.

O "software" encontra-se escrito em C, sobre VAX/VMS; todavia, e visando-se a sua portabilidade para outros computadores, o conjunto de procedimentos dependentes do sistema operativo encontram-se compactados num mesmo e único módulo.