UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
DEPARTAMENTO DE AUTOMAÇÃO E SISTEMAS

Das 5331- sistemas distribuídos e REDES de computadores para controle e AUTOMAÇÃO industrial

PROF. DR.-ING. MARCELO RICARDO STEMMER
Versão 2001

1. INTRODUÇÃO GERAL

2. AS REDES DE COMUNICAÇÃO

2.1. INTRODUÇÃO

2.1.1. HISTÓRICO DAS REDES DE COMUNICAÇÃO

A evolução da microeletrônica e da informática tem possibilitado a obtenção de processadores e outros componentes de computadores cada vez mais potentes e velozes, em um tamanho mais reduzido e com um preço acessível a um número cada vez maior de pessoas. Os microprocessadores existentes hoje em dia, que ocupam espaço menor do que uma caixa de fósforos, substituem e ultrapassam as capacidades dos computadores de alguns anos atrás, que ocupavam salas inteiras. Estes eram máquinas bastante complexas no que diz respeito à sua utilização e que ficavam em salas isoladas, às quais muito poucas pessoas tinham acesso, sendo operadas apenas por especialistas (analistas de sistema). Os usuários daqueles computadores normalmente submetiam seus programas aplicativos como “jobs” (ou tarefas) que executavam sem qualquer interação com o autor do programa.

Uma primeira tentativa de interação com o computador ocorreu no início dos anos 60, com a técnica de “time-sharing”, que foi o resultado do desenvolvimento das teleimpressoras e da tecnologia de transmissão de dados. Nesta técnica um conjunto de terminais era conectado a um computador central através de linhas de comunicação de baixa velocidade, o que permitia aos usuários interagir com os seus programas. A necessidade de conexão de terminais para o processamento interativo foi o ponto de partida para o estabelecimento de necessidades de comunicação nos computadores. A técnica de time-sharing permitia a um grande conjunto de usuários o compartilhamento de um único computador para a resolução de uma grande diversidade de problemas e as aplicações desenvolvidas foram cada vez mais se multiplicando e se diversificando (cálculos complexos, produção de relatórios, ensino de programação, aplicações militares, etc). Este aumento na demanda implicava numa necessidade crescente de atualizações e incrementos nas capacidades de armazenamento e de cálculo na unidade central, o que nem sempre era viável ou possível, dado que os computadores do tipo "mainframe" nem sempre eram adaptados para suportar determinadas extensões.

Nos anos 70, com o surgimento dos minicomputadores, foi possível adaptar as capacidades de processamento às reais necessidades de uma dada aplicação. Além disso, dado que em uma empresa um grande número de usuários operavam sobre conjuntos comuns de

informações, a necessidade do compartilhamento de dados, de dispositivos de armazenamento e de periféricos entre os vários departamentos de uma empresa deu um novo impulso aos trabalhos no sentido de se resolver os problemas de comunicação entre os computadores. Estes novos tipos de aplicações exigiam uma velocidade e uma capacidade de transmissão muito mais elevadas que no caso da conexão de terminais a um computador central. Assim, com a utilização de minicomputadores interconectados, obtinha-se uma capacidade de processamento superior àquela possível com a utilização dos mainframes. Outro aspecto interessante é que as redes podiam ser estendidas em função das necessidades de processamento das aplicações. Além disso, a modularidade natural das redes de computadores era tal que uma falha num minicomputador (ou na comunicação via rede) tinha um efeito bastante limitado em relação ao processamento global.

Atualmente, as vantagens dos sistemas distribuídos e interconectados é uma evidência reconhecida para as aplicações mais diversas, desde a automação de escritórios até o controle de processos, passando por aplicações de gerenciamento bancário, reservas de passagens aéreas, processamento de texto, correio eletrônico, etc...

2.1.2. IMPORTÂNCIA DAS REDES DE COMUNICAÇÃO

Um grande número de empresas possui atualmente uma quantidade relativamente grande de computadores operando nos seus diversos setores. Um exemplo deste fato é aquele de uma empresa que possui diversas fábricas contendo cada uma um computador responsável das atividades de base da fábrica (controle de estoques, controle da produção e, o que também é importante, a produção da folha de pagamentos). Neste exemplo, apesar da possibilidade de operação destes computadores de maneira isolada, é evidente que sua operação seria mais eficiente se eles fossem conectados para, por exemplo, permitir o tratamento das informações de todas as fábricas da empresa. O objetivo da conexão dos diferentes computadores da empresa é permitir o que poderíamos chamar de *compartilhamento de recursos*, ou seja, tornar acessíveis a cada computador todos os dados gerados nas diversas fábricas da empresa.

Um outro ponto importante da existência das Redes de Comunicação é relacionado a um aumento na *confiabilidade* do sistema como um todo. Pode-se, por exemplo, ter multiplicados os arquivos em duas ou mais máquinas para que, em caso de defeito de uma máquina, cópias dos arquivos continuarão acessíveis em outras máquinas. Além disso, o sistema pode operar em regime *degradado* no caso de pane de um computador, sendo que outra máquina pode assumir a sua tarefa. A continuidade de funcionamento de um sistema é ponto importante para um grande número de aplicações, como por exemplo: aplicações militares, bancárias, o controle de tráfego aéreo, etc.

A *redução de custos* é uma outra questão importante da utilização das Redes de Comunicação, uma vez que computadores de pequeno porte apresentam uma menor relação preço/desempenho que os grandes. Assim, sistemas que utilizariam apenas uma máquina de

grande porte e de custo muito elevado podem ser concebidos à base da utilização de um grande número de microcomputadores (ou estações de trabalho) manipulando dados presentes num ou mais servidores de arquivos.

2.1.3. EXTENSÃO E TOPOLOGIA DAS REDES DE COMUNICAÇÃO

2.1.3.1. Redes locais e redes de longa distância

Na seção anterior foram apresentados dois exemplos de implementação de Redes de Comunicação: no primeiro caso, o sistema era composto de diversos computadores espalhados cada um numa fábrica da empresa.

No segundo caso, o sistema era composto de diversos microcomputadores, podendo todos estar localizados na mesma sala ou em salas vizinhas num mesmo edifício.

A diferença na dimensão das Redes de Comunicação introduz diferentes problemas e necessidades e deve, então, ser objeto de uma classificação. No que diz respeito ao exemplo dos microcomputadores, a rede é classificada como sendo uma *Rede de Área Local* (ou *LAN - Local Area Network*), caracterizada particularmente por uma pequena extensão, limitando-se normalmente à interconexão de computadores localizados numa mesma sala, num mesmo prédio ou num campus. Neste último caso, ela recebe a denominação de *CAN* (*Campus Area Network*).

No exemplo da empresa possuindo diversas fábricas, a rede utilizada permitiria conectar computadores localizados em diferentes prédios numa mesma cidade ou mesmo em cidades distantes de uma dada região. Isto caracteriza uma *Rede de Longa Distância* (ou *WAN - Wide Area Network*). Se as estações interligadas estão situadas na mesma cidade, utiliza-se freqüentemente a denominação *MAN* (*Metropolitan Area Network*).

2.1.3.2. As diferentes topologias

Um ponto importante no que diz respeito à concepção de uma rede de comunicação é a definição da maneira como as diferentes estações serão associadas. Inicialmente, podemos distinguir dois tipos principais de concepção: os canais em modo *ponto-a-ponto* e os canais de *difusão*.

Nos *canais em ponto-a-ponto*, a rede é composta de diversas linhas de comunicação, cada linha sendo associada à conexão de um par de estações.

Neste caso, se duas estações devem se comunicar sem o compartilhamento de um cabo, a comunicação será feita de modo indireto, através de uma terceira estação. Assim, quando uma mensagem (ou *pacote*) é enviada de uma estação a outra de forma indireta (ou seja, através de uma ou mais estações intermediárias), ela será recebida integralmente por cada estação e, uma vez que a linha de saída da estação considerada está livre, retransmitida à estação seguinte. Esta política de transmissão é também conhecida como *"store and forward"* ou comutação de pacotes. A maior parte das redes de longa distância é do tipo ponto-a-ponto.

As redes ponto-a-ponto podem ser concebidas segundo diferentes topologias. As redes locais ponto-a-ponto são caracterizadas normalmente por uma topologia simétrica; as redes de longa distância apresentam geralmente topologias assimétricas. A figura 2.1.1 apresenta as diferentes topologias possíveis nas redes ponto-a-ponto.

Uma outra classe de redes, as *redes de difusão*, é caracterizada pelo compartilhamento, por todas as estações, de uma linha única de comunicação. Neste caso, as mensagens enviadas por uma estação são recebidas por todas as demais conectadas ao suporte, sendo que um campo de endereço contido na mensagem permite identificar o destinatário.

Na recepção, a máquina verifica se o endereço definido no campo corresponde ao seu e, em caso negativo, a mensagem é ignorada. As redes locais pertencem geralmente a esta classe de redes.

Nas redes de difusão, existe a possibilidade de uma estação enviar uma mesma mensagem às demais estações da rede, utilizando um código de endereço especial. Esta forma de comunicação recebe o nome de *Broadcasting*. Neste caso, todas as estações vão tratar a mensagem recebida. Pode-se ainda especificar uma mensagem de modo que esta seja enviada a um subgrupo de estações da rede. Esta forma de comunicação recebe o nome de *Multicasting*. A figura 2.1.2 apresenta algumas topologias possíveis no caso das redes a difusão.

Numa rede em barramento, geralmente uma única máquina é autorizada a cada instante a transmitir uma mensagem — é a estação *mestra do barramento*. As demais estações devem esperar autorização para transmissão. Para isto, um mecanismo de *arbitragem* deve ser implementado para resolver possíveis problemas de conflito (quando duas ou mais estações querem enviar uma mensagem), este mecanismo podendo ser centralizado ou distribuído.

No caso das redes de satélite (ou rádio), cada estação é dotada de uma antena através da qual pode enviar e receber mensagens. Cada estação pode “escutar” o satélite e, em alguns casos, receber diretamente as mensagens enviadas pelas demais estações.

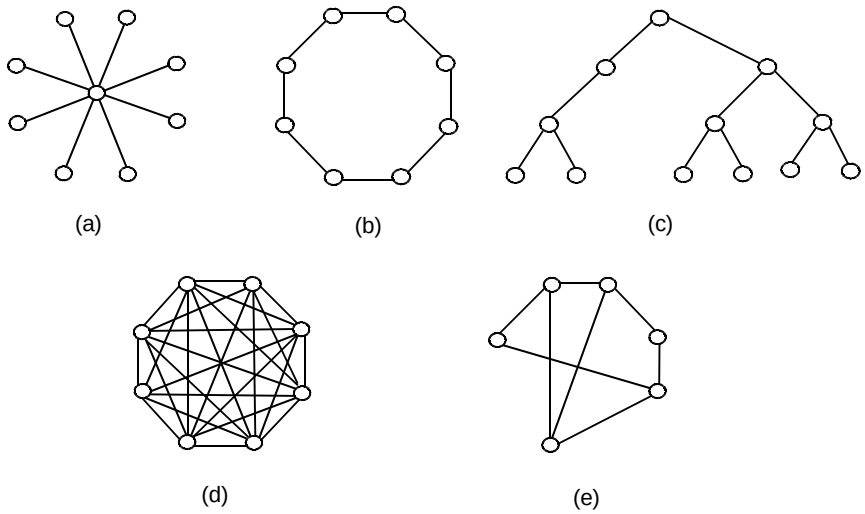


Figura 2.1.1 - Topologias ponto-a-ponto: (a) estrela; (b) anel; (c) árvore; (d) malha regular; (e) malha irregular.

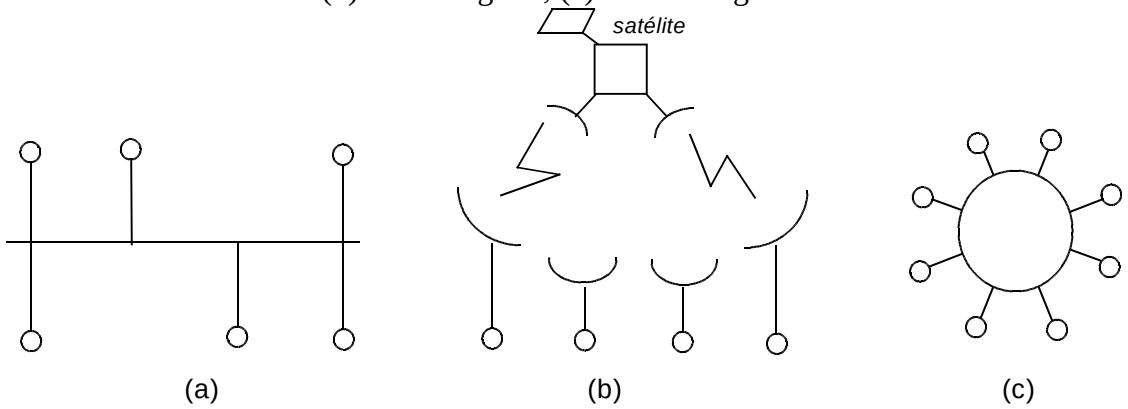


Figura 2.1.2 - Topologias das redes de difusão: (a) barramento; (b) satélite; (c) anel.

No caso do anel, cada bit transmitido é propagado de maneira independente em relação à mensagem (ou pacote) ao qual ele pertence. Em geral, cada bit realiza uma volta completa do anel durante o tempo necessário para a emissão de um certo número de bits, antes mesmo da emissão completa da mensagem. Também nesta topologia, é necessária a implementação de um mecanismo de acesso ao suporte de comunicação. Existem diferentes técnicas para este fim que serão discutidas ao longo do curso.

As redes de difusão podem ainda considerar duas classes de mecanismos de acesso ao suporte de comunicação: estáticas ou dinâmicas. Um exemplo do primeiro caso é a definição de intervalos de tempo durante os quais cada estação tem a posse do canal de comunicação, permitindo então que esta emita a mensagem de maneira cíclica. No entanto, esta política é bastante ineficiente do ponto de vista do envio das mensagens, uma vez que muitas estações não vão enviar mensagens nos intervalos a elas destinadas. Já na outra classe de mecanismos (dinâmicos), o acesso é dado às estações segundo a demanda de envio de mensagens. Nos mecanismos de acesso dinâmicos, pode-se ainda considerar dois casos:

- os mecanismos centralizados, nos quais uma estação central (árbitro) é a responsável da definição do direito de acesso ao suporte de comunicação;
- os mecanismos distribuídos, nos quais cada estação define quando ela vai emitir a mensagem.

Estudaremos estes mecanismos em mais detalhe mais a frente no curso.

2.1.4. ASPECTOS ARQUITETURAIS DAS REDES DE COMUNICAÇÃO

2.1.4.1 Serviços necessários à comunicação

Como visto nas seções precedentes, as redes de computadores podem se caracterizar por diferentes configurações e topologias. Apesar da diversidade no que diz respeito a este aspecto, todas as possíveis configurações têm um objetivo comum: a *transferência de dados*.

O problema que se coloca é então relacionado à especificação dos procedimentos e mecanismos que devem ser implementados para viabilizar o funcionamento da rede. A resolução deste problema é baseada principalmente no conhecimento prévio das funções que devem ser suportadas pela rede, assim como do ambiente no qual ela vai ser inserida. Estes aspectos serão mostrados aqui através de alguns exemplos.

O primeiro exemplo é baseado na política de "time-sharing" mencionada na seção 1. Vamos considerar o caso em que temos apenas um terminal conectado a um computador, como mostrado na figura 2.1.3. Consideremos que um usuário vá servir-se do terminal para processar informações no computador central. Para que isto seja possível, é necessário que o computador central seja dotado do programa necessário ao tratamento daquelas informações. Em caso positivo, o terminal e o computador devem estabelecer um diálogo que permita o bom desenrolar das operações de tratamento das informações. Este diálogo deverá permitir, por exemplo, que o usuário comunique sua intenção (de processar as informações!) ao computador e, em seguida, envie as informações a serem processadas. Uma vez efetuado o tratamento, o computador deve retornar os resultados ao terminal.

Esta seqüência de operações, apesar de aparentemente elementar, requer a satisfação de uma série de condições. Vamos supor, por exemplo, que o computador central e o terminal tenham sido fabricados de forma totalmente independente um do outro, o que pode ter resultado numa diferente filosofia no que diz respeito ao formato das informações. Um primeiro obstáculo a ser vencido é aquele da *linguagem*; o terminal deveria então se adaptar à linguagem do computador central. Resolvido o problema de compreensão, um outro problema encontrado diz respeito aos possíveis *erros de transmissão* que podem ocorrer durante a comunicação, uma vez que as linhas de comunicação estão sujeitas a ruídos e outros fenômenos podendo provocar perdas de informação. Além disso, a *taxa de transmissão* (baudrate) e a *forma de representar os sinais binários* deve ser igual em ambos os lados. Uma outra questão pode ainda estar relacionada à *velocidade de funcionamento dos dois elementos*. Se considerarmos que o computador central opera numa velocidade superior à do terminal, por exemplo, o terminal corre o risco de ser "bombardeado" pelo fluxo de dados vindo do computador, o que vai exigir então o estabelecimento de um mecanismo de controle do fluxo de informação.

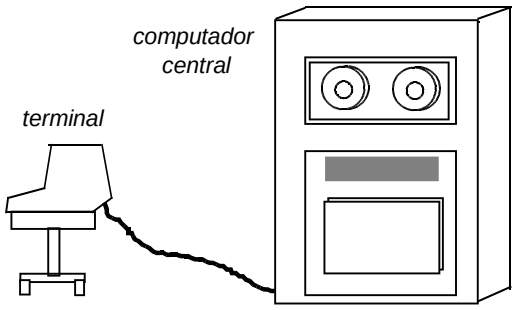


Figura 2.1.3 - Terminal conectado a um computador central.

Resumindo, a rede de comunicação deve, além de suprir as funções de transmissão e tratamento de informações, oferecer serviços de adaptação, detecção e correção de erros de transmissão e controle de fluxo.

Vamos considerar agora que, ao invés de um único terminal, vamos conectar um maior número deles ao computador central (figura 2.1.4).

Aqui, cada terminal pode, a princípio e a qualquer momento, tomar a iniciativa da troca de dados com o computador. Isto significa que cada terminal terá de ser caracterizado por um *endereço* específico, cuja utilização correta vai permitir evitar que o computador central envie as informações aos terminais de maneira indevida. Por outro lado, se o número de terminais conectados ao computador central torna-se relativamente elevado (a fim de permitir a utilização máxima da capacidade de processamento deste), será necessário organizar as interações entre terminais e o computador central em sessões, de tal forma que, ao término de uma sessão entre um terminal e o computador central, este terá liberados determinados elementos (envolvidos naquela sessão) que poderão atender outros terminais em estado de espera.

Ainda, considerando que nem todos os terminais vão efetuar o mesmo tipo de tratamento de forma simultânea, dever-se-á, então, especificar a aplicação associada. Assim, todas as necessidades vistas neste exemplo deverão ser associadas às funcionalidades definidas no exemplo anterior. Mas os problemas não terminam por aqui... (é impossível, no momento, prever onde terminarão os problemas!).

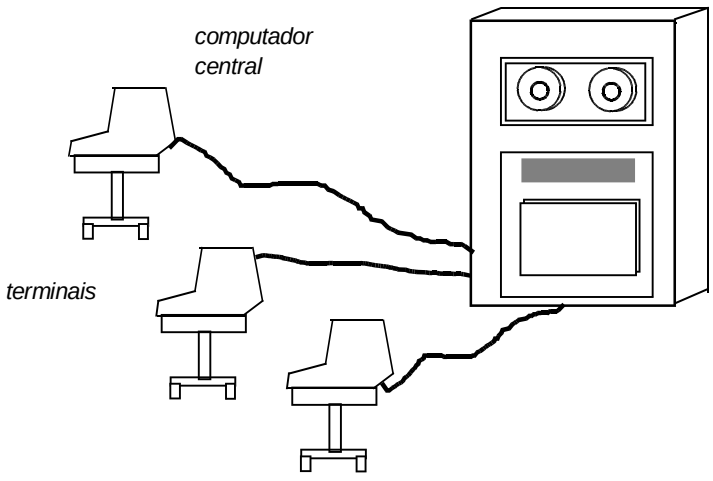


Figura 2.1.4 - Configuração com vários terminais.

Vamos considerar ainda um exemplo, mais particularmente o de uma rede contendo diversos computadores, terminais, etc, cada um destes elementos constituindo um nó da rede (figura 2.1.5). Neste exemplo, os dois elementos envolvidos numa comunicação não serão mais necessariamente adjacentes; além disso, podem existir diversas maneiras de conectá-los, o que conduz a diferentes caminhos de envio de dados.

No exemplo mostrado na figura 2.1.5, os nós 1 e 5 podem ser conectados por pelo menos 10 caminhos diferentes e a escolha de qual caminho utilizar deverá então ser realizada, o que não é uma tarefa tão simples quanto possa parecer. A função de escolha de uma caminho adequado recebe o nome de *Roteamento*.

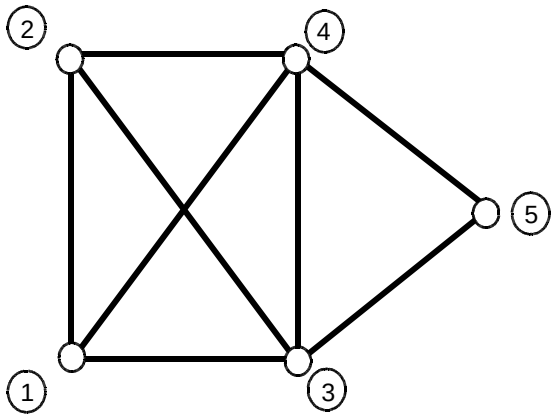


Figura 2.1.5 - Configuração com vários computadores e terminais.

Ainda, se a rede é da classe ponto-a-ponto (ou comutação de pacotes), o sistema de comunicação deve assegurar a correta transmissão (transporte) da informação de um ponto a outro. Particularmente, será necessário garantir que as mensagens enviadas serão recuperadas e reconstituídas na ordem correta no ponto de chegada.

Um requisito também importante é o aspecto da codificação das mensagens de modo a evitar o acesso a informações de parte de usuários alheios ao sistema considerado. A esta função, pode-se eventualmente acrescentar técnicas de *compressão de dados*, necessária se a informação enviada é demasiadamente redundante e o custo da comunicação é alto.

2.1.4.2 Questões organizacionais

Uma vez listadas as diferentes necessidades relacionadas a uma rede de comunicação, a questão que se coloca é a da viabilidade de um projeto de rede, dada a quantidade de funções a implementar.

Uma outra questão é a do ordenamento das funções: o controle de fluxo deve ser realizado antes ou depois da correção de erros? Uma vez resolvida esta questão, que elementos da rede serão responsáveis da implementação destas funções? As soluções adotadas são dependentes do suporte de transmissão utilizado? Elas continuam válidas no caso de expansão da rede? Estas questões representam, de certo modo, a necessidade de levar em conta um certo ordenamento no que diz respeito à adoção das soluções a cada problema.

Uma ilustração típica do problema é aquele das relações internacionais: vamos supor dois países A e B, representados pelos seus respectivos presidentes que devem assinar um acordo de cooperação industrial e comercial. Supondo que a organização política dos dois países é a mesma, cada presidente deve convocar o seu primeiro ministro para acompanhar a execução do acordo.

Em cada país, o primeiro ministro vai convocar o ministro da indústria a fim de implementar o acordo do ponto de vista industrial.

Supondo que faz parte do acordo a construção de um novo avião civil, o ministro da indústria vai convocar o diretor das indústrias aeronáuticas e espaciais para que este faça os primeiros contatos. Finalmente, o diretor vai contactar um industrial do ramo e requisitar que este contacte seu homólogo no país B. Uma vez iniciado o processo de cooperação, os industriais deverão prestar informações sobre o estado da cooperação à administração dos seus respectivos países, sendo que a informação vai subindo na hierarquia da organização dos países, sendo filtrada em toda informação que possa parecer supérflua para o elemento superior.

Este processo é ilustrado na figura 2.1.6 e ele caracteriza, na verdade, a filosofia de concepção das redes de comunicação, que é baseada em dois conceitos fundamentais: o da *hierarquia* e o da *descentralização*, cuja conjunção vai permitir responder à questão de ordenação na adoção das soluções. Segundo esta filosofia, uma tarefa global é vista como sendo decomposta à medida que se vai descendo na hierarquia e que a única interação física se faz no seu nível mais baixo.

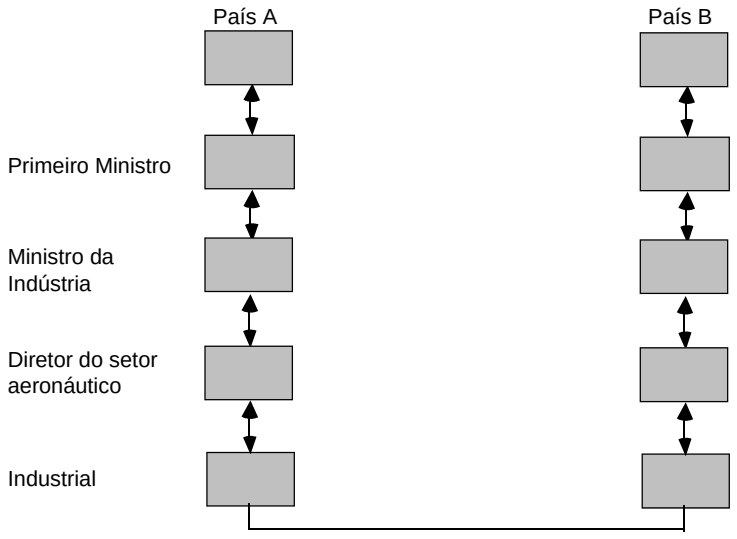


Figura 2.1.6 - A filosofia de concepção das redes, ilustrada por um processo de relações internacionais.

Podemos considerar que a comunicação entre dois nós de uma rede é uma tarefa global que afeta um sistema complexo e, conseqüentemente, sujeita à aplicação dos princípios de hierarquização e de descentralização.

As vantagens da adoção destes princípios são, fundamentalmente:

- facilidade de estudo e de implementação da rede a partir de elementos de base existentes, o que permite a redução dos custos de instalação;
- simplificação de sua operação em função da definição de regras formais;
- garantia de confiabilidade do sistema, particularmente graças ao encapsulamento das funções, o que permite limitar a propagação de erros e facilitar a manutenção;
- garantia, pela modularidade, de um grau satisfatório de evolutividade e de extensibilidade da rede;
- otimizar o desempenho.

Todos estes aspectos nos conduzem a conceber uma arquitetura de comunicação como sendo uma organização de software e hardware estruturada em camadas.

2.1.4.3 A estruturação em camadas

Os conceitos de hierarquia e descentralização podem ser empregados de diferentes formas, cada uma podendo implicar num tipo de rede particular. Em função desta provável multiplicidade, surgiu então a necessidade de uma normalização permitindo a conexão de diferentes classes de hardware.

Para possibilitar a normalização, foi necessário estabelecer um modelo teórico capaz de representar as relações entre as diferentes tarefas implementadas nos diferentes níveis hierárquicos. A possibilidade de interconexão de um número qualquer de sistemas, ou seja, de conjuntos autônomos podendo efetuar tarefas de tratamento ou de transmissão de informação, era uma característica essencial para o modelo a ser estabelecido.

A figura 2.1.7 ilustra a arquitetura hierarquizada em camadas (no caso, 7 camadas), que permitirá introduzir o conjunto de conceitos relacionados ao modelo estabelecido.

O objetivo de cada camada é o oferecimento de um determinado serviço às camadas superiores (utilizando-se, também dos serviços oferecidos pelas camadas inferiores) de forma a evitar que estas necessitem conhecer certos aspectos da implementação destes serviços.

A camada *n* assume a comunicação com a camada *n* de uma outra máquina. Para fazê-lo, ela se serve de um conjunto de convenções e regras que vão permitir gerir esta comunicação. A este conjunto de regras e convenções, dá-se o nome de *protocolo* da camada *n*, ou, simplesmente, *protocolo n*. As entidades representando camadas correspondentes em diferentes sistemas são denominadas *processos pares*, ou *entidades pares*. Os processos pares vão se comunicar então através dos protocolos, como foi visto na parte 1 deste documento.

Como se pode ver na figura, não existe meio de comunicação físico entre as diferentes camadas (apenas o Meio de Transmissão entre as entidades pares da camada 1), o que significa que não existe transferência direta de dados entre a camada *n* de uma máquina à camada *n* de outra máquina. Na realidade, cada camada transfere os dados à camada imediatamente inferior até a camada mais baixa; o dado é então transmitido à outra máquina através do Meio de Transmissão. A comunicação entre as camadas é vista então como uma comunicação *virtual* e é representada, na figura 2.1.7, pelas linhas ligando cada par de processos de uma camada.

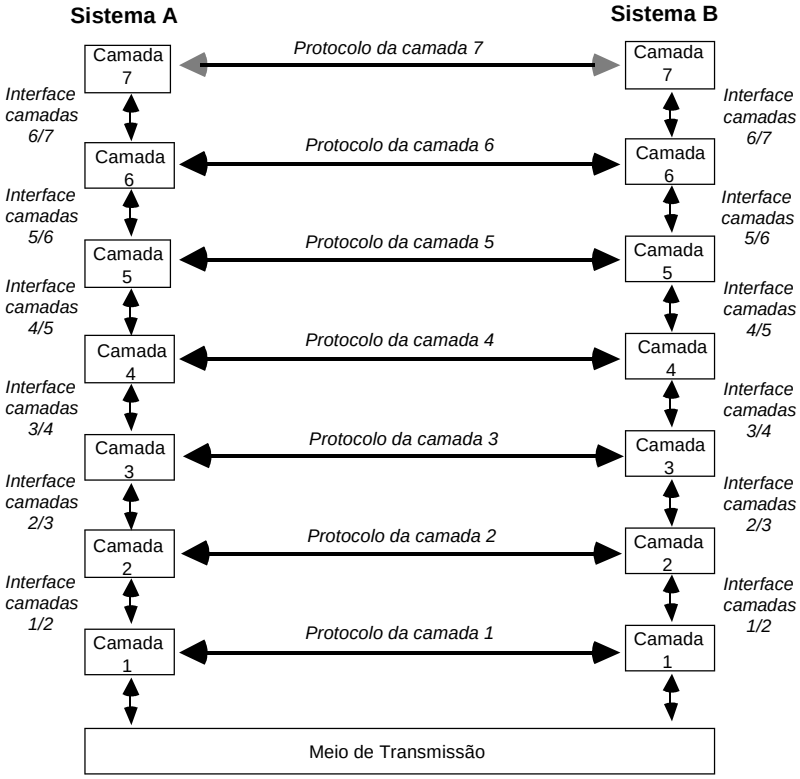


Figura 2.1.7 - Modelo hierarquizado em 7 camadas.

Cada camada comunica-se com as camadas adjacentes através de uma *interface*, que define as operações elementares e os serviços que a camada inferior oferece à camada considerada. No momento da definição do número de camadas que vai compor uma rede e do papel que cada uma delas deve cumprir, uma tarefa importante será a definição completa das

interfaces entre as camadas; isto vai implicar na definição do serviço oferecido por cada camada. Uma vantagem da correta definição das interfaces é a facilidade de introdução de modificações nas implementações das diferentes camadas; os mecanismos podem ser implementados de forma diferente, desde que as interfaces anteriormente definidas sejam respeitadas.

Ao conjunto das camadas compondo uma rede dá-se o nome de *arquitetura da rede*, e as especificações da arquitetura devem conter informações suficientes para permitir o correto desenvolvimento da rede, tanto do ponto de vista do software quanto do hardware. Por outro lado, os detalhes de implementação dos mecanismos a implementar em cada camada, assim como as especificações detalhadas das interfaces não fazem parte da definição da arquitetura da rede.

A figura 2.1.8 permite ilustrar o processo da comunicação no contexto de uma arquitetura multicamadas.

O processo da camada 7 gera uma mensagem *m*, que será transmitida desta à camada inferior segundo o que estiver definido pela interface das camadas 6/7.

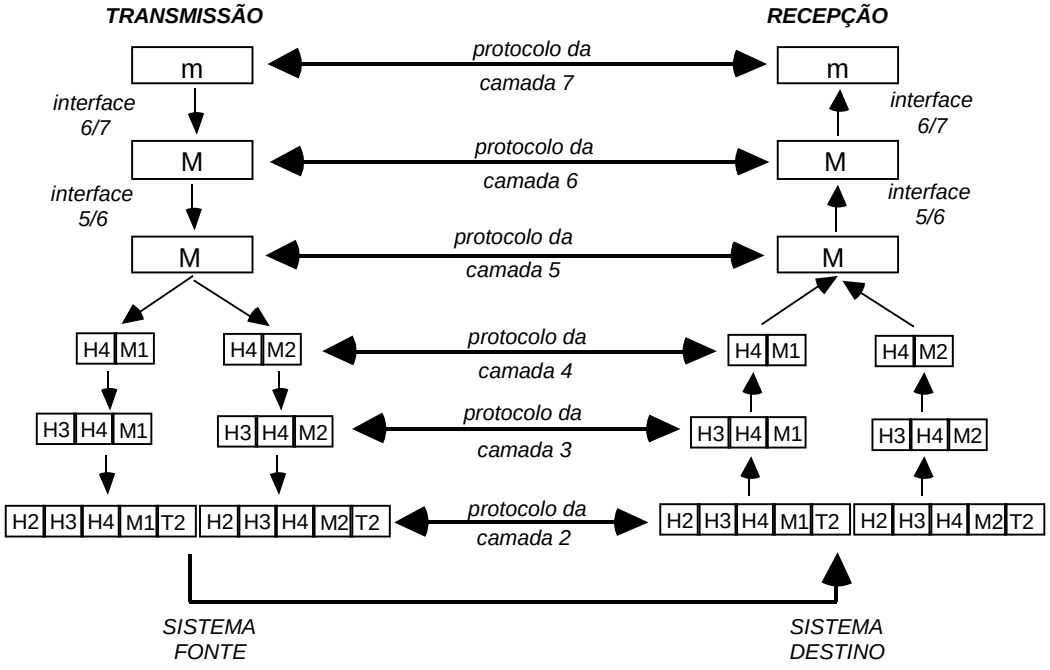


Figura 2.1.8 - Ilustração da comunicação virtual numa arquitetura de rede.

Considera-se que esta transmissão introduz algumas modificações na mensagem (por exemplo, uma compressão de dados), o que justifica uma nova representação desta por *M*. Esta mensagem é, por sua vez, transmitida à camada 5, através da interface das camadas 5/6. No exemplo considerado na figura, a mensagem não sofre modificações, mas esta camada efetua o controle de fluxo.

A camada 4 é responsável da decomposição da mensagem a fim de respeitar as restrições de tamanho que podem ser impostas pelas camadas inferiores. Assim, *M* é decomposta em *M1* e *M2*. Para isto, é inserido também na mensagem (ou nas partes da mensagem) um cabeçalho *H4* contendo uma informação de controle, como, por exemplo, um número de ordem que vai permitir, posteriormente, na camada 4 do sistema destinatário, a reconstrução da mensagem a partir das partes recebidas. Outras informações podem ainda estar contidas neste cabeçalho, como, por exemplo, o tamanho da mensagem, o instante de envio, etc.

Na camada 3, é feita a escolha das linhas de saída (roteamento) e um novo cabeçalho, *H3*, é introduzido às mensagens. Na camada 2, além de um cabeçalho, *H2*, é introduzido também um sufixo, *T2*, contendo informações específicas à esta camada. A mensagem é finalmente entregue à camada 1 para emissão via meio físico.

No sistema destinatário, o processo inverso se desenrola, sendo que as mensagens vão subindo, de camada em camada, e os cabeçalhos retirados nas camadas respectivas, de modo a evitar que estes sejam transferidos às camadas que não lhes dizem respeito.

Um aspecto importante mostrado nesta figura é o da comunicação *virtual* ocorrendo entre as diferentes camadas *pares*. As camadas em cada nível possuem uma visão da comunicação *horizontal*, mesmo se as mensagens são na realidade transmitidas às camadas inferiores pertencentes ao mesmo sistema.

2.2. O MODELO DE REFERÊNCIA OSI

2.2.1. INTRODUÇÃO

A grande importância da interconexão dos computadores através de redes de comunicação deu origem a uma necessidade que foi tornando-se evidente à medida que os desenvolvimentos neste domínio foram acentuando-se: a padronização das redes de comunicação.

Iniciou-se, então, no seio da ISO (International Standards Organization), uma reunião de esforços no sentido de definir uma proposta de arquitetura normalizada para as redes de comunicação. Dada a grande diversidade dos equipamentos e das soluções existentes no que diz respeito à comunicação, o resultado deste trabalho foi de fato a padronização de um modelo (denominado Modelo de Referência) sobre o qual deveriam ser baseadas as arquiteturas de redes de comunicação, de forma a permitir a interconexão de equipamentos heterogêneos, tornando transparente ao usuário a forma como esta interconexão fosse implementada. Um sistema fundamentado em tal modelo de referência é dito um *sistema aberto*, uma vez que este está aberto à comunicação com outros equipamentos, de diferentes classes, fabricantes, modelos, etc.

Baseada nesta filosofia, a proposta, definida numa série de documentos produzidos por aquela organização, foi denominada de *Modelo de Referência para a Interconexão de*

Sistemas Abertos ou *RM-OSI* (*Reference Model for Open Systems Interconnection*), cujos conceitos principais serão apresentados nas seções que seguem.

2.2.2. A ARQUITETURA OSI E AS FUNÇÕES DAS CAMADAS

O modelo OSI foi criado seguindo a filosofia das arquiteturas multicamadas, descrita no capítulo precedente. Como mostra a figura 2.2.1, sua arquitetura define 7 camadas, cujos princípios de definição foram os seguintes:

- cada camada corresponde a um nível de abstração necessário no modelo;
- cada camada possui suas funções próprias e bem definidas;
- as funções de cada camada foram escolhidas segundo a definição dos protocolos normalizados internacionalmente;
- a escolha das fronteiras entre cada camada deveriam ser definidas de modo a minimizar o fluxo de informação nas interfaces;
- o número de camadas deveria ser suficientemente grande para evitar a realização de funções muito diversas por uma mesma camada;
- o número de camadas deveria ser suficientemente pequeno para evitar uma alta complexidade da arquitetura.

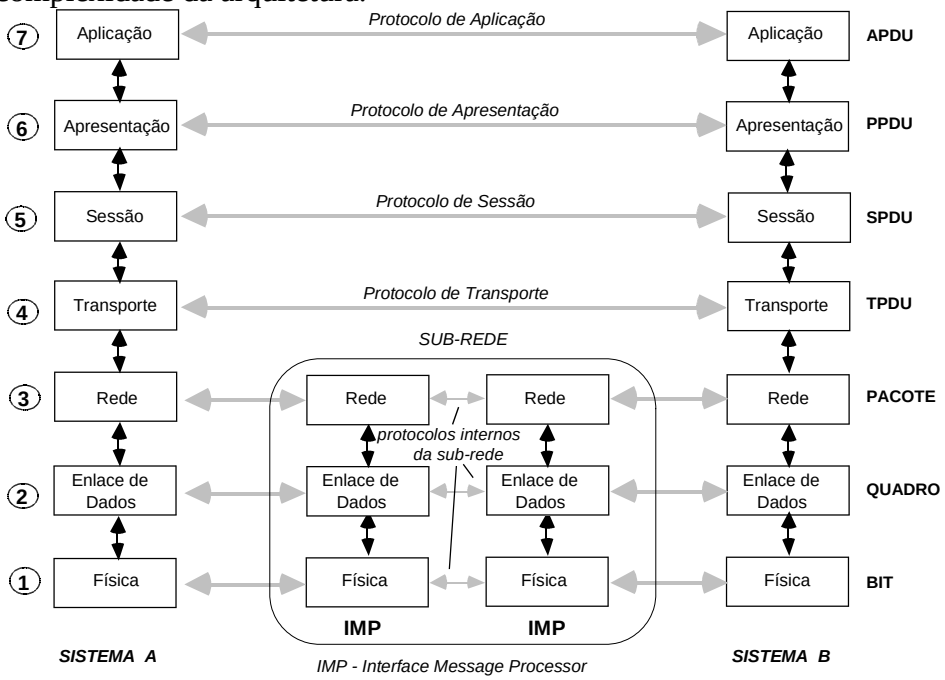


Figura 2.2.1 - Arquitetura a sete camadas do modelo OSI.

Como se pode ver na figura 2.2.1, o modelo prevê a comunicação entre subredes através de *IMPs* (*Interface Message Processors*). Na figura, vemos ainda as unidades de dados trocadas a nível da arquitetura: APDU (Application Protocol Data Unit), PPDU (Presentation Protocol Data Unit), ..., até BIT.

Vamos descrever, a seguir, as principais funções realizadas por cada uma das camadas definidas no modelo RM-OSI.

A *Camada Física* é responsável pela transferência de bits num circuito de comunicação. De maneira geral, a sua função é garantir que cada bit enviado de um lado será recebido do outro lado sem ter alterado o seu valor, ou seja, se o bit enviado está a 1, ele será recebido a 1 e não a 0. Para isto, as questões a serem resolvidas neste nível são do tipo:

- os modos de representação dos bits 0 e 1 de maneira a evitar ambigüidades ou confusões (valor da tensão em volts para a representação dos valores 0 e 1 dos bits, duração de cada sinal representando um bit, a codificação dos sinais, etc...);
- os tipos de conectores a serem utilizados nas ligações (número de pinos utilizado, as funções associadas a cada pino, ...);
- a maneira como as conexões são estabelecidas para a iniciação de um diálogo e como é feita a desconexão ao final deste;
- o modo de transmissão adotado (unidirecional, bidirecional, ...);
- o modo de conexão adotado (ponto-a-ponto, multiponto, ...);
- o modo de tratamento dos erros (detecção, tratamento, etc...).

A concepção desta camada deve se relacionar à definição das interfaces elétricas e mecânicas, seus modos de funcionamento, o suporte de comunicação adotado, etc.

A *Camada de Enlace de Dados* tem por função principal a transformação do meio de comunicação «bruto» em uma linha livre de erros de transmissão para a camada de Rede. Ela efetua esta função através da decomposição das mensagens em unidades de dados denominadas *quadros (Frames)*, que correspondem a algumas centenas de bytes. Estes quadros são transmitidos seqüencialmente e vão gerar quadros de reconhecimento enviados pelo receptor. Nesta camada, as unidades de dados são enriquecidas com um conjunto de bits adicionais (no início e fim de cada quadro) de modo a permitir o reconhecimento destes e a definição de um *endereço* para o destinatário da mensagem.

Um problema típico deste nível é o da ocorrência de uma perturbação sobre a linha de transmissão que provoque a deturpação ou destruição (perda) do quadro enviado. Esta deturpação ou perda deve ser reconhecida e tratada (*controle de erros de transmissão de quadros*). Neste caso, o quadro deve ser retransmitido para garantir a integridade da informação transferida. Por outro lado, deve-se também evitar múltiplas retransmissões de um mesmo quadro, o que pode provocar a sua duplicação, por exemplo, se o quadro de reconhecimento é perdido.

Uma outra função desta camada é evitar uma alta taxa de envio de dados da parte do emissor no caso do sistema receptor não ter capacidade de absorver a informação à mesma

taxa. Este mecanismo deve permitir informar ao emissor a necessidade de armazenamento dos dados a transmitir (*controle de fluxo de quadros*).

A *Camada de Rede* é responsável pela gestão de sub-redes; ela define a forma como os pacotes de dados serão encaminhados do emissor ao receptor (*roteamento*). Os caminhos a serem utilizados podem ser definidos em função de tabelas estáticas ou determinados dinamicamente no momento de cada diálogo em função das condições de tráfego da rede. Esta camada deve ainda efetuar a gestão dos problemas de congestionamento provocados pela presença de uma quantidade excessiva de pacotes de dados na rede. Ela deve, finalmente, resolver todos os problemas relacionados à interconexão de redes heterogêneas, particularmente:

- incompatibilidades no endereçamento;
- incoerências em relação aos tamanhos das mensagens;
- etc...

A *Camada de Transporte* representa uma interface entre as camadas orientadas à comunicação (1, 2 e 3) e as camadas orientadas à aplicação (5, 6 e 7). Ela recebe os dados enviados da camada de sessão, devendo decompô-los, se for o caso, em unidades de dados menores (*partição*) e garantir que todas as partes da mensagem vão ser transmitidas corretamente à outra extremidade. Esta função deve ser suprida de maneira eficiente, inclusive, sem que a camada de Sessão tome conhecimento de possíveis alterações na tecnologia da parte material da rede.

Esta camada cria, normalmente, uma conexão de rede para cada conexão de transporte requerida pela camada de Sessão, embora, se as necessidades de velocidade de transmissão são justificadas, ela possa estabelecer diversas conexões de rede para uma mesma conexão de transporte. Por outro lado, se o custo da manutenção de uma conexão de rede é considerado elevado, esta camada pode efetuar a função inversa, ou seja, a multiplexação de várias conexões de transporte sobre uma mesma conexão de rede, esta tarefa sendo feita de modo transparente para a camada de Sessão.

Ela deve determinar, também, o tipo de serviço oferecido à camada de Sessão e, conseqüentemente, aos usuários da rede. Uma conexão de transporte típica é aquela de um canal ponto-a-ponto, livre de erros de transmissão, transmitindo as mensagens na mesma ordem em que elas foram enviadas. Por outro lado, outras classes de serviços podem fornecer uma conexão capaz de enviar as mensagens de modo isolado, mas sem a garantia de uma ordem correta na transmissão. O tipo do serviço a ser fornecido é definido no momento do estabelecimento da conexão.

Uma característica desta camada é que ela implementa um verdadeiro diálogo *fim-a-fim*, ou seja, o programa executando no sistema fonte dialoga com o programa executando na

máquina destino através dos cabeçalhos e informações de controle contidas nas mensagens deste nível. Já nas camadas mais baixas, os protocolos operam entre máquinas vizinhas e não entre os sistemas fonte e destino, dado que estes podem estar separados por vários IMPs. Esta diferença fundamental, que se estende igualmente às camadas superiores (até a camada 7) pode ser verificada pela ilustração da figura 2.2.1.

Dado que esta camada é responsável do estabelecimento e término das conexões de rede, ela deve definir um mecanismo de endereçamento que permita a um sistema indicar com qual sistema ele deseja dialogar. Este endereçamento indica os processos de aplicação envolvidos no diálogo e não apenas os nós de rede envolvidos, como ocorre no endereçamento feito na camada de enlace de dados.

Finalmente, ela deve implementar um mecanismo de controle de fluxo fim-a-fim para evitar que o sistema fonte envie mensagens numa taxa superior àquela com a qual o sistema destino pode consumi-las.

A *Camada de Sessão* é responsável pelo estabelecimento de sessões de diálogo para os usuários da rede. Uma sessão objetiva permitir o transporte de dados, da mesma forma que os serviços oferecidos pela camada de Transporte, mas ela oferece serviços mais sofisticados de comunicação que podem ser úteis a determinadas aplicações. Um exemplo disto é a possibilidade de envio, através de uma sessão, de um arquivo de dados (ou programa) de um sistema a outro. Outro serviço da camada de Sessão é efetuar a gestão do diálogo, ou seja, definir, por exemplo, se o diálogo vai ser efetuado em modo uni- ou bi-direcional.

Um serviço também importante é aquele da sincronização do diálogo. Considere, por exemplo, que um arquivo deve ser transferido através de uma sessão e esta deve durar duas horas e que, por uma razão qualquer, o tempo médio entre duas panes é de uma hora. Após uma primeira interrupção por pane, a transferência deverá reiniciar, podendo ocasionar erros de transmissão. Uma forma de evitar isto é a inserção de pontos de teste junto aos dados fazendo com que, após uma interrupção de transferência, os dados sejam retomados apenas a partir do último ponto de teste.

A *Camada de Apresentação* oferece algumas funções freqüentemente necessárias na comunicação, de modo a poupar o usuário deste trabalho. Esta camada assume particularmente as funções associadas à formatação, sintaxe e semântica dos dados transmitidos. Um exemplo típico das funções efetuadas por esta camada é a codificação da informação num padrão bem definido (ASCII, EBCDIC, etc).

Esta camada pode ainda suprir outras funções associadas à compressão dos dados, se utilizando do conhecimento do significado da informação para reduzir a quantidade de informação enviada, inclusive para implementar funções de confidencialidade e de autenticação (proteção de acesso).

A *Camada de Aplicação* implementa um conjunto de protocolos bastante diversificado e orientado a aplicações bem definidas. Um exemplo disto é o *protocolo de terminal virtual*, que permite a utilização de um determinado programa (por exemplo, um editor de textos) de forma independente do tipo de terminal conectado à rede. Outro serviço importante é o de transferência de arquivos, que permite adaptar o tipo do arquivo transferido à forma implementada pelo sistema de arquivamento do sistema considerado.

Na parte dedicada a esta camada veremos, além destas, outras classes de serviços implementados a este nível.

2.2.3. A COMUNICAÇÃO NO RM-OSI

A forma como os dados são transferidos ao longo do modelo OSI é ilustrada na figura 2.2.2. Como se pode ver, o processo emissor vai enviar uma certa quantidade de dados ao processo receptor. Ele envia, então, os dados à camada de Aplicação que introduz a estes um cabeçalho de aplicação, *AH*, e envia a mensagem resultante à camada de Apresentação.

Esta camada, por sua vez, introduz à mensagem recebida um cabeçalho de apresentação, *PH*, enviando a mensagem em seguida à camada inferior.

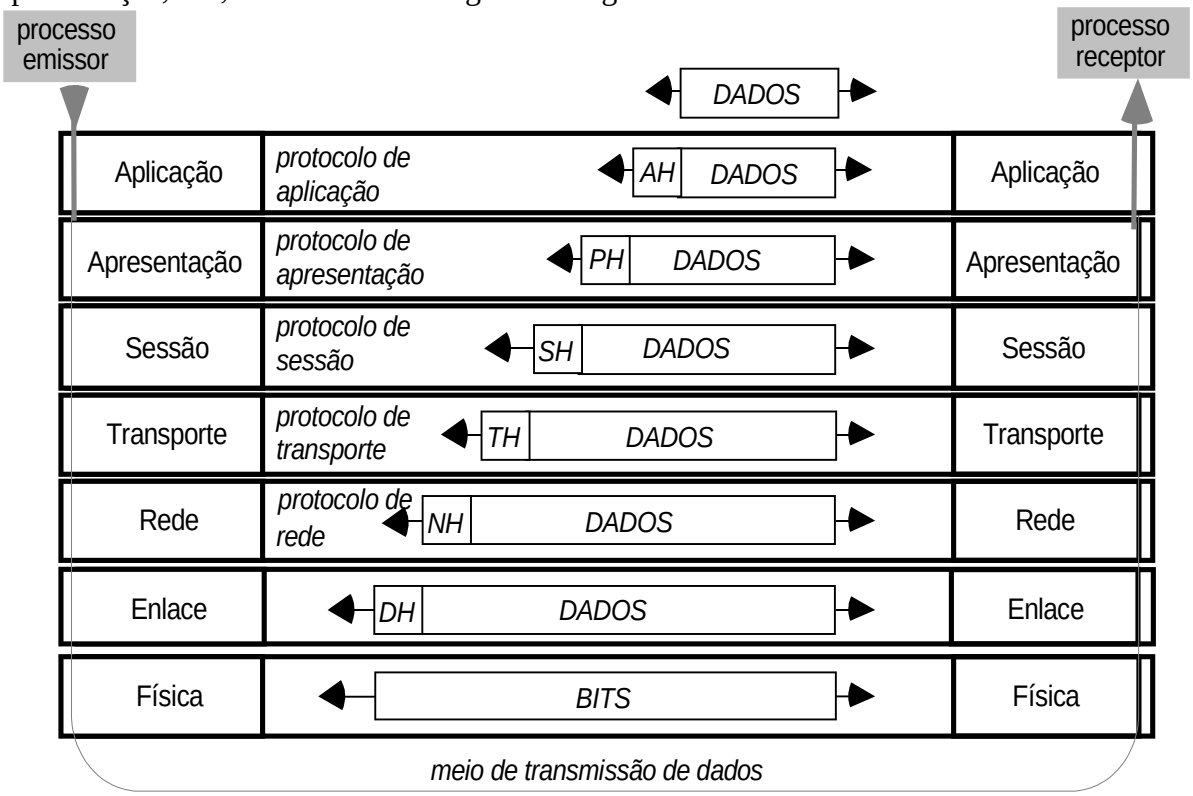


Figura 2.2.2 - Ilustração da comunicação no modelo OSI.

É importante ressaltar aqui que esta camada não toma conhecimento da existência e significado do cabeçalho de aplicação, considerando este como parte dos dados compondo a mensagem. Este processo de transferência de camada a camada vai se repetindo até o nível físico, quando os dados serão, enfim, transmitidos ao sistema destino. Neste sistema, os diversos cabeçalhos introduzidos nas camadas de rede do sistema fonte vão sendo interpretados e eliminados nas camadas correspondentes até que os dados cheguem ao processo receptor.

O conceito fundamental da transferência de dados é que cada camada foi projetada como se ela fosse realmente horizontal, quando na verdade a transmissão se dá de modo vertical.

Isto fica claro, por exemplo, quando a camada de Transporte emissora recebe um dado da camada de Sessão; ela insere um cabeçalho de transporte e envia a mensagem à camada de Rede emissora. Este processo, portanto, para a camada de Transporte, não é mais do que um detalhe técnico. Um exemplo análogo é aquele de um diplomata de um país fazendo um discurso, na sua própria língua, nas Nações Unidas. Este considera estar se dirigindo aos seus colegas diplomatas de outros países (suas entidades pares), embora, na prática, ele esteja dirigindo-se ao seu intérprete (a camada logo abaixo na hierarquia).

2.2.4. OS CONCEITOS DO MODELO RM-OSI

Como visto anteriormente, o objetivo de cada camada definida no modelo OSI é fornecer um determinado conjunto de serviços à camada imediatamente superior. A nível do modelo de referência OSI, foi feita uma série de definições que vão permitir identificar cada componente do modelo de forma clara e não ambígua. É objetivo desta seção introduzir estes conceitos.

2.2.4.1 Terminologia OSI

A nível de cada camada existem elementos ativos que implementam os serviços e protocolos relacionados com aquela camada. A estes elementos ativos, dá-se o nome de *entidades*, que podem ser entidades de software ou de hardware. Às entidades localizadas em diferentes sistemas, mas associadas a um mesmo nível (ou camada), dá-se o nome de *entidades pares*. As entidades recebem também uma denominação complementar em função da camada à qual elas estão relacionadas — por exemplo, entidade de aplicação, entidade de apresentação, etc.

As entidades de uma camada *N* (ou *entidades N*) implementam um serviço que é utilizado pela camada *N+1*. Assim, a camada *N* é dita ser um *fornecedor de serviço* e a

camada *N+1* é denominada um *usuário de serviço*. Por outro lado, a camada *N* poderá utilizar os serviços da camada imediatamente inferior, a camada *N-1* para oferecer os serviços à camada superior. Ela pode ainda oferecer diferentes categorias (ou classes) de serviços: serviços mais eficientes e mais «caros» ou serviços menos eficientes e «econômicos».

Os serviços oferecidos por uma camada são acessíveis em *pontos de acesso aos serviços*, ou *SAP* (*service access point*). Os SAPs da camada *N* são os lugares onde a camada *N+1* poderá acessar os serviços oferecidos, cada SAP sendo identificado por um endereço único. Por exemplo, os SAP de uma rede telefônica são as tomadas às quais podem ser conectados os aparelhos telefônicos e seus endereços são os números de telefone associados à tomada considerada.

Para que duas camadas possam trocar informações, existe uma série de regras a serem respeitadas, definidas pela *interface*. Através de uma interface, a camada *N+1* envia uma *unidade de dados de interface*, ou *IDU* (*Interface Data Unit*) à entidade da camada *N* pelo *SAP*. A IDU é composta de uma parte denominada *unidade de dados de serviço*, ou *SDU* (*Service Data Unit*) e de *informações de controle de interface,* ou ICI (Interface Control Information). A SDU é a informação transmitida via rede à entidade par e, em seguida, à camada *N+1*. A ICI é utilizada para auxiliar a gestão da camada inferior em seu trabalho (por exemplo, o número de bytes compondo a SDU correspondente).

Para transmitir uma SDU, a entidade da camada *N* pode fragmentá-la em diversas partes, e cada parte vai receber um cabeçalho, sendo enviada como uma *unidade de dados de protocolo*, ou *PDU* (*Protocol Data Unit*). Os cabeçalhos de PDU são utilizados pelas entidades pares para o transporte do protocolo. Elas identificam a PDU contendo os dados e aquelas contendo informações de controle (números de seqüência, contagens, etc). A figura 2.2.3 ilustra o processo descrito. As PDUs recebem normalmente uma denominação segundo a camada à qual estão associadas. Por exemplo, as PDUs de aplicação são ditas APDU, assim como as de apresentação são as PPDU, as de sessão SPDU, e assim por diante.

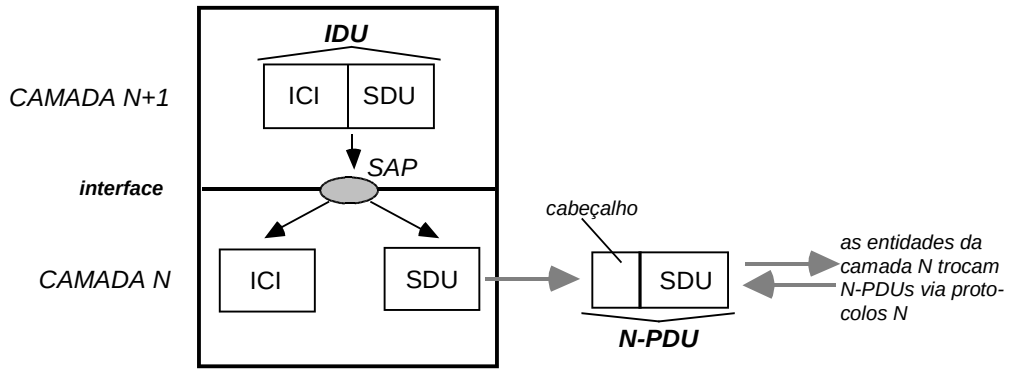


Figura 2.2.3 - Conceitos associados ao modelo de referência OSI.

2.2.4.2 Serviços orientados à conexão e sem conexão

Como já foi dito, as camadas podem oferecer diferentes classes de serviços às camadas superiores. Estes serviços podem ser orientados à conexão ou sem conexão.

No que diz respeito aos *serviços orientados à conexão*, podemos citar, como exemplo típico o sistema telefônico. Para que seja possível falar a alguém no telefone, é necessário, inicialmente, tirar o fone do gancho, digitar (ou discar) um número, esperar que o interlocutor atenda, falar e, finalmente, desligar. Este é o princípio de base de um serviço orientado à conexão: estabelecimento da conexão, utilização do serviço e término da conexão. O aspecto principal da conexão é o fato de que ela funciona como uma espécie de canal através do qual irão transitar os objetos ou mensagens envolvidas na realização do serviço.

Já os serviços sem conexão são estruturados como o sistema postal, onde cada mensagem (ou carta, se consideramos o exemplo citado) contém o endereço do destinatário e é então encaminhada pelo sistema, independente de outras mensagens. Normalmente, se duas mensagens são enviadas a um mesmo destinatário, a primeira a ser enviada deve ser a primeira a ser recebida. Por outro lado, neste modo de serviço pode ocorrer que uma mensagem seja atrasada fazendo com que a segunda mensagem seja recebida primeiro. Já nos serviços orientados à conexão, isto jamais poderá acontecer.

Cada serviço é caracterizado por uma qualidade de serviço. Um serviço dito *confiável* é aquele em que os dados nunca podem ser perdidos. Normalmente, a implementação de serviços confiáveis é feita através da definição de mensagens de reconhecimento enviadas pelo receptor, para cada mensagem recebida do emissor. Este processo, embora extremamente benéfico, introduz uma lentidão na transferência de dados, o que significa que nem sempre ele é desejável num sistema.

Nem todas as aplicações requerem a utilização de conexões. Um exemplo disto pode ser o de uma aplicação de correio eletrônico. Pode-se imaginar uma aplicação de correio em que o usuário não se interesse pelo estabelecimento de uma conexão e tampouco por uma confiabilidade de 100% no que diz respeito à chegada das mensagens. Os serviços sem conexão e não-confiáveis são denominados serviços de *datagrama.*

Existem casos, porém, em que, apesar de não necessitar o estabelecimento de conexão, a confiabilidade é essencial. O serviço utilizado neste caso é dito de *datagrama com reconhecimento*.

O serviço de *pedido-resposta* já é um outro tipo de serviço no qual o emissor envia um datagrama contendo um serviço e o receptor envia um outro contendo a resposta a este pedido.

A tabela a seguir ilustra os diferentes serviços com e sem conexão, com exemplos de aplicação destes serviços.

Serviços	***Comentário***	***Exemplo***
Transf. confiável de mensagens	*com ou sem conexão*	*sequenciamento de páginas*

Transf. confiável de dados	*com ou sem conexão*	*transferência de arquivos*
Transf. sem controle de erros	*com ou sem conexão*	*voz digitalizada*
Datagrama s/ reconhecimento	*sem conexão*	*correio eletrônico*
Datagrama c/ reconhecimento	*sem conexão*	*correio eletrônico «registrado»*
Pedido-Resposta	*sem conexão*	*consulta a bases de dados*

2.2.4.3 As Primitivas de Serviço

Um serviço é definido formalmente por um conjunto de primitivas (ou operações) disponíveis a um usuário ou a outras entidades para o acesso àquele serviço. Estas primitivas permitem indicar a ação a ser executada pelo serviço ou ainda um pedido de informação sobre uma ação executada previamente.

No modelo OSI, as primitivas de serviço são divididas em quatro classes: as primitivas de pedido (*request*), as primitivas de indicação (*indication*), as primitivas de resposta (*response*) e as primitivas de confirmação (*confirm*).

A tabela a seguir mostra o significado de cada uma destas primitivas no que diz respeito à execução de um serviço.

Primitiva	**Significado**
REQUEST	*Pedido enviado por uma entidade que solicita um serviço*
INDICATION	*Através dela, a entidade par é informada de uma solicitação de serviço*
RESPONSE	*A entidade par responde ao pedido de serviço*
CONFIRM	*A entidade solicitante é informada do resultado do serviço*

Um exemplo da utilização das primitivas de serviço é o pedido de um estabelecimento de conexão. Para requisitar o estabelecimento de uma conexão, a entidade que quer iniciar o diálogo envia uma primitiva de serviço de pedido de abertura de conexão, *«CONNECT.request»,* que vai se refletir, na entidade destinatária, por uma primitiva de indicação, *«CONNECT.indication»*. A entidade que recebeu a indicação vai enviar uma primitiva de resposta, *«CONNECT.response»*, para informar se esta aceita ou não a conexão. Finalmente, a entidade emissora vai saber do resultado do seu pedido pela recepção de uma primitiva de serviço de confirmação, *«CONNECT.confirm»*. Este procedimento é ilustrado na figura 2.2.4.

Figura 2.2.4 - Ilustração da troca de primitivas de serviço (confirmado).

Parâmetros podem ser associados às primitivas. No caso do serviço de conexão, por exemplo, os parâmetros podem especificar os seguintes aspectos relacionados à conexão desejada:

- a máquina com a qual se deseja dialogar;
- o tipo de serviço desejado;
- o tamanho máximo das mensagens;
- etc...

Se a entidade invocada não está de acordo com os parâmetros contidos na primitiva de indicação recebida, esta pode fazer uma contraproposta, através dos parâmetros da primitiva de resposta, que será transmitida à entidade emissora através dos parâmetros da primitiva de confirmação.

Os serviços no modelo OSI podem ser de dois tipos: confirmados ou não-confirmados.

No caso dos *serviços confirmados*, as quatro classes de primitivas são definidas, ou seja, pedido (*request*), indicação (*indication*), resposta (*response*) e confirmação (*confirm*). Isto significa que a entidade que requisitou o serviço terá sempre uma informação sobre as condições de realização deste até mesmo se este foi realizado com sucesso ou não.

Nos serviços *não-confirmados*, apenas as duas primeiras classes de primitivas são utilizadas, ou seja, pedido e indicação. Neste tipo de serviços, a entidade emissora do pedido não receberá nenhuma informação sobre as condições de realização do serviço requisitado, nem mesmo se este foi realizado.

A tabela abaixo apresenta um conjunto de primitivas associadas a um serviço orientado à conexão. Neste exemplo, *CONNECT* é um serviço confirmado enquanto os serviços *DATA* e *DISCONNECT* são não-confirmados.

Primitiva	**Significado**
CONNECT.request	*pedido de estabelecimento de uma conexão*
CONNECT.indication	*indicação à entidade invocada*
CONNECT.response	*utilizada para indicar a aceitação ou não da conexão*
CONNECT.confirm	*informa à entidade emissora se a conexão é aceita*

DATA.request	*pedido de envio de dados*
DATA.indication	*sinalização da chegada de dados*
DISCONNECT.request	*pedido de término da conexão*
DISCONNECT.indication	*indicação do pedido à entidade par*

2.2.4.4 A Relação entre Serviço e Protocolo

Embora sejam freqüentemente confundidos, serviço e protocolo são dois conceitos distintos. O importante nesta distinção é de poder estabelecer a relação entre os dois conceitos.

O *serviço* corresponde a um conjunto de operações que uma camada é capaz de oferecer à camada imediatamente superior. Ele define *o que* uma camada é capaz de executar sem se preocupar com a maneira pela qual as operações serão executadas. O serviço está intimamente relacionado com as interfaces entre duas camadas, a inferior sendo a fornecedora do serviço e a superior, a usuária deste.

Por outro lado, o *protocolo* define um conjunto de regras que permitem especificar aspectos da realização do serviço, particularmente, o significado dos quadros, pacotes ou mensagens trocadas entre as entidades pares de uma dada camada. A nível de uma camada, o protocolo pode ser mudado sem problemas, desde que as interfaces com a camada superior não sejam alteradas, ou seja, que aquela continue a ter a mesma visibilidade no que diz respeito aos serviços realizados pela camada considerada.

2.3. SERVIÇOS E PROTOCOLOS OSI

2.3.1. INTRODUÇÃO

A definição de um modelo de referência para a interconexão de sistemas abertos, o modelo RM-OSI foi, sem dúvida, uma contribuição positiva para a padronização das arquiteturas de comunicação.

Entretanto, a definição das camadas presentes do modelo OSI está relacionada à definição de soluções em termos de serviços e protocolos que implementem as funções relacionadas a cada uma destas camadas.

O objetivo desta parte do documento é apresentar os principais problemas e as respectivas soluções relacionadas com cada uma das camadas do modelo OSI.

Como poderá ser visto ao longo desta parte, as sete camadas do modelo OSI podem ser organizadas em duas classes distintas:

- as camadas baixas, que compreendem desde a camada física até a camada de transporte e cujos serviços e protocolos estão relacionados com a transmissão dos dados propriamente dita;

- as camadas altas, que compreendem desde a camada de sessão até a camada de aplicação, cujos serviços e protocolos são mais orientados a resolver questões envolvendo as aplicações que irão utilizar o suporte de comunicação considerado.

2.3.2. A CAMADA FÍSICA

O objetivo da camada Física é assegurar o transporte dos dados representados por um conjunto de bits entre dois equipamentos terminais, via um suporte de transmissão.

Abordaremos, nesta parte do documento, os principais aspectos e problemas relacionados à transmissão de dados, como, os suportes de transmissão, os modos de transmissão, a multiplexação e a comutação.

Ainda nesta seção, serão vistos alguns exemplos de interfaces físicas padronizadas e adotadas em muitas aplicações e arquiteturas de comunicação.

2.3.2.1. Os Suportes de Transmissão

Os suportes de transmissão podem se caracterizar pela existência ou não de um guia físico para o envio do sinal. Na primeira classe estão os cabos elétricos, as fibras óticas e, na segunda classe, as ondas de rádio, as ondas de luz, etc.

a) Transmissão com guia físico

- O par de fios trançados (twisted pair)

Em diversas aplicações, é necessário se manter uma conexão direta e permanente entre dois computadores. O suporte de transmissão mais clássico utilizado até o momento é o par de fios trançados, o qual é composto de dois fios elétricos em cobre, isolados, e arranjados longitudinalmente de forma helicoidal. Esta técnica de enrolar os fios permite diminuir os efeitos das induções eletromagnéticas parasitas provenientes do ambiente no qual este estiver instalado.

A utilização mais típica deste suporte de transmissão é a rede telefônica, onde, graças às suas características elétricas, os sinais podem percorrer várias dezenas de quilômetros, sem a necessidade de amplificação ou regeneração de sinal.

Estes podem, ainda, ser utilizados tanto para a transmissão de sinais analógicos quanto de sinais digitais, a banda passante atingida sendo função da sua composição (particularmente, diâmetro e pureza dos condutores, natureza dos isolantes e do comprimento do cabo). A taxa de transmissão obtida pela utilização deste suporte de transmissão situa-se na faixa de

algumas dezenas de Kbits/s, podendo atingir, em condições particulares, a faixa dos Mbits/s para pequenas distâncias.

O fato de representar um baixo custo e uma grande faixa de utilização o tornam um dos suportes mais utilizados atualmente e, provavelmente, nos próximos anos.

- *Os cabos coaxiais*

Os cabos coaxiais são também altamente empregados como suporte de transmissão. Dois tipos de cabos são tipicamente utilizados: o primeiro tipo apresenta uma impedância característica de 50 ohms, utilizado nas transmissões digitais denominada *transmissão em banda base*; o segundo tipo, com uma impedância característica de 75 ohms, é mais adequado para a transmissão de sinais analógicos. Eles são constituídos de dois condutores arranjados de forma concêntrica: um condutor central, a *alma*, envolto por um material isolante de forma cilíndrica. Esta capa isolante é, por sua vez, envolta por uma trança metálica condutora em cobre. Finalmente, o conjunto é envolto numa capa de proteção em plástico isolante, como mostrado na figura 2.3.1.

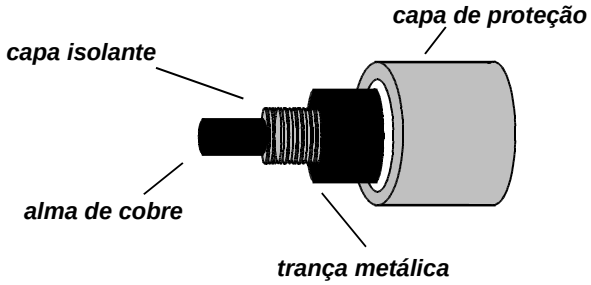


Figura 2.3.1 - Cabo coaxial.

Em relação aos pares de fios trançados, os cabos coaxiais apresentam melhores características elétricas, oferecendo um boa relação entre a banda passante e a proteção contra interferências eletromagnéticas. Seu custo, porém, é bastante superior.

A largura de banda vai depender igualmente da qualidade da composição do cabo e do seu comprimento. Para distâncias em torno de 1 km, é possível obter uma taxa de transmissão em torno de 10 Mbits/s, podendo-se obter taxas superiores para distâncias mais curtas. Os cabos coaxiais são muito utilizados como suporte de transmissão nas Redes Locais Industriais.

- *As fibras óticas*

As fibras óticas são o meio de transmissão pelo qual os sinais binários são conduzidos sob a forma de impulsos luminosos. Um impulso luminoso representa um bit a 1, enquanto a ausência deste impulso representa um bit a 0. A luz visível é uma onda luminosa cuja freqüência está na ordem de 10^8 Hz, o que dá ao sistema uma banda passante potencial bastante grande. As taxas de transmissão num suporte a fibra ótica ficam na faixa dos Gbit/s (10^9 bit/s).

Um sistema de transmissão a base de fibra ótica é composto de três elementos principais: o suporte de transmissão (a fibra ótica propriamente dita), o dispositivo de emissão e o dispositivo de recepção da onda luminosa.

A fibra ótica é constituída de um filamento bastante fino, à base de silício e outros componentes dopantes. Ela consiste de um núcleo no qual se propaga a luz e uma capa externa de proteção que mantém a luz no interior do núcleo. O dispositivo de emissão consiste de um diôdo emissor de luz (*LED*) ou de um diôdo laser. O dispositivo de recepção é constituído geralmente de um fotodiôdo ou de um fototransistor (figura 2.3.2).

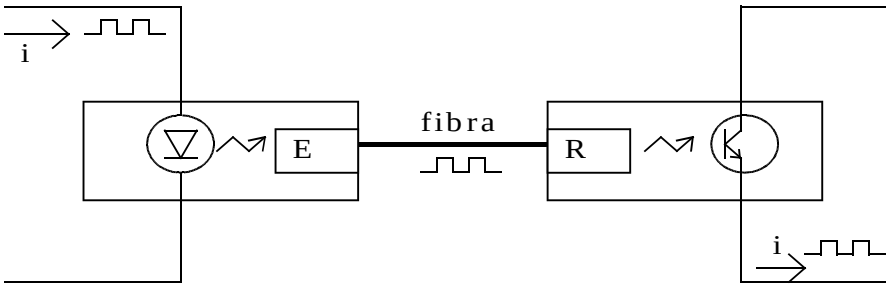


Figura 2.3.2 - Transmissão ótica de sinais

O princípio da transmissão das fibras óticas é o da reflexão da luz na interface entre dois meios. Quando um raio luminoso deixa um meio homogêneo para se propagar num outro meio, o seu percurso sofre um desvio na interface entre os dois meios (refração). Entretanto, como é mostrado na figura 2.3.3, existe um ângulo de incidência limite, a partir do qual o raio luminoso, ao invés de ser refratado, será refletido na interface, sendo mantido no meio no qual ele havia sido introduzido. Desta forma, a luz poderá ser propagada ao longo do meio, em distâncias de vários quilômetros.

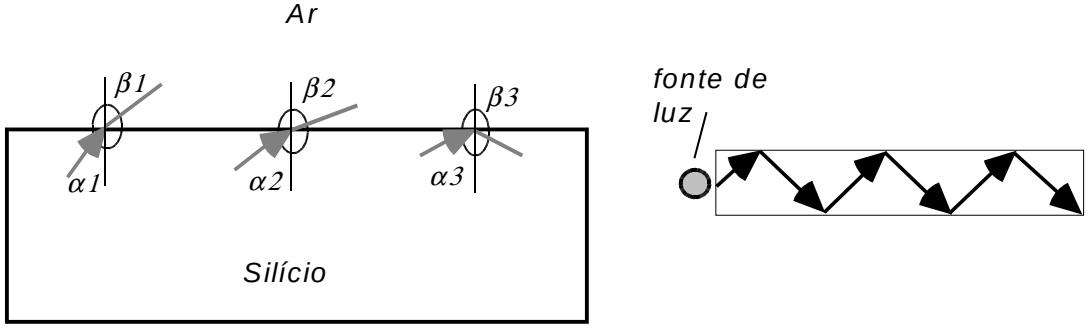


Figura 2.3.3 - Princípio da transmissão em fibras óticas.

Atualmente, os suportes de comunicação à base de fibra ótica são utilizados em redes de comunicação em longa distância, substituindo sistemas mais antigos à base de cabos coaxiais. Isto deverá continuar a ocorrer nos próximos anos, contribuindo para que se tenha, num futuro próximo, em distâncias relativamente grandes, sistemas de comunicação oferecendo altas taxas de transmissão, garantindo assim o envio de grandes volumes de informação.

Em redes locais industriais também se vêm procurando adotar a tecnologia das fibras óticas nos últimos anos. Um problema, porém, associado às características das redes locais é o da definição dos nós de rede, particularmente, no que diz respeito às derivações, uma vez que estas freqüentemente têm topologia em barramento. Uma alternativa é a construção de redes em anel, constituídas de ligações ponto-a-ponto.

A realização das derivações em fibra ótica pode se basear em duas diferentes técnicas. A primeira consiste da definição de uma derivação *passiva*, composta de uma bifurcação simples do filamento ótico. O inconveniente desta técnica é a perda de intensidade do sinal luminoso em cada bifurcação, o que limita o número máximo de nós a serem implementados.

A segunda técnica baseia-se no *repetidor ativo*, onde a luz proveniente do suporte a base de fibra ótica é convertida novamente em sinal elétrico em cada derivação. Estes são transmitidos ao computador ou equipamento considerado e, em seguida, reconvertidos em sinal luminoso para a continuidade da transmissão ao longo da rede. Uma desvantagem desta técnica é que, no caso de pane de um destes repetidores, o funcionamento da rede como um todo também será comprometido. Outro inconveniente é o aumento do custo de realização das bifurcações, que passam a exigir um pequeno circuito eletrônico em cada repetidor. Por outro lado, uma vez que os sinais são reconstituídos em cada repetidor, a distância entre cada nó da rede pode ser grande, assim como o número de pontos da rede (sem limite teórico), constituindo assim uma vantagem sobre os outros sistemas apresentados.

b) A transmissão de dados sem guia físico

Como visto nas seções anteriores, a transmissão de dados utilizando guias físicos é altamente dependente da tecnologia de concepção do suporte de comunicação considerado. Um outro problema ligado à utilização de guias físicos é a necessidade, dependendo da aplicação e da extensão da rede, da instalação de sistemas de canalização para passagem do suporte de transmissão. Isto pode elevar consideravelmente o custo de uma rede, em certas situações.

Nestes casos, uma rede de comunicação baseada em um meio sem guia físico pode ser uma solução mais interessante. Por exemplo, numa rede que deve estender-se ao longo dos

diversos prédios num campus de uma universidade, pode ser mais interessante a instalação de um conjunto emissor-receptor a raios laser ou infravermelhos no teto de cada um dos prédios.

Já existem também redes locais baseadas em sinais de rádio. Cada estação de rede tem que possuir uma unidade de transmissão e recepção (transceptor) de rádio, podendo ser livremente deslocada dentro do ambiente. Dentre as vantagens deste tipo de sistema, podemos citar:

- Flexibilidade
- Interconexão completa
- Estações móveis

Por outro lado, uma rede via rádio apresenta também as seguintes desvantagens:

- Problema de autenticação
- Privacidade
- Dependência de regulamentação pública

Deve-se também levar em consideração as seguintes limitações e compromissos:

- banda passante;
- área de cobertura;
- interferências;
- regulamentações;
- custos.

No caso de comunicação em longas distâncias, as ondas de rádio em alta freqüência podem também ser utilizadas, constituindo uma boa alternativa à utilização dos cabos coaxiais ou fibras óticas. Neste caso, antenas parabólicas de emissão-recepção são instaladas no alto de pilares de concreto e uma faixa de freqüência é então estabelecida entre duas antenas situadas numa distância da ordem de dezenas de quilômetros.

Os satélites de comunicação são uma outra opção para a transmissão de dados em uma larga gama de aplicações. Estes podem ser vistos como estações repetidoras instaladas no céu dotadas de um certo número de emissores-receptores, capazes de receber sinais numa certa faixa de freqüência, amplificá-los e, em seguida, retransmití-los em outra faixa de freqüência.

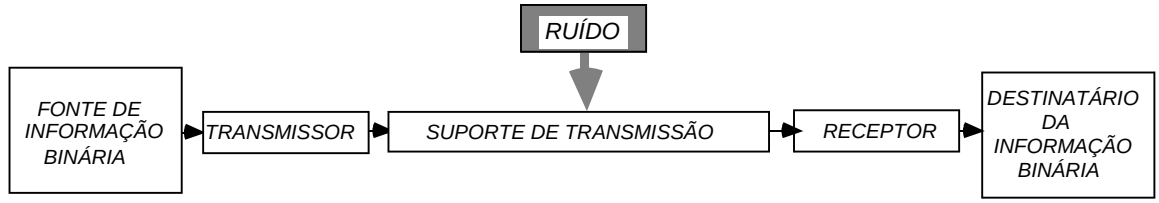
Dependendo do satélite, os feixes de onda difundidos para a terra podem cobrir uma zona geográfica relativamente grande ou mais restrita. Um satélite utiliza, em geral, uma faixa de freqüência em torno de 500 MHz distribuída entre 10 a 12 repetidores, cada um deles utilizando uma faixa em torno de 36 MHz, dentro da qual este pode transmitir um fluxo de dados a uma taxa de 50 Mbit/s.

2.3.2.2. Aspectos da Transmissão de Dados

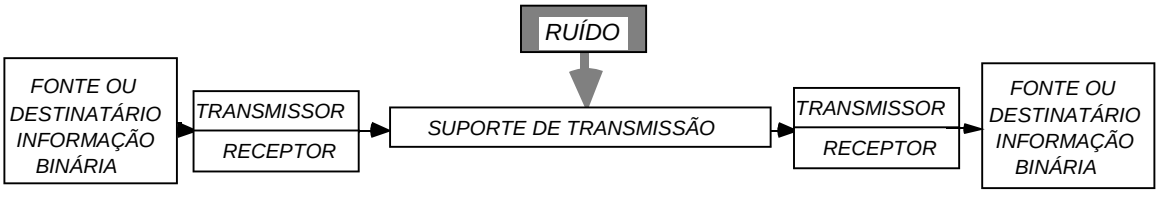
O meio de transmissão consiste geralmente de um conjunto de recursos e regras que permitem a transmissão de informações de um ponto a outro numa rede de comunicação. Este processo é ilustrado pela figura 2.3.4(a), onde podemos observar os seguintes elementos:

- a *fonte de informação*, que pode ser um computador ou um terminal, por exemplo, que gera as informações que deverão ser transmitidas, estas sendo representadas, usualmente, por um conjunto de dígitos binários, ou *bits*;
- o *transmissor*, que é responsável da adaptação ou conversão do conjunto de informações, de bits, para sinal elétrico ou eletromagnético, adaptando-o ao meio de transmissão;
- o *suporte de transmissão*, encarregado do transporte dos sinais representando a informação e que pode ser caracterizado por uma das técnicas apresentadas na seção precedente; é o suporte de transmissão quem realiza a "ligação física" entre os elementos envolvidos na comunicação;
- o *receptor*, responsável pela reconstituição da informação a partir dos sinais recebidos via suporte de transmissão, e que, inclusive pode ter sofrido distorções provocadas por ruídos existentes no meio;
- o *destinatário da informação*, que pode ser um computador, um terminal ou outro equipamento e que vai consumir a informação gerada pelo elemento fonte.

Geralmente, a transmissão pode ser realizada de forma bidirecional, de forma alternada ou simultânea. Assim, a cada nó deverá estar associado um equipamento transmissor e um receptor compondo o conjunto *transceptor* como mostrado na figura 2.3.4(b).



(a)



(b)

Figura 2.3.4 - (a) Sistema de transmissão ponto-a-ponto unidirecional; (b) modelo bidirecional.

A transmissão de dados em um único sentido é denominada *simplex*, e quando realizada nos dois sentidos é denominada *duplex*. No caso em que ela se realiza alternadamente, ou seja, ora num sentido, ora no outro, ela se denomina *half-duplex*. No caso em que ela se realiza simultaneamente nos dois sentidos, esta será denominada *full-duplex*.

Nas seções seguintes serão discutidos diversos aspectos básicos referentes à transmissão de dados.

a) Modos de transmissão

Os modos de transmissão caracterizam as diferentes formas como os bits de informação transmitidos são delimitados e encaminhados ao longo da linha de comunicação.

No que diz respeito à forma como os bits são encaminhados ao longo de uma linha de comunicação, pode-se distinguir o modo de transmissão *paralelo* e o modo *serial*.

Na *transmissão paralela*, os bits são transportados simultaneamente por um suporte composto de várias linhas em paralelo. É um modo de transmissão mais adequado à comunicação entre equipamentos localizados a curtas distâncias. A ligação interna na arquitetura de computadores ou entre computadores e periféricos próximos são exemplos da aplicação da transmissão paralela. Esta forma de transmissão permite uma maior velocidade de envio de dados, devida ao paralelismo. Por outro lado, são necessários mais condutores, o que torna o cabo de transmissão mais caro. Além disso, como os sinais sendo transmitidos nos diversos condutores paralelos tem que ter um referencial elétrico comum (sinal terra), a comunicação apresenta elevada sensibilidade à ruídos (perturbações eletromagnéticas).

Na *transmissão serial*, mais adequada a comunicação entre equipamentos separados por grandes distâncias, os bits são encaminhados serialmente através de uma única linha de comunicação, usualmente com 2 ou 4 condutores.

Pode-se considerar outros parâmetros para a classificação dos modos de transmissão, como, por exemplo, a forma de sincronia entre emissor e receptor, associada à temporização. No caso particular das transmissões seriais, a forma de delimitar os bits pode levar em conta duas diferentes filosofias: a *transmissão síncrona* e a *transmissão assíncrona*.

Na *transmissão síncrona*, os bits de dados são enviados segundo uma cadência pré-definida, obedecendo a um sinal de temporização (clock). O receptor, por sua vez, conhecendo os intervalos de tempo que permitem delimitar um bit, poderá identificar a seqüência dos bits fazendo uma amostragem do sinal recebido.

Na *transmissão assíncrona*, não existe a fixação prévia de um período de tempo de emissão entre o transmissor e o receptor. A separação entre os bits é feita através de um sinal

especial com duração variável. Um caso típico de transmissão assíncrona é a transmissão de caracteres; neste caso, a cada grupo de bits constituindo um caracter são adicionados bits especiais para representar o início (*start bit*) e final deste (*stop bit*). Neste tipo de comunicação, apesar da assincronia ao nível de caracteres, ocorre uma sincronização ao nível de bit.

Um outro aspecto a ser destacado é a maneira como a banda passante (faixa de freqüência de transmissão) do canal de comunicação é explorada.

Em um primeiro modo, a *transmissão em banda base* ("*baseband"*), a banda passante do suporte de transmissão é atribuída totalmente a um único canal de transmissão. Neste modo, os sinais são transmitidos através do meio de comunicação multiplexados no tempo. Em pequenas distâncias, uma taxa de transmissão de até 10 Mbit/s pode ser obtida.

Em um segundo modo, a *transmissão em banda larga* ("*broadband"*), a banda passante do suporte de transmissão é dividida num determinado número de canais de faixa de freqüência estreita, permitindo que os dados sejam transmitidos utilizando uma técnica de multiplexação em freqüência. A banda passante dos canais é normalmente definida em função da taxa de transmissão desejada e do modo de modulação empregado. Neste modo de transmissão, cada canal pode atingir uma taxa de transmissão de até 3 Mbit/s, inferior, portanto, à transmissão em banda base. Esta técnica permite a definição de canais diferentes de transmissão sobre o mesmo guia físico e é utilizada para a realização da comunicação full-duplex.

b) Transmissão analógica versus transmissão digital

A *transmissão analógica*, na qual os sinais são transmitidos sob a forma de tensões ou correntes de amplitudes ou freqüências variáveis, tem dominado historicamente a indústria das telecomunicações. No entanto, a presença cada vez maior dos computadores e sistemas a base de microprocessadores tende a mudar esta filosofia de transmissão de dados.

Até bem recentemente, a tecnologia telefônica era baseada na transmissão de sinais analógicos. Apesar da grande difusão dos sistemas a transmissão digital, ainda serão necessários muitos desenvolvimentos para que a transmissão analógica desapareça totalmente, se isto vier a ocorrer.

A *transmissão digital* consiste no transporte de dados na forma de trens de sinais representando os níveis lógicos 0 e 1. A correta interpretação da informação transmitida requer que o emissor e o receptor adotem uma codificação idêntica para determinar o significado de cada seqüência de bits (como o código ASCII ou EBCDIC). Por exemplo, a letra "A", no código ASCII (American Standard Code for Information Interchange), seria

representada por uma seqüência de 8 bits na forma 01100101. As diferentes formas de codificação dos sinais 0 e 1 serão discutidas no item seguinte.

A adoção da técnica digital constitui uma evolução no campo da transmissão de dados e isto considerando duas principais vantagens:

- as taxas de erro de transmissão digital são muito inferiores àquelas da transmissão analógica. Na transmissão analógica, os amplificadores utilizados para compensação dos enfraquecimentos de sinal não compensam de maneira ótima, o que freqüentemente introduz distorções nestes sinais. Isto não ocorre no caso dos repetidores utilizados em transmissão digital;

- uma maior facilidade para a multiplexação das informações, antes da sua transmissão através de um único suporte de comunicação.

Um outro ponto a favor da transmissão digital são as constantes quedas nos preços dos componentes digitais, o que conduz a sistemas de transmissão digital tão baratos quanto seus equivalentes para transmissão analógica. Uma das técnicas mais utilizadas em transmissão digital é a *PCM (Pulse Code Modulation*), onde os sinais analógicos são transformados em sinais digitais através de uma amostragem.

Dentro dos sistemas a base de transmissão analógica, pode-se destacar as redes telefônicas públicas, os modems e algumas interfaces industriais. Aqui, a informação a ser transmitida é representada por variações contínuas de amplitude ou freqüência entre extremos pré-definidos, como por exemplo, entre -10V e +10V, entre 0 mA e 20 mA ou entre 4 mA e 20 mA (níveis usados em comunicação de dados para sensores e atuadores na indústria). Os sinais transmitidos pelo sistema telefônico são sinais sinusoidais, com faixa de freqüência entre 300 e 3400 Hertz, limitada graças à adição de filtros.

A conexão de um assinante a um centro de conexão é implementada normalmente na forma de um par de fios trançados. Se os filtros limitadores da faixa de freqüência são retirados, é possível então, dependendo da qualidade do cabo utilizado, transmitir sinais digitais pela rede telefônica a uma taxa de 1 a 2 Mbit/s sobre uma distância de alguns quilômetros.

c) Técnicas de codificação de bits

Na transmissão digital existem diferentes formas de codificar os sinais lógicos 0 e 1. Em uma primeira abordagem, o sinal 1 pode ser representado por um nível de tensão enquanto o sinal 0 é representado por outro (ou por dois intervalos de tensão, como veremos a seguir).

Uma forma possível de representação dos sinais 0 e 1 é a codificação *binária*, que usa um sinal do tipo "on-off", como mostrado na figura 2.3.5(a). Aqui, o sinal lógico 1 é representado por uma tensão positiva (por exemplo, +5V) enquanto o nível 0 é representado por 0V (sinal ativo alto). Outra opção é a codificação *bipolar*, que representa o sinal lógico 1 por uma tensão positiva (por exemplo, +12V) e o sinal lógico 0 por uma tensão negativa (por exemplo, -12V). É possível também definir os sinais de forma oposta, isto é, uma tensão negativa ou nula para o sinal lógico 1 e uma tensão positiva para o sinal lógico 0 (sinal ativo baixo). Na prática, devido a possibilidade de presença de perturbações e a queda de tensão causada pela própria impedância do meio de transmissão, trabalha-se com intervalos de tensão: por exemplo, uma tensão entre +3 e +12V é considerada como representando o sinal lógico 1 (ou 0), enquanto uma tensão entre -3 e -12V é aceita como lógico 0 (ou 1). Qualquer sinal no intervalo entre -3 e +3V é considerado "indefinido" e indica um erro de transmissão.

Quando um sinal digital é aplicado a uma das extremidades de um cabo, ele será recebido na outra extremidade mais ou menos enfraquecido e deformado, isto devido às características do cabo, particularmente, ao seu comprimento e à sua impedância.

A transmissão analógica faz uso da modulação de um sinal portador sinusoidal, usualmente com uma freqüência compreendida entre 1 e 2 MHz. A *modulação de amplitude* consiste em modificar a amplitude do sinal da portadora com as informações digitais a transmitir. Por exemplo, um valor da amplitude é atribuído à informação binária 0 e outro ao 1.

A *modulação em freqüência* utiliza o mesmo formalismo, mas o parâmetro considerado aqui é a freqüência, utilizando, desta vez, dois valores distintos de freqüência para indicar os valores binários 0 e 1. Esta forma de modulação é denominada transmissão em *banda portadora* ("*carrierband"*) ou FSK (Frequency Shift Keying). As figuras 2.3.5(b) e 2.3.5(c) ilustram os processos de modulação em amplitude e em freqüência, respectivamente.

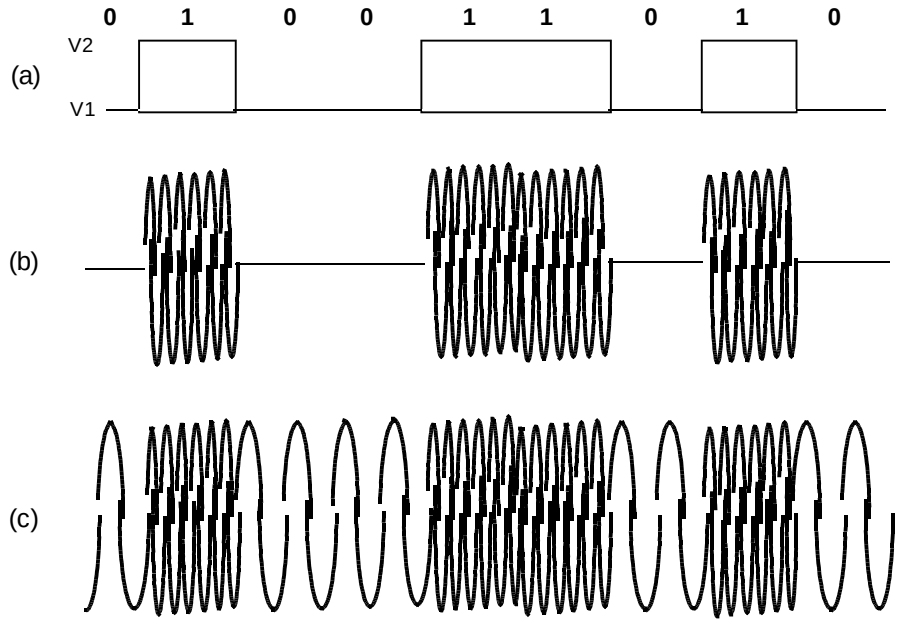


Figura 2.3.5 - (a) Sinal digital; (b) Sinal modulado em amplitude (ASK); (c) Sinal modulado em freqüência (FSK).

O equipamento capaz de tratar um fluxo de informações binárias num sinal modulado, ou reconstituí-lo a partir do sinal modulado é o *modem* (***mo****dulador-****dem****odulador*). O modem é normalmente inserido entre o equipamento informático e o meio de transmissão.

d) Técnicas de sincronização em nível de bits

Para a correta leitura do sinal digital transmitido, o receptor precisa conhecer a freqüência adotada pelo emissor e amostrar o sinal recebido em pontos adequados. A amostragem deve ocorrer o mais próximo possível do centro do sinal, uma vez que a transição de uma tensão (ou freqüência) que representa o sinal 0 para outra que representa o sinal 1 leva, na prática, um tempo diferente de zero. Para tal, é necessário conhecer o tempo médio de duração de cada sinal 0 ou 1, que depende da freqüência de transmissão adotada e é denominado "tempo bit", identificar o flanco de transição de 0 para 1 ou de 1 para 0 e ler o valor da tensão após metade do tempo bit, isto é, no centro do intervalo, como ilustrado na figura 2.3.6.

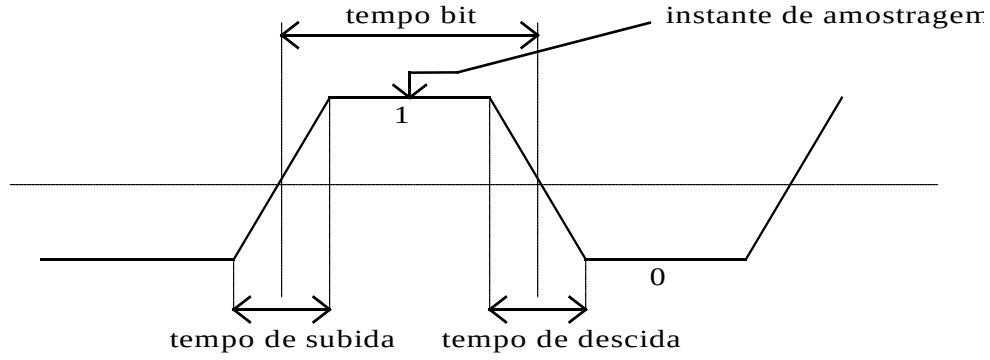


Figura 2.3.6 - Tempo bit e amostragem de sinal

Na transmissão síncrona, um sinal de temporização ("clock") é enviado por meio de um par separado de fios e define os momentos corretos de amostragem no receptor. O inconveniente está justamente na necessidade de adotar um segundo par de condutores, além daquele usado para o envio dos dados, o que encarece o cabo.

Na transmissão assíncrona é usado apenas um par de fios, mas uma forma de sincronização a nível de bits se faz necessária para a determinação do momento de amostrar o sinal. Já que aqui não há um sinal separado de sincronização, é necessário garantir que o trem

de sinais contenha muitas transições, que podem ser detectadas facilmente por circuitos eletrônicos adequados, e fazer a amostragem em um instante pré-determinado após estas.

Para garantir a presença de transições suficientes no sinal de modo a permitir uma correta sincronização, são adotadas técnicas alternativas de codificação dos sinais 0 e 1.

Uma destas técnicas é a codificação RZ (*Return to Zero*), que é usada em conjunto com a codificação bipolar já vista anteriormente. Aqui, o sinal sempre retorna a zero entre dois dígitos consecutivos, como ilustrado na figura 2.3.7. Isto tem o efeito de garantir a presença de transições mesmo que o sinal contenha uma longa seqüência de sinais lógicos 0 ou 1 seguidos (e que normalmente não apresentariam transições em uma codificação bipolar simples). A amostragem é feita no meio do tempo bit, contado a partir da detecção da transição (logicamente tanto o emissor quanto o receptor devem usar o mesmo tempo bit, isto é, devem operar na mesma freqüência).

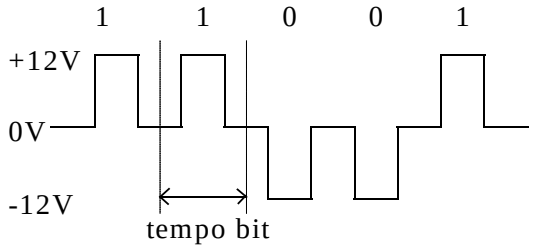


Figura 2.3.7 - Codificação RZ

Outra técnica é a chamada codificação NRZ (*Non Return to Zero*), ou *Manchester*. Aqui, o sinal lógico 1 é representado por uma *transição* entre uma tensão positiva e uma negativa (figura 2.3.8(a)), enquanto o sinal lógico 0 é representado por uma *transição* entre uma tensão negativa e uma positiva (figura 2.3.8(b)). A forma resultante de um trem de sinais é exemplificada na figura 2.3.8(c). Da mesma forma que na técnica anterior, isto garante a presença de transições suficientes para a sincronização do sinal. Neste caso, a amostragem do sinal é feita na segunda metade do tempo bit.

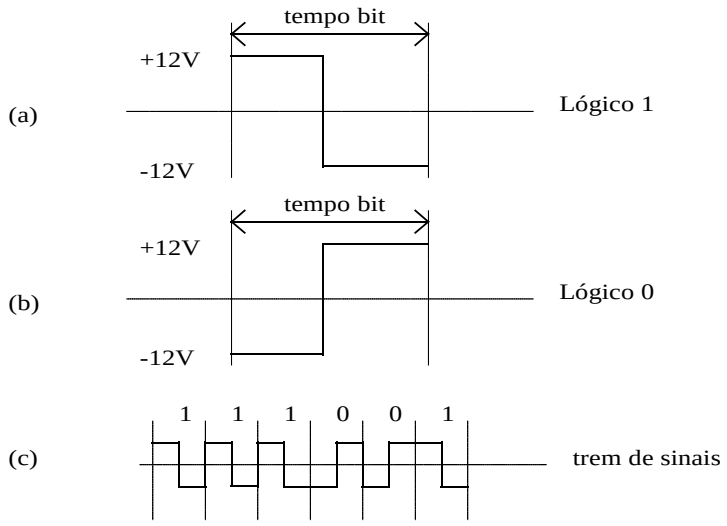


Figura 2.3.8 - Codificação NRZ (Manchester): (a) Sinal lógico 1; (b) Sinal lógico 0; (c) exemplo: seqüência 111001.

Uma terceira técnica é a chamada NRZI (*Non Return to Zero Inverted*), também conhecida como *Manchester Diferencial* ou *Manchester Modificada*. Nesta técnica, a ocorrência de uma transição no inicio do tempo bit indica a presença de um zero no trem de sinais, conforme ilustrado na figura 2.3.9. Se não houver transição (flanco) no inicio, o sinal é interpretado como lógico 1, independente do valor de tensão lido. Note que o sinal lógico 1 também possui uma transição, porém esta fica no meio do intervalo de sinalização e não no inicio.

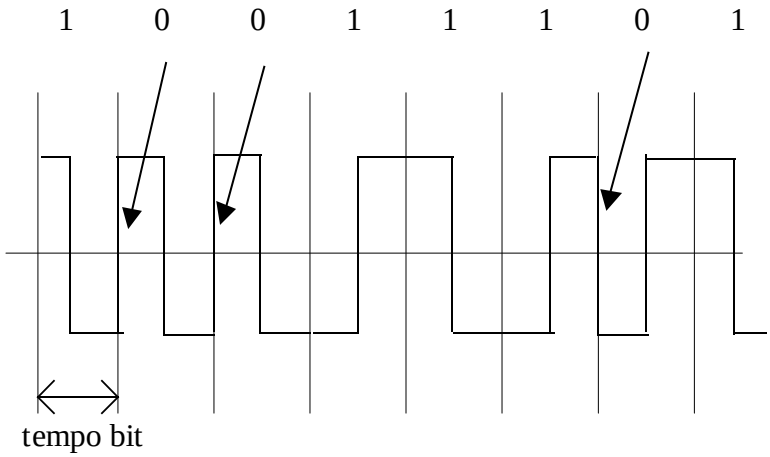


Figura 2.3.9 - Codificação NRZI (Manchester Diferencial)

e) Multiplexação

A multiplexação é um mecanismo muito utilizado na transmissão de dados, pois ela permite otimizar a utilização de um suporte de comunicação.

Duas técnicas de multiplexação são mais utilizadas: a técnica de Multiplexação em Freqüência ou FDM (Frequency Division Multiplexing) e a técnica de Multiplexação no Tempo ou TDM (Time Division Multiplexing).

A técnica FDM consiste em permitir o compartilhamento da faixa de freqüências oferecida pelo suporte de comunicação entre diferentes usuários. Assim, cada faixa de freqüências deve estar permanentemente alocada a um usuário de forma exclusiva.

Um exemplo de aplicação desta técnica consiste na transmissão de ondas de rádio em ondas médias, onde a banda passante situa-se entre 500 kHz e 1.500 MHz. Diferentes portadoras situadas nesta faixa de freqüência são atribuídas a estações de radiodifusão, mantendo, porém, entre estas, um espaço suficiente para evitar interferências. A mesma técnica pode ser usada em redes para a transmissão simultânea de vários sinais sobre o mesmo guia físico (broadband).

A técnica TDM consiste em permitir o compartilhamento da capacidade de transmissão do suporte de comunicação entre as diversas estações durante um curto espaço de tempo (isto é, cada estação dispõe do meio só para si durante um intervalo de tempo). Esta técnica é bastante utilizada em telecomunicações, particularmente, na telefonia, onde, a cada 125 mseg, um usuário pode enviar um byte de informação.

2.3.2.3. Interfaces industriais típicas para a camada física

Ao nível da camada física existem diversas interfaces padronizadas em uso na industria. Uma interface define as características elétricas, mecânicas e funcionais de uma conexão física.

Exemplos bem conhecidos de interfaces padronizadas para esta camada são: RS-232-C, RS-422, RS-423 e RS-485, definidas pela EIA (Electronics Industries Association).

As interfaces RS-232-C, RS-422 e RS-423 permitem somente a ligação ponto-a-ponto entre dois elementos de comunicação, suportando assim redes com topologias em estrela, anel, árvore ou malha. Enquanto a RS-232-C opera com sinais refereciados a terra, o que os torna sensíveis à perturbações, as interfaces RS-422 e RS-423 trabalham com sinais diferenciais, reduzindo consideravelmente a sensibilidade à ruídos e permitindo o uso de cabos mais longos que os usados com RS-232-C.

A interface RS-485 permite a conexão de até 32 elementos de comunicação (nós ou estações) em um mesmo barramento, suportando assim redes de difusão.

Em função de sua grande aceitação nos meios industriais, a interface RS-232-C será apresentada com um pouco mais de detalhes aqui. A EIA define dois elementos envolvidos

numa comunicação, o DTE (Data Terminal Equipment) e o DCE (Data Circuit-Terminating Equipment).

O conector adotado nesta interface possui 25 pinos (DB25), de forma trapezoidal, correspondendo à norma ISO-2110. Os pinos superiores, em número de treze, são numerados de 1 a 13 da esquerda para a direita, os pinos inferiores são numerados de 14 a 25, também da esquerda para a direita. Alguns equipamentos mais recentes utilizam um conector com apenas 9 pinos (DB9).

No que diz respeito às especificações elétricas, uma tensão negativa inferior a -3 volts permite representar uma informação binária 1 e uma tensão positiva, superior a +3 volts, permite representar uma informação binária 0 (sinal ativo baixo). A taxa de transmissão média é de aproximadamente 20 Kbit/s num cabo de comprimento não superior a 15 metros. Ambos os equipamentos envolvidos na comunicação tem que ser ajustados para a mesma freqüência.

No que diz respeito as especificações funcionais, podemos descrever os diferentes sinais envolvidos numa comunicação com auxílio da figura 2.3.10, que permite representar os sinais mais utilizados em aplicações típicas.

Quando o DTE torna-se ativo, ele apresenta na interface o *DTR* (*Data Terminal Ready*), pino 20, a 1. Quando o DCE é ativado ele mantém o sinal *DSR* (*Data Set Ready*), pino 6, também a 1. Uma vez que o DCE (um modem, por exemplo) detecta a presença da portadora na linha, ele previne o DTE (o computador) colocando o sinal *CD* (*Carrier Detect*), pino 8, a 1. O DTE, por sua vez, indica ao modem a sua intenção de transmitir a partir do sinal correspondente ao pino 4, *RTS* (*Request To Send*). A resposta do modem é dada através do sinal *CTS* (*Clear To Send*), pino 5, o que significa que o DTE pode iniciar a transmissão. Os dados são então enviados pela linha *TD* (*Transmitted Data*), pino 2. No caso de uma transmissão em duplex, o DTE pode receber os dados, de forma simultânea ao envio, através da linha *RD* (*Received Data*), pino 3. O pino 2 do DCE deve ser ligado ao pino 3 do DTE e vice-versa.

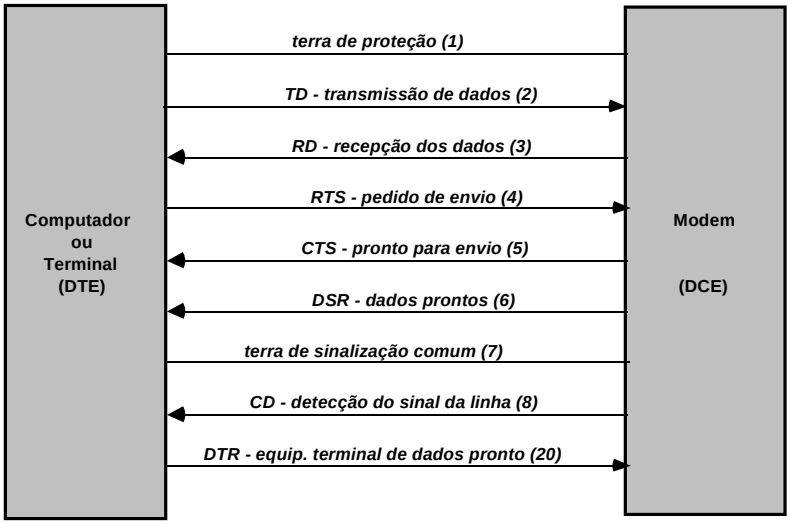


Figura 2.3.10 - Sinais típicos da interface RS-232-C.

A ligação mais simples entre dois equipamentos utiliza apenas as linhas RD (pino 3 para DB25, ou 2 para DB9), TD (pino 2 para DB25, ou 3 para DB9) e terra (pino 7 para DB25, ou 5 para DB9), onde o TD de um equipamento deve estar ligado ao RD do outro, conforme ilustrado na figura 2.3.11.

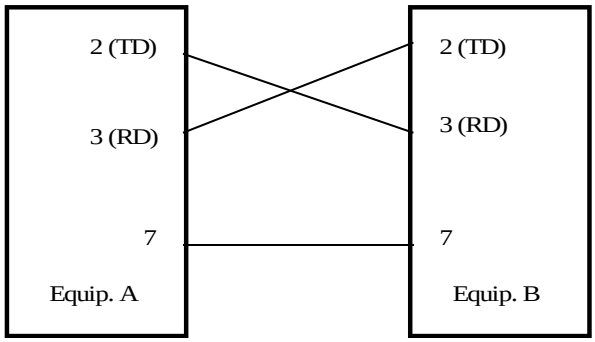


Figura 2.3.11 - Conexão mínima via RS-232-C

2.3.3. A CAMADA DE ENLACE DE DADOS

A camada de Enlace de Dados tem por função oferecer um caminho para o transporte das informações entre entidades da Camada de Rede, que será estudada na parte seguinte do documento. Dentre os fatores com os quais a camada de Enlace deve preocupar-se estão:

- a forma como os bits provenientes da camada Física serão agrupados em quadros;
- os mecanismos de detecção e correção de erros a serem implantados, uma vez que as informações trocadas através da camada Física não são isentas de erros de transmissão;
- os mecanismos de controle de fluxo para limitar o volume de informação trocados entre entidades fonte e destino;
- a gestão das ligações entre as entidades;
- o controle de acesso ao meio, em redes de difusão.

Nas redes de difusão a camada de enlace de dados é usualmente decomposta em duas subcamadas, conforme proposta da IEEE:

- Subcamada de Controle de Acesso ao Meio (MAC - Medium Access Control), responsável pelo acesso ordenado e compartilhado do canal de comunicação (no caso, um barramento);
- Subcamada de Controle Lógico de Enlace (LLC - Logical Link Control), responsável pelo estabelecimento de conexões e oferecimento de serviços de comunicação às camadas acima.

2.3.3.1. Subcamada MAC

Do ponto de vista da programação distribuída, o meio de transmissão das redes locais constitui um recurso compartilhado entre as estações a ela conectadas. Os métodos de definição de direito de acesso utilizados nas redes locais são os denominados protocolos de acesso ao meio.

Se tomarmos o tempo de acesso ao meio como critério, podemos organizar os protocolos de acesso ao meio em duas principais classes: os protocolos *determinísticos* e os *não determinísticos*.

Os protocolos de acesso determinísticos são caracterizados pela concessão do direito ao acesso independentemente das necessidades de transmissão de cada nó da estação. Dentre os protocolos conhecidos desta classe, podemos destacar o protocolo TDMA (Time Division Multiple Access), onde o acesso é dado a cada estação considerando faixas de tempo bem definidas. Este método apresenta um baixo desempenho, uma vez que muito tempo pode ser perdido no caso de estações que não tenham mensagens a transmitir. Outro exemplo de protocolos de acesso determinísticos são aqueles baseados na passagem de ficha (token passing), onde uma ficha correspondendo ao direito de transmissão é passada de estação a estação da rede. Ao receber a ficha, uma estação que não tenha mensagens a transmitir repassa a ficha à estação seguinte na lista de estações compondo a rede. Vamos analisar alguns destes protocolos mais adiante nesta seção.

Os protocolos de acesso não determinísticos, podem ser também denominados protocolos de competição, uma vez que as estações querendo transmitir vão competir pelo meio de transmissão. Estes protocolos são mais adaptados às condições de transmissão da rede local. Um exemplo desta classe são os protocolos de tipo CSMA, cujas variações serão estudadas nos parágrafos a seguir.

a) Os protocolos não determinísticos

- *CSMA persistente e não persistente*:

Estes protocolos, pertencentes à classe de protocolos ditos de detecção de portadora (carrier sense), baseiam-se no conceito de escuta do meio de transmissão para a seleção do direito de acesso a este.

Um primeiro exemplo deste protocolo é o *CSMA 1-persistente* (CSMA - Carrier Sense Multiple Access ou Acesso Múltiplo por Detecção de Portadora). Neste protocolo, quando uma estação está pronta a enviar um quadro de dados, ela escuta o que está ocorrendo no suporte de transmissão. No caso em que o canal já está sendo ocupado por alguma estação, a estação aguarda na escuta até que o meio esteja livre para a sua emissão (daí o nome

"*persistente"*); quando isto ocorre, ela pode então emitir um quadro. O método é chamado "1"-persistente porque, quando a linha esta livre, a estação enviará os dados com 100% de probabilidade. Após a transmissão dos dados, e estação emissora espera uma resposta (chamada *quadro de reconhecimento*) da estação receptora, indicando a correta recepção dos dados.

Se uma outra estação estava a espera de uma oportunidade de enviar dados ao mesmo tempo que a primeira, pode ocorrer que ambas detectem o meio como estando livre ao mesmo tempo. Neste caso, ambas irão enviar seus dados simultaneamente, de forma que o sinal no barramento será uma "mistura" ininteligível das duas mensagens. Esta condição recebe o nome de "*Colisão*". Na ocorrência de uma colisão, a estação receptora não envia o quadro de reconhecimento esperado e a estação emissora tenta a emissão novamente após um determinado tempo.

O protocolo CSMA 1-persistente é altamente influenciado pelo tempo de propagação dos quadros no suporte de transmissão. Isto é ilustrado pelo exemplo de duas estações A e B querendo emitir um quadro. Vamos supor que A detecta o meio livre e emite um quadro; em seguida, B vai escutar o meio para ver o seu estado; se o atraso de propagação do quadro emitido por A é tal que o sinal ainda não pode ser detectado a nível da estação B, então esta vai considerar o meio livre e emitir o seu quadro, gerando naturalmente uma colisão. Isto significa que, quanto maior o tempo de propagação no suporte de comunicação, pior o desempenho do protocolo devido à ocorrência de colisões.

Na verdade, embora as probabilidades não sejam muito grandes, as colisões podem ocorrer mesmo se o tempo de propagação é considerado nulo. Vamos supor agora as estações A e B com quadros a transmitir, mas que uma terceira estação, C está utilizando o meio. Neste caso, as duas estações vão aguardar a liberação do meio e, quando este estiver liberado, ambas vão emitir seus quadros, caracterizando a colisão.

Outro exemplo de protocolo CSMA é o *CSMA não persistente*. Segundo este protocolo, as estações comportam-se de maneira menos "afoita" para o envio de mensagens. Assim, uma estação que deseje emitir um quadro vai escutar o suporte de transmissão para verificar se este está disponível. Em caso positivo, o quadro será transmitido. Caso contrário, ao invés de ficar escutando à espera da liberação do canal, ele vai esperar um período de tempo aleatório e, após a expiração deste, vai escutar o canal novamente para verificar a sua liberação (ou não). Este protocolo permite reduzir as possibilidades de ocorrência de colisões, embora ele introduza um maior atraso de emissão a nível das estações que o protocolo persistente.

O *CSMA p-persistente* é mais um exemplo de protocolo de acesso, funcionando da seguinte maneira: quando uma estação tem um quadro a enviar, ela escuta o canal para verificar a disponibilidade; se o canal está disponível , a probabilidade da estação emitir o quadro é igual a p. A probabilidade de que esta aguarde o próximo intervalo de tempo é igual

a q = 1 - p; se, no início do segundo intervalo de tempo, o canal está disponível, as probabilidades de envio ou de espera continuam as mesmas; o processo continua, então, até que o quadro seja finalmente transmitido ou que outra estação tenha tomado posse do canal.

- *O protocolo CSMA/CD (CSMA com detecção de colisão)*:

Os protocolos descritos até aqui, embora apresentando aspectos interessantes, podem ser melhorados considerando-se que cada estação poderia detectar, antes da emissão, o estado de conflito com outras estações da rede, evitando assim a emissão do quadro considerado.

O protocolo CSMA/CD (Carrier Sense Multiple Access with Collision Detection) é um protocolo baseado neste princípio e muito utilizado nas redes locais (foi proposto originalmente pelos criadores da rede Ethernet). Neste protocolo, quando mais de uma estação esta pronta para emitir uma mensagem com o meio livre, estas emitem o quadro, o que vai gerar uma colisão. A primeira estação que detectar a colisão interrompe imediatamente a sua transmissão, reiniciando o processo todo após a expiração de um período de tempo aleatório, de forma a tornar improvável a ocorrência de uma nova colisão (figura 2.3.3.1). Para detectar a colisão, a estação emissora deve escutar aquilo que ela mesma colocou no meio (ao menos a primeira palavra de código enviada deve ser escutada pela própria estação emissora).

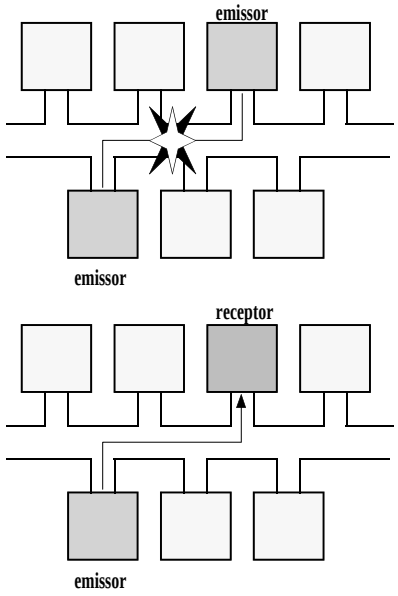


Figura 2.3.3.1 - Método de acesso CSMA/CD

Para melhor entender o mecanismo deste protocolo, vamos analisar o caso em que duas estações iniciem uma transmissão num instante de tempo t0. O tempo mínimo para a detecção de uma colisão é o tempo de propagação do sinal emitido por uma estação até a outra estação.

Isto significa, em uma primeira análise que, se uma estação que emitiu um quadro não detecta uma colisão num período de tempo igual ao tempo de propagação do sinal ao longo do canal de comunicação, pode considerar-se possuidora do meio e que as demais estações abstiveram-se de enviar.

Esta análise apresenta, porém, uma imprecisão. Vamos considerar t o tempo de propagação de um sinal entre duas estações, as mais distantes uma da outra. Consideremos que em t0 uma estação emite um quadro. Vamos supor que em t - e , pouco antes que o sinal tenha chegado, outra estação efetue uma transmissão. Ela vai detectar, imediatamente a colisão e interromper a sua emissão. Por outro lado, a perturbação ocasionada pela colisão não vai poder ser detectada pela primeira estação que emitiu o quadro antes de um período de tempo igual a 2t - e. Assim, no pior caso, uma estação só poderá estar segura de que ela adquiriu o acesso ao canal de transmissão após um período de tempo de 2t. Assim, no protocolo CSMA/CD, uma estação emite uma palavra de dados, espera por um tempo 2t e, se não detectar nenhuma colisão, considera que é a única a usar o meio naquele momento e passa a enviar o restante dos dados.

O método CSMA/CD propicia uma grande otimização no uso do meio em relação aos protocolos anteriores, pois a ocorrência de uma colisão é detectada logo na primeira palavra e a emissão é interrompida, não tendo que ser completamente repetida depois. Vale salientar que nada impede que o CSMA/CD seja também 1-persistente ou p-persistente.

No entanto, em todos os métodos de acesso CSMA temos que, quanto maior o número de estações, maior a probabilidade de ocorrência de colisões (esta probabilidade aumenta exponencialmente), de forma que o tempo de reação aumenta consideravelmente e não pode ser exatamente determinado. Em função deste comportamento, métodos de acesso não determinísticos são considerados inadequados para aplicações em tempo real, muito comuns em ambiente fabril.

b) Os protocolos determinísticos

Os métodos de acesso determinísticos são aqueles com tempo de resposta univocamente determinável. Estes métodos podem ser classificados em:

- métodos com comando centralizado (ex.: Mestre-Escravos) e
- métodos com comando distribuído (ex.: Token-Passing).

Nos sistemas com *comando centralizado*, somente uma estação pode agir como detentora do direito de transmissão (Mestre). O direito de acesso ao meio físico é distribuído por tempo limitado pela estação mestre as demais (Escravas). Aqui todas as trocas de dados ocorrem apenas entre mestre e escravos (figura 2.3.3.2). Esta configuração deixa o sistema

dependente da estação central, mas é a configuração usual dos sistemas de controle na maioria de suas aplicações. Este método de acesso ao meio também garante um tempo entre transmissões consecutivas a qualquer estação da rede e segue a prática atual de fazer um controle distribuído com uma supervisão centralizada.

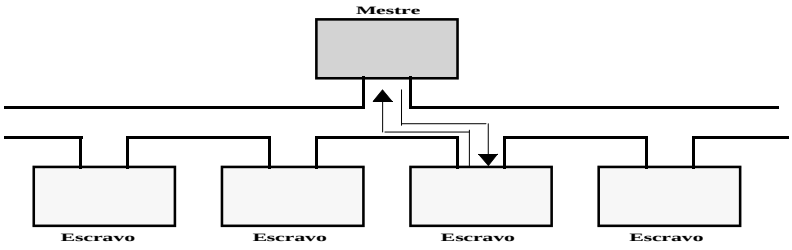


Figura 2.3.3.2 - Método de acesso mestre/escravos

Os sistemas com *comando distribuído* permitem a definição de mais de uma estação com direito de acesso ao meio físico. Este direito de acesso (chamado "Token") é transmitido ciclicamente entre as várias estações, que podem livremente trocar dados entre si (figura 2.3.3.3). Este sistema é, no entanto, bem mais complexo do que o Mestre-Escravos, já que providências especiais tem que ser tomadas no caso da perda do token ou da entrada ou saída de uma das estações da rede. Este método é mais adequado para sistemas nos quais diversas unidades independentes desejam trocar livremente informações entre si. Neste método, é possível determinar um tempo máximo entre duas oportunidades consecutivas de transmissão para cada estação.

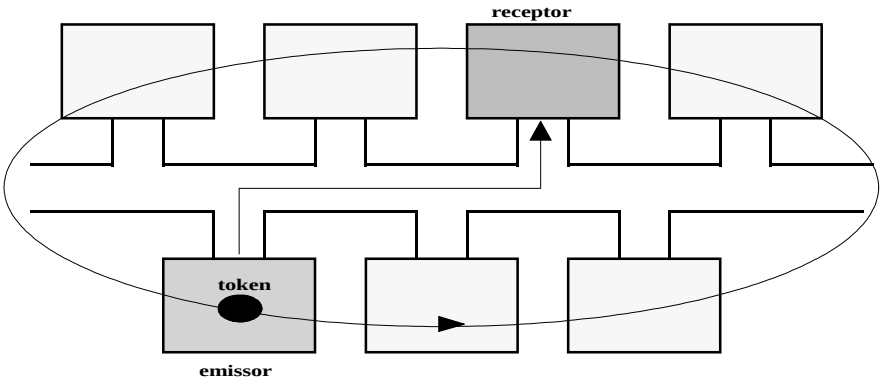


Figura 2.3.3.3 - Controle de acesso por passagem de Token

2.3.3.2. Subcamada LLC

a) As Classes de Serviços de Enlace

A subcamada LLC da camada de Enlace de Dados oferece serviços classificados em três principais categorias, estas dependendo do sistema no qual elas serão implantadas:

- serviço sem conexão e sem reconhecimento (não confiável);
- serviço sem conexão com reconhecimento (confiável);
- serviço orientado à conexão (confiável).

Na primeira classe de serviços, a máquina fonte da informação envia os quadros de dados à máquina destinatária sem recepção de reconhecimento da informação enviada; além disso, não existe estabelecimento prévio de conexão e, conseqüêntemente, não existe liberação desta ao final do diálogo. Se um quadro de dados é perdido no suporte de transmissão como conseqüência de um ruído, não existe nenhum mecanismo que permita solucionar o problema. Esta classe de serviços é adequada quando implantados sobre um suporte de comunicação cuja taxa de erros é muito baixa ou que a correção dos erros é prevista nas camadas superiores. Eles podem ser empregados em algumas aplicações tempo-real e em redes locais.

A segunda classe de serviços, embora ainda não defina o estabelecimento prévio de conexão, prevê a existência de quadros de reconhecimento, de modo que a máquina fonte será notificada pela máquina destinatária da recepção do quadro previamente enviado. Um mecanismo que pode ser implantado no caso de perda do quadro — o que corresponde à não recepção do quadro de reconhecimento após um certo tempo (*timeout*) — é a reemissão daquele. Num serviço sem conexão, existe a possibilidade da reemissão de quadros provocando a recepção múltipla do mesmo quadro (duplicação de mensagem).

A terceira classe de serviços é a mais sofisticada, uma vez que ela define a necessidade do estabelecimento prévio de conexão e a liberação destas ao final do diálogo. Neste caso, cada quadro enviado é numerado e a camada de Enlace garante que cada quadro enviado será recebido, uma única vez, e que o conjunto de quadros enviados será recebido ordenado da mesma forma que foi enviado. Esta classe de serviços oferece à camada de Rede um canal de comunicação confiável.

Os serviços orientados à conexão são caracterizados por três principais etapas:

- a etapa de *estabelecimento de conexão*, durante a qual são definidos todos os parâmetros relacionados à conexão, como por exemplo, os contadores de seqüência de quadros;

- a etapa de *transmissão de dados*, durante a qual são realizadas todas as trocas de informação correspondentes ao diálogo entre duas máquinas;

- a etapa de *liberação da conexão*, que caracteriza o fim do diálogo e na qual todas as variáveis e outros recursos alocados à conexão serão novamente disponíveis.

A comunicação entre as camadas de Rede e de Enlace é feita através de *primitivas de serviço* (*request*, *indication*, *response* e *confirm*) da interface entre as duas camadas, como definidas na parte relativa à apresentação do modelo OSI. Um esquema permitindo demonstrar o modo de utilização das primitivas é mostrado na figura 2.3.3.4. Neste esquema, representa-se nas extremidades a camada de Rede, caracterizada pelos dois sistemas envolvidos no diálogo (no caso, A e B, que são as entidades pares) e no centro as camadas de Enlace (de A e B), que tornam a comunicação transparente para as camadas de rede envolvidas.

Os deslocamentos verticais das primitivas representam o tempo decorrido entre as ocorrências destas (por exemplo, no caso das primitivas request e indication, o tempo decorrido entre o envio da primeira pela camada de Rede do sistema A e a recepção da segunda pela camada de Rede do sistema B).

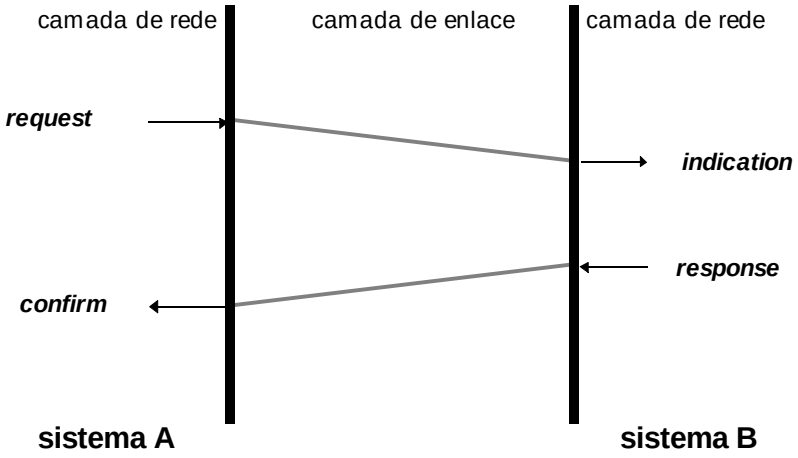


Figura 2.3.3.4 - Primitivas de serviço trocadas entre as camadas de Rede e de Enlace.

b) O conceito de quadro

Para que o serviço seja oferecido à camada de Rede, a camada de Enlace utiliza-se dos serviços fornecidos pela camada Física que, como já foi descrito na parte precedente, é responsável da transmissão de bits de um ponto a outro na rede de comunicação, sendo que o conjunto de bits transmitido pode sofrer distorções produzindo erros de transmissão. Uma conseqüência típica pode ser que o número de bits recebidos seja inferior ao número de bits enviados ou os valores de alguns bits podem ter sido modificados.

Com o objetivo de permitir um controle de erro eficiente, a camada de Enlace decompõe as mensagens em porções menores denominadas quadros, aos quais são adicionados códigos especiais de controle de erro. Desta forma, o receptor pode verificar se o código enviado no contexto de um quadro indica ou não a ocorrência de erros de transmissão e ele pode, assim, tomar as providências necessárias para evitar as conseqüências devido àquele erro.

A definição e delimitação dos quadros pode obedecer a diferentes políticas. Uma possível política a adotar pode ser, por exemplo, a contagem de caracteres. Nesta política, é introduzido um caracter especial que indica o número de caracteres compondo o quadro. Deste modo, a nível da camada de Enlace do receptor, basta que a entidade leia este caracter e em seguida conte o número de caracteres para definir o tamanho do quadro. O inconveniente desta técnica, no entanto, é que o caracter que define o tamanho do quadro pode ser deturpado durante a transmissão, o que significa que o receptor vai ler erroneamente os quadros transmitidos. A figura 2.3.3.5 ilustra este problema.

Uma técnica melhor, que apresenta uma solução a este problema, consiste na adição de seqüências especiais de caracteres de modo a representar o início e fim da transmissão de um quadro.

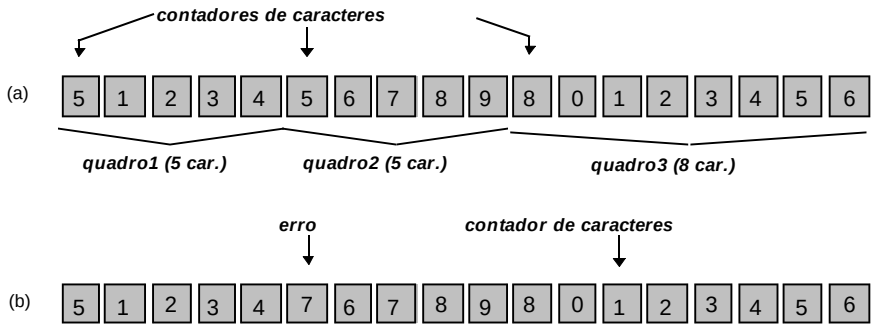


Figura 2.3.3.5 - Seqüência de caracteres: (a) sem erro; (b) com erro.

A figura 2.3.3.6 ilustra um caso relativo a esta técnica, onde a seqüência de caracteres *DLE* (*Data Link Escape*) e *STX* (*Start of TeXt*) é inserida para representar o início de um quadro e a seqüência *DLE* e *ETX* (*End of TeXt*) para representar o fim do quadro. Esta técnica apresenta, ainda, um ponto de vulnerabilidade: se, dentro do texto, uma seqüência de bits coincide com uma das seqüências de caracteres citada (dado que os bits podem assumir qualquer combinação de valores), a entidade receptora na camada de Enlace pode ser "enganada" por esta seqüências e, assim, interpretar erroneamente o quadro.

A solução a este problema vem através da introdução, pela entidade de Enlace emissora, de um caracter *DLE* adicional a cada vez que, em uma seqüência de bits da parte de dados, aparecer um byte que coincidir com um caracter *DLE*. Desta forma, basta à entidade de dados receptora eliminar os caracteres *DLE* dos dados antes de emiti-los à camada de Rede. Isto vai permitir então, às entidades receptoras de Enlace, fazer a distinção entre as seqüências delimitadoras de quadro (*DLE-STX* e *DLE-ETX*) das seqüências «acidentais» nos dados. Os caracteres *DLE* introduzidos a nível dos dados são denominados *caracteres de transparência*. Esta técnica de delimitação de quadros é utilizada em protocolos simples de comunicação serial orientados à caracter.

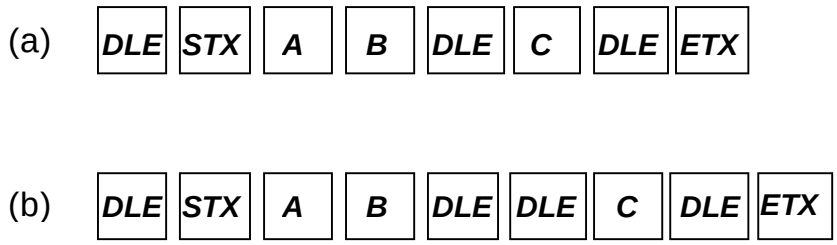


Figura 2.3.3.6 - (a) Dados enviados pela camada de Rede com seqüências de delimitação; (b) idem, com introdução dos caracteres de transparência.

Uma técnica muito utilizada e definida mais particularmente para a utilização em redes de comunicação é aquela em que os quadros são delimitados por uma seqüência de bits particular, mas desta vez dissociados da codificação de caracteres. A seqüência 01111110 é freqüentemente adotada para representar a delimitação dos quadros.

De maneira análoga à utilização dos caracteres de transparência da técnica anterior, *bits de transparência* são introduzidos a nível da parte de dados para evitar a confusão, por parte do receptor, com os delimitadores de quadro (esta técnica também é conhecida como "bit-stuffing"). Assim, no receptor, a cada vez que 5 bits “1” consecutivos são detectados na parte de dados, um bit 0 é adicionado após a seqüência. Do lado do receptor, a cada vez que ocorrer uma seqüência de 5 bits “1” consecutivos de dados seguidos de um bit 0, este último será eliminado da parte de dados. A figura 2.3.3.7 ilustra a aplicação desta técnica. Em 2.3.3.7 (a) são apresentados os dados originais e, em 2.3.3.7(b), os mesmos dados com a introdução dos bits de transparência.

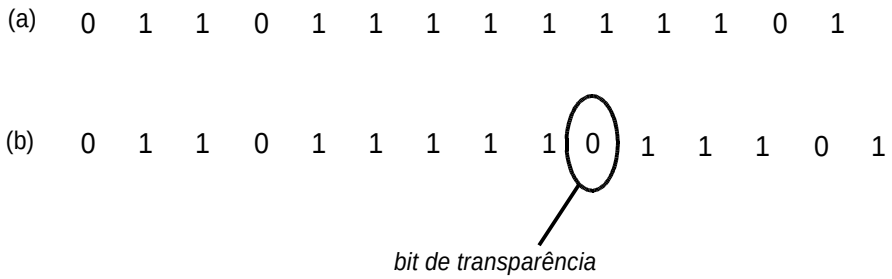


Figura 2.3.3.7 - (a) dados originais; (b) dados com adição do bit de transparência

2.3.3.3. O controle de erros

Os erros que podem ocorrer sobre os suportes de transmissão tem como causas os mais diversos fenômenos físicos, como por exemplo, o ruído térmico, provocado pela agitação dos elétrons nos cabos de cobre. Outro fenômeno importante são os ruídos impulsivos causados pelo chaveamento de relês ou outros dispositivos eletromecânicos.

Verifica-se entretanto que, independentemente do fenômeno causador de erro, estes tendem a gerar normalmente verdadeiros pacotes de erros (*error bursts*) e não erros simples.

Isto pode ter um aspecto positivo, uma vez que, num conjunto relativamente grande de bits, um menor número de pacotes vai conter erros. Por outro lado, os erros agrupados em pacotes são mais difíceis de modelizar e de detectar.

O controle de erros de transmissão é uma das funções mais importantes asseguradas pela camada de enlace (usualmente implementado na subcamada LLC).

Os protocolos de controle de erro são caracterizados, em geral, pela definição de um quadro de controle, correspondente a um reconhecimento positivo ou negativo. Caso a entidade emissora receba um reconhecimento positivo de um quadro previamente enviado, ela entende que aquele foi corretamente recebido.

Por outro lado, se ela recebe um reconhecimento negativo, ficará ciente de que o quadro foi mal transmitido e que, neste caso, ele deverá ser retransmitido.

Ainda, se, por uma intensidade relativamente forte de ruído, o quadro inteiro não é recebido pela entidade destinatária, esta não vai reagir ao quadro emitido e a entidade emissora corre o risco de esperar indefinidamente pelo reconhecimento. I isto é evitado pela adição de temporizadores, estabelecendo assim um tempo máximo de espera pelo reconhecimento, antes da retransmissão (*time-out*). O tempo de espera deve ser determinado em função dos atrasos relativos à transmissão dos quadros de modo que os quadros de reconhecimento, se existentes, cheguem antes do esgotamento da temporização.

Deste modo, se o quadro ou o reconhecimento são perdidos, a temporização será esgotada, podendo provocar a retransmissão do quadro. Neste caso, é possível que o quadro seja aceito mais de uma vez pela camada de Enlace e transmitido à camada de Rede, causando uma duplicação de quadros. Para evitar este problema, deve-se introduzir um mecanismo de distinção dos quadros a fim de que o receptor possa separar os quadros duplicados de seus originais.

Existem praticamente duas técnicas para o tratamento de erros. A primeira consiste na introdução, a nível dos quadros, de informações suficientemente redundantes que permitam ao receptor reconstituir os dados enviados a partir da informação recebida. A segunda técnica consiste em adicionar unicamente um conjunto de informações redundantes o suficiente para que o receptor possa detectar a ocorrência de um erro (sem corrigi-lo) e requisitar a retransmissão do quadro. A primeira técnica é denominada *código corretor de erros* e a segunda *código detetor de erros*.

Uma palavra de código de comprimento igual a *n* bits é composta de um número *m* de bits de dados e um número *r* de bits de controle (*n = m + r)*. Dadas duas palavras de código, por exemplo, 10001001 e 10110001, é possível determinar de quantos bits elas diferem (no caso do exemplo, elas diferem de 3 bits). Isto pode ser feito efetuando um “ou exclusivo” entre as duas palavras e contando o número de bits "1" do resultado. A este número é dado o nome de *Distância de Hamming*. Se a distância de Hamming entre duas palavras é *d*, serão necessários *d* erros simples para transformar uma palavra em outra.

A parte de dados da palavra de código pode ter até 2^m valores diferentes. Devido à forma com que os *r* bits de controle são calculados, nem todos os 2^n valores possíveis para a palavra de código são efetivamente utilizados. Sendo conhecida a técnica de cálculo dos bits de controle, é possível fazer uma lista de todas as palavras de código válidas. Pode-se agora verificar entre as palavras desta lista quais as que apresentam a menor distância de Hamming. Esta será a distância de Hamming do código como um todo.

A propriedade de um código ser detetor ou corretor depende da sua distância de Hamming. Para que seja possível detectar *d* erros numa palavra, é necessário que o código tenha uma distância de Hamming de *d*+1. Isto porque é impossível que *d* erros possam transformar uma palavra de código em outra palavra de código autorizada.

Por outro lado, para que seja possível corrigir *d* erros, a distância de Hamming do código deverá ser de *2d* + 1. Neste caso, mesmo que *d* erros simples ocorram, a palavra de código continua ainda mais próxima da palavra transmitida.

Assim, fica claro que um código corretor de erros precisa ter palavras válidas mais diferentes entre si do que as usadas com um código apenas detetor de erros.

Veremos a seguir exemplos de ambas as técnicas.

*a) Os códigos de correção de erro (*forward error control*)*

Os códigos corretores são utilizados para as transmissão de dados em casos particulares como, por exemplo, quando os canais são unidirecionais ou quando é impossível ou indesejável requisitar a retransmissão de um quadro.

Um exemplo extremamente simplificado de código corretor de erros pode ser visto a seguir, sendo este composto de apenas 4 palavras de código: 0000000000, 0000011111, 1111100000 e 1111111111.

Este código apresenta uma distância de Hamming de 5, o que significa que ele pode corrigir até 2 erros em cada palavra de código. Se o receptor detecta uma palavra de código igual a 0000000111, ele é capaz de reconhecer a palavra original como sendo 0000011111, pois esta é a palavra válida mais próxima da recebida. Por outro lado, se um erro triplo ocorre, modificando, por exemplo, a palavra 0000000000 em 0000000111, o receptor será incapaz de corrigi-lo.

Um código corretor de erros bastante usado na prática foi proposto pelo próprio Hamming em 1950. Este código só pode corrigir erros simples, de 1 bit. Nele, o número mínimo de bits de controle é dado por $(m + r + 1) \; 1 \; 2^r$, onde *m* é o número de bits de dados e *r* o número de bits de controle. No *código de Hamming*, cada palavra é composta por uma mistura de bits de dados e de controle, numerados da esquerda para a direita (assim, o bit 1 é o primeiro da esquerda). Todos os bits que são potências de 2 (1, 2, 4, 8, 16, etc...) são usados como bits de controle e os demais (3, 5, 6, 7, 9, 10, etc...) são usados como bits de dados (figura 2.3.3.8).

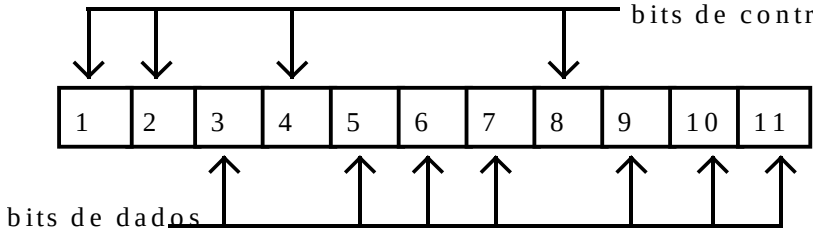


Figura 2.3.3.8 - Palavra de código de 11 bits, com 7 bits de dados e 4 de controle

Um determinado bit de dados *k* tem sua paridade controlada pelos bits de controle que fazem parte da expansão de *k* em potências de 2 (isto é, cuja soma é igual a *k*). Por exemplo, o bit de dados 3 é controlado pelos bits 1 e 2, o bit 5 pelos bits 1 e 4, o bit 6 pelos bits 2 e 4, o bit 11 pelos bits 1, 2 e 8, etc.

Os bits de controle são calculados de forma que um OU-exclusivo (XOR) entre eles e o bit de dados que estes controlam sempre é par. Por exemplo:

(Bit de dados 3) XOR (Bit de Controle 1) XOR (Bit de controle 2) = 1.

Quando uma palavra de código é recebida, a unidade receptora inicializa um contador em zero. Em seguida, a paridade de cada bit de controle é verificada. Se uma delas não estiver correta, o número do bit de controle onde o erro foi detectado é adicionado ao contador. Se o contador estiver em zero no final do processo, assume-se que não houve erro (ou, ao menos, nada foi detectado). Se o contador não estiver em zero, o seu valor indica o número do bit errado, que pode então ser corrigido por simples inversão, sem necessidade de retransmissão!

Para exemplificar, suponha que queremos transmitir o texto "Hamming code" entre dois computadores. Adotou-se palavras de código com 7 bits de dados, de forma que o número mínimo de bits de controle é 4. As palavras de código a transmitir são mostradas na tabela abaixo:

Caracter	Código ASCII	Código de Hamming
H	1001000	00110010000
a	1100001	10111001001
m	1101101	11101010101
m	1101101	11101010101
i	1101001	01101011001
n	1101110	01101010110
g	1100111	11111001111
	0100000	10011000000

c	1100011	11111000011
o	1101111	00101011111
d	1100100	11111001100
e	1100101	00111000101

*b) Os códigos de detecção de erro (*feedback error control*)*

O exemplo mais simples de código detetor de erros é o *controle de paridade*. Ele é baseado na adição de 1 bit de paridade à cada palavra de código enviada, obtido por meio de um OU-exclusivo (XOR) dos bits que compõem a palavra. Por exemplo: para a seqüência 00100110, o bit de paridade vale 1 (0 xor 0 xor 1 xor 0 xor 0 xor 1 xor 1 xor 0 = 1, paridade impar). Logo, o emissor envia a seqüência 001001101, que inclui o bit de paridade no final.

O receptor, por seu turno, realiza independentemente uma operação OU-exclusivo sobre os bits que vão sendo recebidos (isto é, calcula a paridade novamente) e compara o resultado obtido com o último bit enviado (que é o bit de paridade calculado pelo emissor); se houver diferença, houve erro de transmissão (isto é, algum bit foi distorcido). No entanto, se um número par de bits foi distorcido, o método não é capaz de detectar o erro. Assim, o bit de paridade permite detectar unicamente erros simples.

Uma variante melhorada deste método é a paridade longitudinal, conhecida como *BSC (Block Sum Check)*, usada para o conjunto de palavras de código que compõem um quadro. Aqui, além de acrescentar um bit de paridade para cada palavra, é calculada uma palavra completa adicional, da forma mostrada no exemplo seguinte:

caracter	ASCII	binário	bit paridade
‘R’	52H	01010010	1
‘E’	45H	01000101	1
‘D’	44H	01000100	0
‘E’	45H	01000101	1
BSC		00010110	1

A palavra BSC é enviada no final do quadro, após as palavras que fazem parte dos dados a enviar. O receptor, a exemplo da técnica de paridade simples, calcula o BSC dos dados a medida que são recebidos e compara o resultado com a última palavra enviada (que é o BSC calculado pelo emissor); se houver diferença, houve erro de transmissão. Nesta variante, erros que não foram detectados pela paridade simples provavelmente serão detectados pela paridade longitudinal. Esta técnica é largamente utilizada em protocolos de rede mais simples, mas tem uma eficiência limitada.

Um método de detecção de erros ainda melhor e largamente utilizado em redes locais é a definição de um código polinomial, também denominado *CRC* (*cyclic redundancy code*).

Nos códigos polinomiais, considera-se que os bits de uma cadeia de caracteres são os coeficientes de um polinômio, coeficientes estes, capazes de assumir apenas dois valores: 0 ou 1. Assim, um bloco de *k* bits é visto como uma série de coeficientes de um polinômio de *k* termos, indo de x^{k-1} a x^0. Por exemplo, a palavra 110001 contém 6 bits e representa o polinômio $x^5 + x^4 + 1$ (coeficientes contados da direita para a esquerda).

A utilização de códigos de detecção polinomiais é baseada na escolha de um código especial que caracteriza um *polinômio gerador G(x)*. Uma exigência em relação a este polinômio é que os bits mais significativo e menos significativo (correspondendo, respectivamente aos coeficientes de mais alta ordem e de mais baixa ordem do polinômio) sejam de valor 1. A técnica consiste em adicionar a um bloco de dados (caracterizando um polinômio *M(x)*) um conjunto de bits de controle de modo que o quadro (dados + bits de controle) seja divisível por *G(x)*. Na recepção, a entidade de Enlace efetua a divisão dos bits compondo o quadro pelo polinômio gerador. Caso o resto seja diferente de zero, é caracterizada então a ocorrência de um erro de transmissão.

O Algoritmo básico para o cálculo do CRC é o seguinte:

- Seja ***r*** o grau do polinômio gerador G(x); acrescentar ***r*** bits a zero no final da mensagem a enviar, obtendo o polinômio $x^r.M(x)$;
- executar a divisão em módulo-2 do polinômio resultante $x^r.M(x)$ por G(x);
- Subtrai em módulo-2 o resto da divisão acima do polinômio $x^r.M(x)$
- Resultado: quadro a transmitir com FCS

Vamos ilustrar o cálculo do CRC por meio de um exemplo:

- Seja o Frame de dados a transmitir: M(x) = 11100110
- Tomemos como polinômio gerador: G(x) = 11001 = x^4+x^3+1
- O Frame com acréscimo do zeros fica: $x^r.M(x)$ = 11100110 0000

A divisão do polinômio $x^r.M(x)$ por G(x) é ilustrada a seguir:

```
                                1  0  1  1  0  1  1  0
1  1  0  0  1 | 1  1  1  0  0  1  1  0  0  0  0  0
            ⊕ | 1  1  0  0  1  ↓
              | 0  0  1  0  1  1
              ⊕ | 0  0  0  0  0  ↓
                | 0  1  0  1  1  1
                ⊕ | 1  1  0  0  1  ↓
                  | 0  1  1  1  0  0
                  ⊕ | 1  1  0  0  1  ↓
                    | 0  0  1  0  1  0
```

$$
\begin{array}{cccccccc}
\oplus & 0 & 0 & 0 & 0 & 0 & \downarrow & \\
 & 0 & 1 & 0 & 1 & 0 & 0 & \\
 & \oplus & 1 & 1 & 0 & 0 & 1 & \downarrow \\
 & & 0 & 1 & 1 & 0 & 1 & 0 \\
 & & \oplus & 1 & 1 & 0 & 0 & 1 \\
 & & & 0 & 0 & 0 & 1 & 1 \\
 & & & \oplus & 0 & 0 & 0 & 0 \\
 & & & & & 0 & 1 & 1
\end{array}
\quad \downarrow \quad 0 \quad 0 \quad 0 \quad 0
$$

Para o exemplo acima, o resto R(x)= 0110 é a própria FCS. O Frame a transmitir é dado por $x^r.M(x) - R(x) = 11100110\ 0110$ (que corresponde ao frame original acrescido dos 4 bits de R(x)) .

Todas as operações descritas acima podem ser implementadas em hardware com registradores de deslocamento e portas lógicas XOR.

Alguns exemplos de polinômios geradores freqüentemente adotados na detecção de erros são:

CRC-12 = $x^{12} + x^{11} + x^3 + x^2 + x^1 + 1$

CRC-16 = $x^{16} + x^{15} + x^2 + 1$

CRC-CCITT = $x^{16} + x^{12} + x^5 + 1$

Tomando-se o polinômio gerador da CCITT como exemplo, verificamos que é um polinômio de ordem 16, de forma que serão acrescentados 16 bits de CRC em cada quadro.

Este polinômio é capaz de detectar:

- todos os erros de paridade
- todos os erros de 2 bits
- todos os pares de erros de 2 bits cada
- blocos de erros não excedendo 16 bits

Esta última técnica é bem mais eficiente que as anteriores, apesar de não ser infalível (*nenhuma* delas é). Observe que os *códigos detetores* de erro não corrigem as palavras erradas recebidas. Eles permitem apenas detectar a ocorrência do erro e o computador que recebe os dados deve requerer uma retransmissão da palavra de código errada, ao contrário do que ocorre com os *códigos corretores* !

2.3.3.4. Análise de alguns protocolos de enlace

Os protocolos que analisaremos aqui vão permitir detectar uma série de problemas relacionados ao controle de erros, que é a função mais importante a ser efetuada pela camada

de Enlace. A ordem de apresentação destes protocolos segue uma ordem de complexidade que vai crescendo em função de novas hipóteses que levantaremos ao longo da análise. Nesta análise, vamos considerar a comunicação entre entidades de comunicação situados em dois sistemas distantes, denominados aqui **A** e **B**.

No primeiro caso, vamos supor que o suporte de comunicação utilizado permite uma comunicação unidirecional apenas e que as camadas de Rede dos sistemas A e B estão sempre prontas a enviar e receber dados, respectivamente. Ainda, neste caso, o canal de comunicação é perfeito, ou seja, nenhum quadro é corrompido ou perdido. Assim, do lado do receptor, o único evento possível é a recepção de quadros corretos. A figura 2.3.3.9 ilustra a troca de quadros entre as entidades localizadas nos sistemas A e B. Esta troca é realizada numa única direção (de A para B), sem a ocorrência de perdas dada a consideração de que o canal de comunicação é perfeito.

O segundo caso, indo do imaginário ao mais realista, leva em conta o fato de que a camada de Rede do sistema receptor não tem capacidade de tratar os dados com a mesma velocidade em que eles são gerados pela camada de Rede emissora. Por outro lado, a hipótese de que o canal é perfeito e que os dados circulam numa única direção é ainda verdadeira.

O problema que se coloca, neste caso, é o de impedir que o emissor envie os quadros numa velocidade superior àquela que o receptor pode consumi-los (tratá-los e retransmiti-los). Uma forma de fazê-lo é estabelecer um mecanismo de temporização do lado do emissor de modo a provocar um pequeno espaço de espera pelo tratamento do dado previamente enviado antes do envio do seguinte. Esta solução não é, evidentemente, a melhor, uma vez que uma má escolha do limite de espera pode conduzir a atrasos indesejáveis na comunicação.

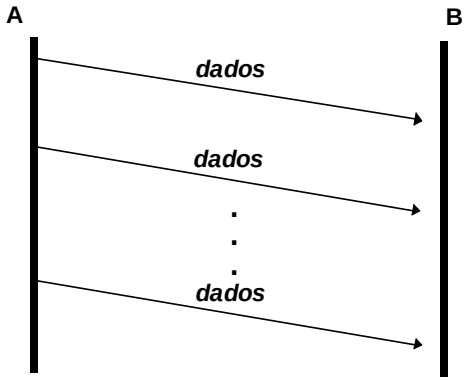


Figura 2.3.3.9 - Troca de quadros considerando um protocolo onde a comunicação é unidirecional e sem corrupção nem perda de quadros.

Uma forma mais eficiente de fazê-lo é determinar uma maneira de informar ao emissor, do estado corrente do receptor. Isto conduz à definição de quadros de reconhecimento que serão enviados pelo receptor, após o tratamento do dado recebido, para informar ao emissor que ele pode enviar o quadro de dados seguinte. Isto significa que, apesar da comunicação ser considerada unidirecional, o canal conectando as duas camadas de Enlace

deve permitir a comunicação bidirecional para permitir que a entidade de Enlace receptora emita os quadros de reconhecimento. Este protocolo, denominado *send-wait, stop-and-wait* ou *envia-espera*, é ilustrado pela figura 2.3.3.10.

O terceiro caso a ser analisado aqui torna-se muito mais próximo do real, uma vez que consideraremos aqui a "hipótese" (nem tão hipotética assim!) de que os dados possam chegar à outra extremidade do canal corrompidos ou que estes sejam perdidos ao longo do canal. Vamos, porém, considerar a possibilidade de detectar a ocorrência de quadros incorretos através de uma das técnicas apresentadas na seção 2.3.3.3.

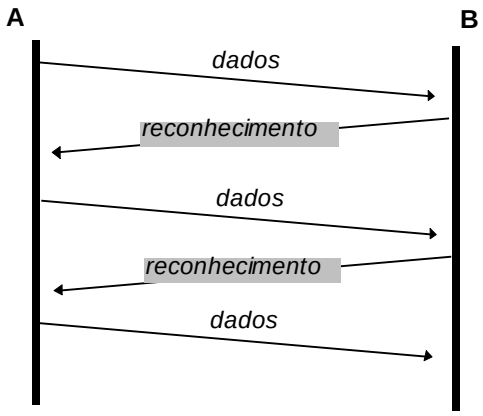


Figura 2.3.3.10 - Protocolo *envia-espera*.

Uma primeira solução é aquela em que o receptor emite um quadro de reconhecimento somente no caso em que o quadro de dados recebido esteja correto. Do lado do receptor, é estabelecido um mecanismo de temporização que é disparado após o envio de cada quadro de dados. Se após a expiração da temporização, um quadro de reconhecimento não é recebido, o emissor considera que o quadro de dados foi perdido ou que foi recebido incorretamente e o retransmite. A operação é repetida pelo número de vezes que sejam necessárias para que um quadro de reconhecimento seja, enfim recebido de B (figura 2.3.3.11).

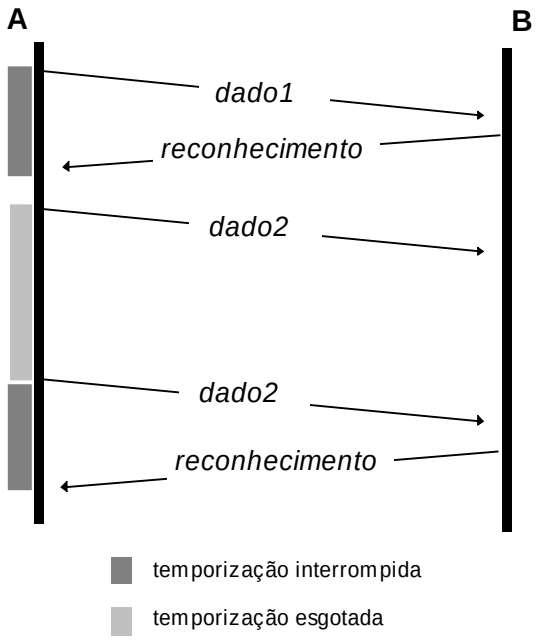


Figura 2.3.3.11 - Protocolo considerando controle de erro por quadro de reconhecimento com temporização.

Esta solução, porém, apresenta uma grave deficiência: no caso da entidade em B receber corretamente o dado e enviar o quadro de reconhecimento e este último for perdido no canal de comunicação, a entidade emissora, após a expiração da temporização, vai reemitir o quadro de dados, gerando um problema de duplicação de mensagem (figura 2.3.3.12).

Isto significa que o receptor deve ser capaz de reconhecer os quadros recebidos de modo a saber se estes são duplicados ou não. Para isto, será preciso introduzir um campo contendo um número de seqüência (um bit, no caso) do quadro de modo a permitir esta distinção. Se uma entidade receptora recebe sucessivamente dois quadros contendo o mesmo número de seqüência, o segundo será rejeitado.

Outro ponto importante a ser destacado no caso deste terceiro tipo de protocolo é a escolha do valor da temporização no emissor (para espera do reconhecimento). Uma temporização muito longa pode conduzir a atrasos indesejáveis na comunicação. Uma temporização excessivamente curta, por outro lado, pode conduzir a erros de protocolo consideráveis. Tomando como exemplo este caso, vamos supor que o emissor enviou um quadro, que este foi recebido corretamente pelo receptor que, por sua vez, emitiu um quadro de reconhecimento.

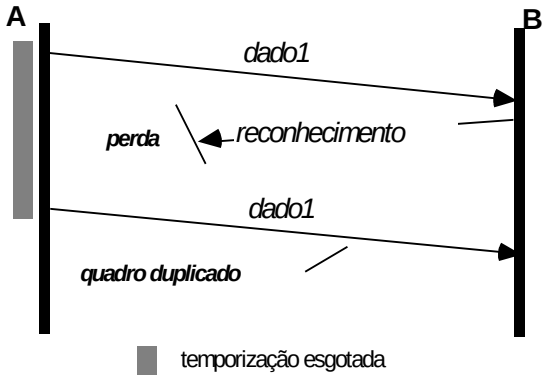


Figura 2.3.3.12 - Protocolo considerando controle de erro por quadro de reconhecimento com temporização, no caso de perda do quadro de reconhecimento (gerando duplicação de quadro).

Considerando que a temporização é excessivamente curta, pode ocorrer que esta seja expirada antes que o quadro de reconhecimento tenha sido integralmente transmitido ao emissor. Desta forma, o receptor entende que o quadro foi perdido ou transmitido incorretamente e retransmite o mesmo. Em seguida, o emissor recebe o quadro de reconhecimento daquele enviado inicialmente e, erroneamente, pensa que o reconhecimento é relativo àquele que ele havia enviado posteriormente. Entretanto, o emissor desconhece que um novo quadro de reconhecimento está a caminho (relativo ao quadro duplicado). Se, portanto o quadro seguinte é perdido e o emissor recebe um quadro de reconhecimento ele, desta vez, vai interpretar que o último quadro enviado foi corretamente recebido, caracterizando então um erro de protocolo. A figura 2.3.3.13 ilustra este problema.

Os protocolos analisados nesta seção, do tipo *envia-espera*, apesar dos problemas que podem gerar, são bastante utilizados, principalmente devido à sua relativa simplicidade de construção. Esta classe de protocolos, além disso, subutiliza o canal de comunicação, uma vez que eles devem esperar a recepção do quadro de reconhecimento para poderem enviar o quadro de dados seguinte.

Uma classe de protocolos mais eficiente é aquela em que o emissor pode enviar um certo número de quadros de dados sem que ele tenha recebido o reconhecimento dos quadros já emitidos. Isto vai requerer, obviamente, a implementação de um mecanismo de reconhecimento de quadros mais completo que aquele citado no caso do protocolo anterior. São os denominados protocolos *contínuos* ou de *largura de janela*.

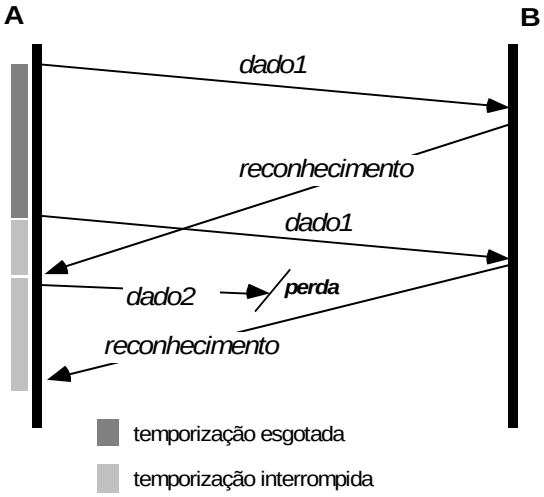


Figura 2.3.3.13 - Protocolo com quadro de reconhecimento no caso de temporização excessivamente curta.

Os protocolos de largura de janela também são chamados protocolos de "janela deslizante" ou "Continuous-RQ".

A idéia básica é numerar também os ACK. Em conseqüência da numeração dos ACK, o emissor não precisa esperar um ACK para cada quadro: ele pode enviar vários quadros e manter uma lista de retransmissão com um time-out para cada quadro. A lista de retransmissão opera de forma FIFO. O receptor retorna um ACK com o número do quadro recebido N(r). O receptor mantém uma lista de recepção, contendo os n últimos quadros recebidos sem erro (n é a "largura" da janela).

Quando o emissor recebe um ACK, deleta o quadro com número correspondente da lista de retransmissão. Este procedimento é ilustrado na figura 2.3.3.14.

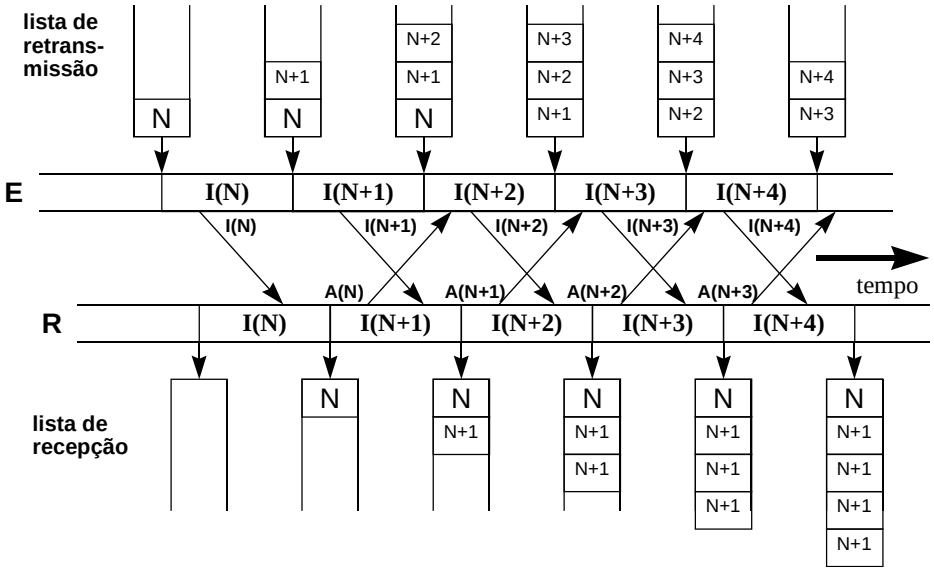


Figura 2.3.3.14 - Protocolo de janela deslizante

O Exemplo visto assume que não ocorrem erros. Duas estratégias podem ser adotadas em caso de erros:

- Retransmissão seletiva: emissor retransmite somente quadros cujos ACK não foram recebidos após time-out. Receptor tem que manter buffer com n (largura de janela) grande...
- Retorna-N (Go-Back-N): quando o receptor recebe um quadro fora da sequencia, pede ao emissor para reiniciar do último quadro recebido na ordem correta. Receptor mantém janela com largura 1 !

2.3.3.5. Exemplo de Protocolo de Enlace: o HDLC

HDLC (*High-level Data Link Control*) é uma família de protocolos orientados ao bit, resultado do esforço realizado pela ISO para a padronização da camada de enlace. O HDLC foi definido pela ISO a partir do protocolo SDLC (Synchronous Data Link and Control), da IBM. Existem atualmente diversos protocolos compatíveis com o HDLC: por exemplo, ADCCP (ANSI), LAP E LAPB (CCITT).

O protocolo HDLC prevê 2 modos de operação:

- NRM (Normal Response Mode): define uma estação mestra e várias escravas com topologia em barramento;
- ABM (Asynchronous Balanced Mode): operação multimestre para topologia ponto-a-ponto.

A forma de operação mais comum para a maioria das aplicações de HDLC é a utilização do modo NRM (mestre / escravos).

Um quadro HDLC apresenta a estrutura conforme mostrado na figura 2.3.3.15, cujos campos serão descritos a seguir. Os campos *flag* correspondem à seqüência 01111110, que permite delimitar o quadro. Todos os quadros HDLC iniciam e terminam com esta seqüência. No caso de envio de quadros consecutivos, um mesmo flag pode marcar o fim de um quadro e o início de outro. O campo *endereço*, de 8 bits, permite identificar a estação destinatária do quadro (no caso de comando) ou fonte (no caso de resposta). O campo *controle* permite identificar a função do quadro, assim como especificar os números de seqüência. A figura 2.3.3.16 mostra a configuração dos bits compondo este campo. O campo *informação* é aquele que contém os dados a serem transferidos via camada de Enlace. O campo *FCS* (*Frame Check Sequence*) contém a seqüência de detecção de erros utilizada. Esta seqüência é calculada sobre os campos endereço, controle e informação, utilizando como polinômio gerador a combinação $x^{16} + x^{12} + x^5 + 1$ (CRC-CCITT).

Um quadro de enlace é considerado incorreto e, assim, ignorado pelo destinatário nas seguintes condições:

- se não estiver delimitado pelos flags;
- se o comprimento for inferior a 32 bits entre flags;
- se contiver mais de sete 1's consecutivos.

O protocolo HDLC prevê três tipos de quadro: de *informação*, de *supervisão* e *não numerados*, diferenciados pelo *campo de controle*, como mostra a figura 2.3.3.15. Os bits 1 e 2 deste campo permitem identificar o tipo de quadro: se bit 1 = 0 indica um quadro de informação; se bit 1 = 1, pode identificar tanto um *quadro de supervisão* (bit 2 = 0) ou um *quadro não numerado* (bit 2 = 1). Os quadros de *supervisão* são utilizados principalmente para controle de fluxo (indicar se receptor está pronto ou não para receber dados, rejeição de quadros com defeito, quadro de reconhecimento), enquanto os quadros *não numerados* são usados principalmente para o estabelecimento e o término de conexões.

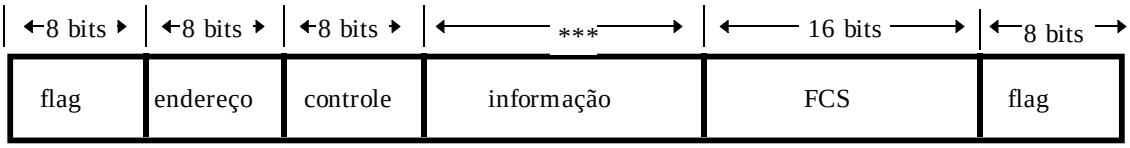


Figura 2.3.3.15 - Estrutura de um quadro HDLC.

bits	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
informação	**0**	**N(s)**			**P/F**	**N(r)**		
supervisão	**1**	**0**	**S**	**S**	**P/F**	**N(r)**		
não numerado	**1**	**1**	**M**	**M**	**P/F**	**M**	**M**	**M**

Figura 2.3.3.16 - Campo *controle* dos quadros HDLC.

O campo *N(s)* permite definir o número de seqüência do quadro e o campo *N(r)* para o reconhecimento de quadros (indica o número do próximo quadro esperado) — N(s) e N(r) correspondem, de fato, aos campos *seq* e *ack* mostrados anteriormente. O bit *P/F* depende do modo de operação do protocolo. No primeiro modo de operação, *resposta normal (NRM)*, o bit *P* a 1 indica que a estação emissora deseja consultar uma secundária. Desta forma, o último quadro de uma seqüência de quadros de resposta enviados por uma estação secundária conterá o bit *F* a 1. No segundo modo, *resposta assíncrona e balanceada (ABM)*, o

recebimento de um quadro com o bit *P* a 1 indica que a estação secundária deve enviar o próximo quadro com o bit *F* também a 1.

Os bits *S* são utilizados para identificar comandos de *supervisão*, particularmente para comandos de reconhecimento e de controle de fluxo. A tabela abaixo mostra as possíveis combinações dos bits *S* e os comandos / respostas associados.

S	**S**	**Código**	**Comando/Resposta**
0	0	RR	*Receiver Ready (Receptor Pronto)*
1	0	RNR	*Receiver Not Ready (Receptor Não Pronto)*
0	1	REJ	*Reject (Rejeitado)*
1	1	SREJ	*Selective Reject (Recusa Seletiva)*

Os bits *M* são utilizados para identificar comandos e respostas *não numeradas*. Eles permitem, nos quadros sem numeração, representar 32 diferentes comandos e respostas, entre os quais *DISC* para terminar uma conexão de enlace previamente estabelecida, *FRMR* para indicar a rejeição de um quadro, *SNRM* (Set Normal Response Mode, Connect), *SABM* (Set Asynchronous Balanced Mode), *UA* (Unnumbered ACK), *UP* (Unnumbered Poll, usado pelo mestre em NRM), etc...

A operação do protocolo pode ser ilustrada como segue:

- Inicio de operação:
  - ajuste do modo de operação e estabelecimento de uma conexão (frame SNRM ou SABM)
  - parceiro responde com frame UA
- Troca de Dados:
  - em NRM, escravo só envia dados a pedido do mestre: mestre envia UP, escravo responde com I-Frame (se tem dados) ou RNR (se não tem dados)
  - em ABM, operação full-duplex. Estação envia I-Frame, receptor responde com RR (ACK), REJ (NAK) ou SREJ (NAK)
- Fim conexão:
  - em NRM só mestre pode concluir conexão (frame DISC); escravo responde com frame UA.
  - em ABM, qualquer lado pode enviar DISC; outro responde com UA.

2.3.3.6. Padrão IEEE 802 para a Camada de Enlace

O IEEE definiu um padrão para a camada de enlace (norma IEEE 802.2), hoje largamente difundido. Estudaremos a norma IEEE 802 mais detalhadamente em um capítulo posterior. Aqui nos limitaremos a descrever brevemente as primitivas propostas nesta norma.

- Primitivas da Interface MAC / LLC:

    - M_DATA.request (local address, remote address, user data, service class): pedido originado da subcamada LLC para a subcamada MAC solicitando o envio de um frame de dados entre estações;
    - M_DATA.confirm (status): confirmação de envio de um frame retornada pela subcamada MAC à subcamada LLC (valor local, não é ACK do receptor);
    - M_DATA.indication (local address, remote address, user data, service class): indica para a subcamada LLC do receptor a chegada de um frame de dados de uma estação remota.

Na proposta IEEE 802, a subcamada MAC opera somente com serviços sem conexão e sem reconhecimento. Os serviços oferecidos pela subcamada LLC ao usuário (ou à camada de rede) podem ser com ou sem conexão e são apresentados na tabela a seguir:

SERVIÇOS ORIENTADOS À CONEXÃO
L_CONNECT.request (local address, remote address, service class) *L_CONNECT.indicationt (local address, remote address, status, service class)* *L_CONNECT.response (local address, remote address, service class)* *L_CONNECT.confirm (local address, remote address, status, service class)*
L_DISCONNECT.request (local address, remote address) *L_DISCONNECT.indication (local address, remote address, reason)* *L_DISCONNECT.response (local address, remote address)* *L_DISCONNECT.confirm (local address, remote address, status)*
L_DATA_CONNECT.request (local address, remote address, user_data) *L_DATA_CONNECT.indication (local address, remote address , user_data)* *L_DATA_CONNECT.response (local address, remote address)* *L_DATA_CONNECT.confirm (local address, remote address¸ status)*

L_RESET.request (local address, remote address) *L_RESET.indication (local address, remote address, reason)* *L_RESET.response (local address, remote address)* *L_RESET.confirm (local address, remote address, status)*
L_CONNECTION_FLOWCONTROL.request (local address, remote address, amount of data) *L_CONNECTION_FLOWCONTROL.indication (local address, remote address, amount of data)*
SERVIÇOS SEM CONEXÃO
L_DATA.request (local address, remote address, user_data, service class) *L_DATA.indication (local address, remote address, user_data, service class)*

2.3.4. A CAMADA DE REDE

O objetivo da camada de Rede é assegurar o transporte de unidades de dados, denominadas *pacotes*, do sistema fonte ao sistema destinatário, definindo uma trajetória apropriada. Esta trajetória pode significar a passagem por diversos nós intermediários da rede, o que significa que a camada de Rede deve ter o conhecimento de todos os aspectos topológicos da rede considerada e, com esta informação, ser capaz de escolher o caminho a ser traçado pelas mensagens (*Roteamento*). Nesta escolha, é interessante que seja levado em conta o estado corrente da rede, particularmente no que diz respeito ao tráfego das mensagens, evitando assim a sobrecarga de certos trechos das linhas de comunicação.

Ainda, se o sistemas fonte e destinatário estão conectados a redes diferentes, estas diferenças devem ser levadas em conta e compensadas pela camada de Rede.

Duas funções essenciais da camada de Rede, descritas brevemente aqui, refletem, respectivamente, os problemas de *roteamento* e *congestionamento*, serão tratados nesta parte do documento. Os mecanismos relacionados às funções de *interconexão de redes* merecerão uma parte dedicada mais adiante.

2.3.4.1. Serviços oferecidos pela camada de Rede

a) Serviços orientados à conexão e sem conexão

Uma das primeiras discussões realizadas no que diz respeito à concepção da camada de Rede foi a questão do tipo de serviço a ser oferecido às camadas superiores, particularmente relacionado à existência ou não de conexão.

Aqueles que defendiam a proposta de um serviço sem conexão, estabeleciam que a função desta camada era garantir o transporte dos pacotes e nada mais que isso, as outras funções como o controle de erro, controle de fluxo, etc, devendo ficar a cargo das camadas superiores da arquitetura de comunicação. Desta forma, a camada de Rede deveria, então, oferecer unicamente dois tipos de primitivas de serviço: *SEND PACKET* e *RECEIVE PACKET*.

Por outro lado, outro grupo defendia a proposta de um serviço confiável, orientado à conexão, com comunicação bidirecional.

O resultado desta discussão foi a definição, a nível do modelo OSI, de duas classes de serviço, sem conexão e orientado à conexão, sendo que uma certa liberdade foi dada no sentido de se definir a que nível o serviço com conexão seria implantado, ou mesmo se este seria implantado. Abriu-se, assim, a possibilidade de se ter, nos diferentes níveis, as duas classes de serviço, como ilustrado pela figura 2.3.24.

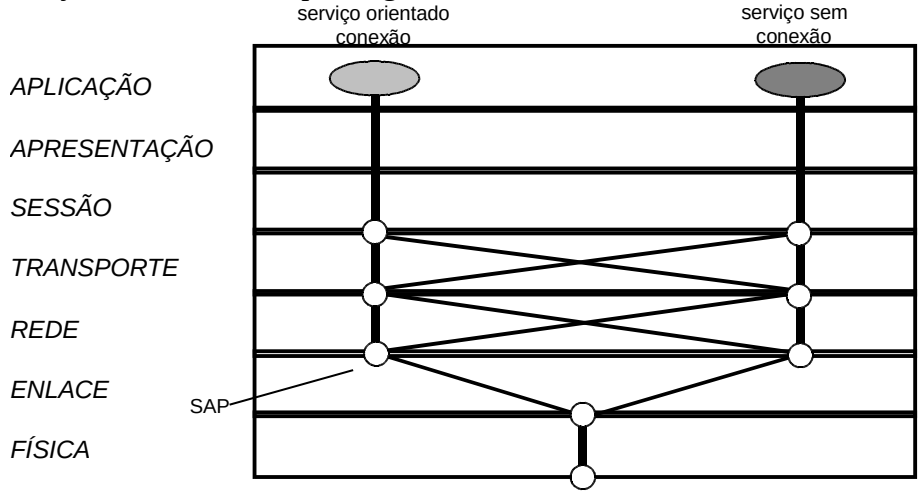


Figura 2.3.24 - Ilustração das classes de serviço do modelo OSI.

Os SAPs são localizados na interface entre duas camadas, permitindo que uma camada tenha acesso aos serviços oferecidos pela camada inferior. Como se pode ver na figura 2.3.24, da camada de Enlace para cima, os serviços podem ser sem ou com conexão.

Pode-se ter, desde o nível Aplicação, um serviço totalmente orientado à conexão ou, de maneira oposta, sem conexão. Ainda, é possível ter-se num dado nível, um serviço orientado conexão, mesmo se os serviços oferecidos pelas camadas inferiores são sem conexão. O inverso também é verdadeiro, embora não seja uma escolha das mais interessantes (implantar serviços sem conexão sobre redes oferecendo serviços orientados à conexão).

Uma conexão de Rede é vista, da ótica do modelo OSI, como um par de filas *FIFO* (*FIrst In FIrst Out*), cada uma orientada num sentido, conectado entre dois *NSAPs* (endereços de rede). A figura 2.3.25 ilustra o estado da conexão, considerando a adoção de um serviço orientado à conexão. A figura 2.3.25(a) ilustra o estado da conexão antes do seu

estabelecimento; 2.3.25(b) após o estabelecimento da conexão e, 2.3.25(c), após o envio de três pacotes de dados.

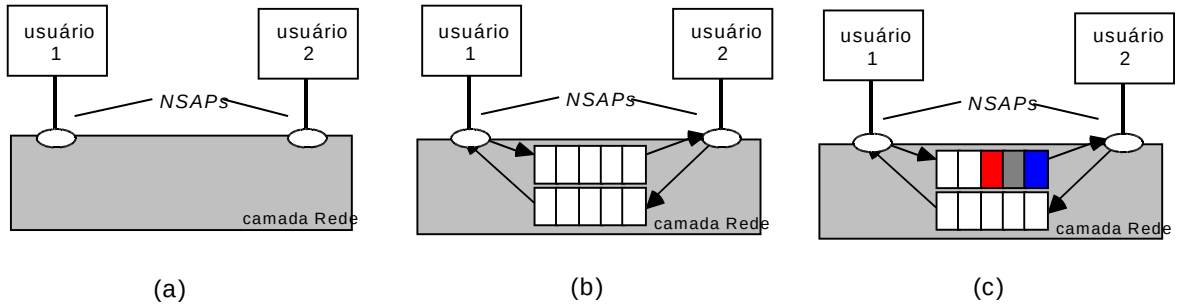


Figura 2.3.25 - Modelo de uma conexão de Rede: (a) antes do estabelecimento; (b) após o estabelecimento; (c) após o envio de três pacotes (de 1 para 2).

b) As primitivas de serviço

A tabela a seguir apresenta as primitivas de serviço disponíveis pela camada de Rede do modelo OSI, tanto para os serviços orientados à conexão como para os sem conexão.

SERVIÇO ORIENTADO À CONEXÃO
N_CONNECT.request(called, calling,acks_wanted,exp_wanted,qos,user_data) *N_CONNECT.indication(called,calling,acks_wanted,exp_wanted,qos,user_data)* *N_CONNECT.response(responder,acks_wanted,exp_wanted,qos,user_data)* *N_CONNECT.confirm(responder,acks_wanted,exp_wanted,qos,user_data)*
N_DISCONNECT.request (source, reason, user_data, responding_address) *N_DISCONNECT.indication (source, reason, user_data, responding_address)*
N_DATA.request (user_data) *N_DATA.indication (user_data)* *N_DATA_ACKNOWLEDGE.request ( )* *N_DATA_ACKNOWLEDGE.indication ( )* *N_EXPEDITED_DATA.request (user_data)* *N_EXPEDITED_DATA.indication (user_data)*
N_RESET.request (source, reason) *N_RESET.indication (source, reason)* *N_RESET.response ( )* *N_RESET.confirm ( )*
SERVIÇO SEM CONEXÃO
N_UNITDATA.request (source_address,dest_address, qos, user_data) *N_UNITDATA.indication (source_address,dest_address, qos, user_data)*

N_FACILITY.request (qos) *N_FACILITY.indication (dest_address, qos, reason)* *N_REPORT.indication (dest_address, qos, reason)*

N_CONNECT.request é a primitiva utilizada para requisitar o estabelecimento de uma conexão de Rede e através de seus parâmetros deve-se indicar o endereço ao qual se deseja conectar (*calling*), assim como o endereço do iniciador (*called*). Encontra-se, ainda, nos seus parâmetros, duas variáveis booleanas que permitem requisitar serviços adicionais.

O parâmetro *acks_wanted* permite indicar o pedido de reconhecimento de pacotes. Se a camada de Rede não permite o fornecimento de reconhecimento, o fornecedor vai colocar esta variável a *falso* na primitiva de indicação; da mesma forma, se a entidade destinatária não pode fornecer reconhecimento, ela mesma o fará (colocar *acks_wanted* a *falso*) na primitiva de resposta.

O outro parâmetro booleano, *exp_wanted*, permite a utilização de serviços de dados expressos (ou urgentes), o que significa que um pacote pode violar a ordem normal dos pacotes na fila para se colocar na cabeça desta. Isto permite, de certo modo, estabelecer um nível de prioridade entre as mensagens a nível de Rede.

O parâmetro *qos* (quality of service) permite determinar a qualidade do serviço a ser oferecido, sendo composto de duas listas de valores. A primeira lista contém o contexto desejado pela entidade iniciadora; a segunda indica os valores mínimos aceitáveis. Se o serviço de Rede é incapaz de fornecer pelo menos os valores mínimos estabelecidos pelo usuário iniciador, a conexão não será estabelecida. Os valores estabelecidos neste parâmetro são a taxa de erro, a taxa de transmissão, a confidencialidade e custo da transmissão.

O iniciador pode, através do parâmetro *user_data*, introduzir dados no pedido de conexão, podendo a entidade chamada consultar estes dados antes da aceitação da conexão.

A aceitação de uma conexão é feita pelo chamador através da primitiva *N_CONNECT.indication*; por outro lado, a recusa é implementada através da primitiva *N_DISCONNECT.request*, que informa, através do parâmetro *reason*, o motivo do não estabelecimento da conexão.

Se a conexão é estabelecida, a transmissão de dados é feita utilizando a primitiva *N_DATA.request*, que será refletida no receptor por uma primitiva *N_DATA.indication*. Caso um pedido de reconhecimento de pacotes foi acertado entre os usuários da camada no momento da conexão, o reconhecimento será implementado através do envio, pelo receptor, de uma primitiva *N_DATA_ACKNOWLEDGE.request*.

As primitivas *N_RESET* são utilizadas para sinalizar problemas de comunicação, tais como o bloqueio de uma entidade de transporte envolvida na comunicação ou mesmo do fornecedor do serviço. O efeito deste serviço é o esvaziamento das filas de espera, sendo que

as informações presentes nestas serão perdidas. As perdas deverão ser recuperadas pela camada de Transporte.

As primitivas *N_UNITDATA* são utilizadas para o envio de dados no serviço sem conexão; elas não oferecem nem controle de erros nem de fluxo. *N_FACILITY* é o serviço que permite ao usuário obter informações sobre as características da transmissão de informação, por exemplo, a percentagem de pacotes distribuídos.

Finalmente, a primitiva *N_REPORT* permite à camada de Rede informar a ocorrência de problemas relativos ao serviço de Rede, como, por exemplo, a indisponibilidade momentânea de um determinado endereço destinatário.

c) O endereçamento de Rede

Uma função importante desta camada é o fornecimento de uma codificação espacial de endereços coerente para uso da camada de Transporte. O fato é que, para cada rede, foi definida uma estrutura distinta de endereçamento, o trabalho de uniformização ficando a cargo da camada de Rede.

De uma forma geral, 2 esquemas básicos de endereçamento são possíveis:

- **endereçamento hierárquico**: o endereço é constituído conforme a posição de cada entidade na hierarquia da rede, sugerindo o local onde esta se encontra. Este esquema de endereçamento é usado em WANs e MANs (ex.: Internet). Esta técnica facilita o roteamento.
- **endereçamento horizontal**: aqui, o endereço não tem relação com localização da entidade na rede. Este esquema é usado em LANs (ex.: padrão IEEE 802). Este tipo de técnica facilita a reconfiguração da rede sem necessitar alterar os endereços das estações.

No modelo OSI, a estrutura de endereçamento a nível de Rede foi concebida de modo a incorporar as características das diversas estruturas de endereçamento existentes. Todas as primitivas de serviço de Rede utilizam o endereçamento orientado aos NSAPs para identificar a origem ou o destinatário de um pacote. A figura 2.3.26 mostra o formato do endereço de NSAP, que é composto de três campos:

- *AFI* (*Authority and Format Identifier*), identifica o tipo de endereçamento existente no terceiro campo do endereço, possibilitando uma numeração entre 10 e 99, correspondente aos diferentes formatos existentes e deixando ainda possibilidades de extensão;
- *IDI* (*Initial Domain Identifier*) indica o domínio ao qual pertence o número do *DSP* (o terceiro campo) — se o *DSP* é um número de telefone, este campo indicará o código do país;

- *DSP* (*Domain Specific Part*) contém o endereço específico do NSAP no domínio considerado.

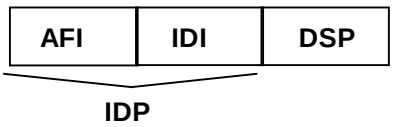


Figura 2.3.26 - Endereçamento de um NSAP.

2.3.4.2. A função de Roteamento

Como dito na parte introdutória, a função principal da camada de Rede é efetuar o encaminhamento dos pacotes trocados entre duas entidades oferecendo uma comunicação fim-a-fim. Durante a trajetória os pacotes sofrerão uma série de saltos, sendo que a decisão de que caminho utilizar é feita a nível da camada de Rede. Esta decisão pode levar em conta (ou não) a situação da rede do ponto de vista do tráfego de informação.

Num primeiro ponto, dado este último aspecto, pode-se distinguir os diferentes algoritmos de roteamento em duas principais classes: os algoritmos *adaptativos* e *não adaptativos*. Os algoritmos não adaptativos não levam em conta a situação de tráfego da rede, fazendo o denominado *roteamento estático*; já os adaptativos o fazem considerando modificações de topologia da rede e do tráfego real.

As seções que seguem vão discutir alguns dos conjuntos de algoritmos de roteamento existentes.

a) Algoritmo do caminho mais curto

Este algoritmo é baseado numa representação da subrede na forma de um grafo, onde os nós são os *IMPs* e os arcos são as linhas de comunicação. A escolha de uma trajetória é, então, baseada neste grafo, o objetivo sendo encontrar o caminho mais curto entre dois IMPs.

O conceito de caminho mais curto pode levar em conta diferentes aspectos:

- o número de nós entre os dois pontos;
- a distância geográfica entre os pontos;
- os tempos de espera em cada nó da trajetória, etc...

No terceiro caso, por exemplo, os arcos são etiquetados com um valor que representa o tempo médio de espera entre o envio e a recepção de um pacote de teste, expedido periodicamente.

Um exemplo de algoritmo do caminho mais curto é aquele definido por Dijkstra, onde cada nó é etiquetado pela distância do nó fonte seguindo o caminho mais curto conhecido. Como, inicialmente, nenhum caminho é conhecido, os nós são etiquetados com «infinito», ou "1". Estas vão sendo atualizadas à medida que o algoritmo progride e que os caminhos vão se tornando conhecidos, as etiquetas vão indo do provisório ao permanente uma vez que o caminho mais curto foi encontrado para os diferentes nós.

A figura 2.3.27 ilustra o funcionamento deste algoritmo. Vamos considerar o grafo apresentado em 2.3.27(a), onde os arcos são etiquetados com o valor da distância entre os nós. Vamos considerar que o objetivo aqui é encontrar o caminho mais curto entre *A* e *D*.

Como o nó *A* é o nó de origem, ele será marcado por um nó de referência (fundo cinza) e serão analisados os nós adjacentes. Estes serão etiquetados pela distância que os separa de *A*, marcando também, na etiqueta, o último nó a partir do qual o cálculo foi feito. Após a análise dos nós adjacentes a *A*, marca-se aquele contendo a etiqueta de menor valor, como mostrado em 2.3.27(b), este nó passando a ser o nó ativo (ou de referência).

Repetindo a análise para o nó *B*, define-se o nó *E* como sendo o novo nó ativo, mostrado em 2.3.27(c). O algoritmo vai progredindo até que o caminho mais curto seja então encontrado, as etiquetas sendo modificadas segundo as análises efetuadas. A progressão do algoritmo para o exemplo é ilustrada por 2.3.27(d), 2.3.27(e) e 2.3.27(f).

Para este exemplo, o caminho mais curto fica sendo pelas estações A-B-E-F-H-D, sendo A a origem e D o destino final.

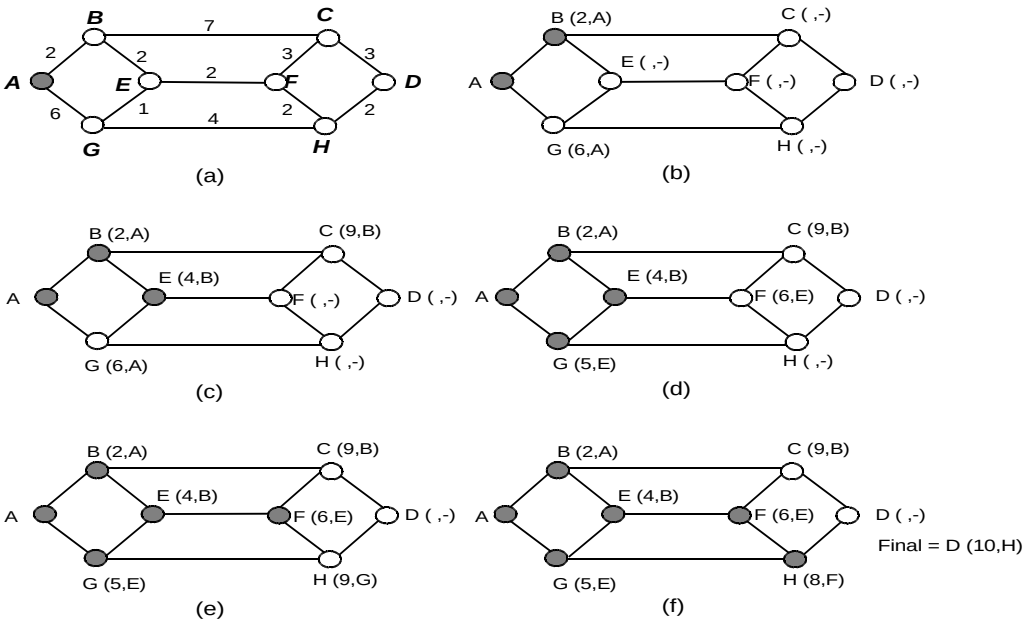


Figura 2.3.27 - Ilustração do algoritmo de Dijkstra.

O algoritmo do caminho mais curto pode ser executado localmente em cada nó, partindo de uma informação fixa sobre os custos de cada linha (roteamento estático distribuído).

O algoritmo pode também ser executado por uma estação encarregada de definir rotas entre todos os demais nós. Esta estação é chamada RCC (Routing Control Center). A RCC recebe novos dados sobre custo de certas linhas cada vez que uma mudança ocorre (roteamento dinâmico centralizado). O uso da RCC pode causar sobrecarga na rede se ocorrem alterações muito freqüentes nos custos de cada linha, pois a RCC tem que receber muitas mensagens contendo novos dados sobre custos das linhas afetadas.

b) Roteamento multicaminhos

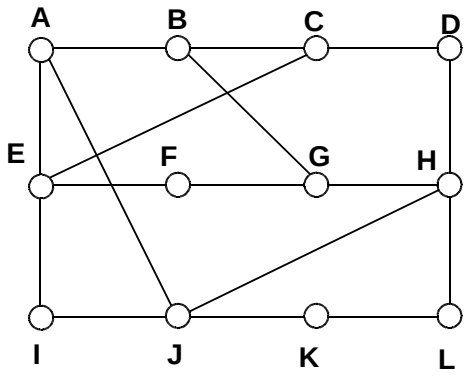
Este algoritmo leva em conta a possibilidade de existência de diversos caminhos entre dois nós de valores quase equivalentes. Desta forma, o desempenho da rede pode ser melhorado, se o tráfego é assumido por vários caminhos, reduzindo assim a carga de uma dada linha de comunicação.

Esta técnica de roteamento é implantada da seguinte forma: cada *IMP* mantém atualizada uma tabela com uma linha para cada IMP da rede. Para cada IMP, tem-se as diferentes linhas de saída para este destino, classificadas em ordem decrescente, do melhor ao menos eficiente, com um peso relativo.

Antes do envio de um pacote, o IMP gera um número aleatório para definir o caminho, utilizando os pesos como probabilidade. As tabelas são criadas de maneira estática pelo administrador do sistema e carregadas em cada IMP na inicialização da rede.

O grafo da figura 2.3.28(a) ilustra o desenvolvimento deste algoritmo, sendo que em 2.3.28(b) está apresentada a tabela para o IMP *J*. Se *J* recebe um pacote destinado a *A*, ele pode optar por um dos três caminhos, consultando a tabela na linha associada ao nó *A*.

A primeira escolha é o caminho direto a *A*, as outras sendo via *I* e *H*, respectivamente. A decisão é, então, baseada na geração de um número aleatório entre 0,00 e 0,99. Se o número é inferior a 0,63, a linha *A* será escolhida, se estiver entre 0,63 e 0,83, a linha passando por *I* será escolhida; senão, será a linha que passa por *H*.



(a)

A	A	0,63	I	0,21	H	0,16
B	A	0,46	H	0,31	I	0,23
C	A	0,34	I	0,33	H	0,33
D	H	0,50	A	0,25	I	0,25
E	A	0,40	I	0,40	H	0,20
F	A	0,34	H	0,33	I	0,33
G	H	0,46	A	0,31	K	0,23
H	H	0,63	K	0,21	A	0,16
I	I	0,65	A	0,22	H	0,13
.	.	.	.	.	.	.
K	K	0,67	H	0,22	A	0,11
L	K	0,42	H	0,42	A	0,16

(b)

Figura 2.3.28 - Ilustração do algoritmo multicaminho: (a) grafo da rede; (b) tabela p/ nó J.

A vantagem desta técnica sobre a anterior é a possibilidade de definir diferentes classes de tráfego sobre diferentes caminhos. Uma outra vantagem é a confiabilidade, uma vez que várias linhas podem ser perdidas sem que a rede perca a sua conectividade.

c) Roteamento dinâmico distribuído

Nesta técnica, cada estação troca periodicamente informações de roteamento com suas vizinhas. Cada estação envia aos seus vizinhos imediatos uma tabela contendo informação sobre custo de transmissão a partir dela para cada uma das demais estações da rede. Esta tabela contém, para cada destino possível, o nó preferencial de saída e o custo estimado de transmissão por este nó. Este tipo de roteamento também é conhecido como Roteamento por Vetor de Distancia (Distance Vector Routing).

Cada nó tem que conhecer o custo de transmissão para cada um dos seus vizinhos imediatos. Para decidir a rota, a estação emissora soma o custo de transmissão até o vizinho imediato com custo estimado dali até destino final.

A operação deste tipo de algoritmo é ilustrada na figura 2.3.29.

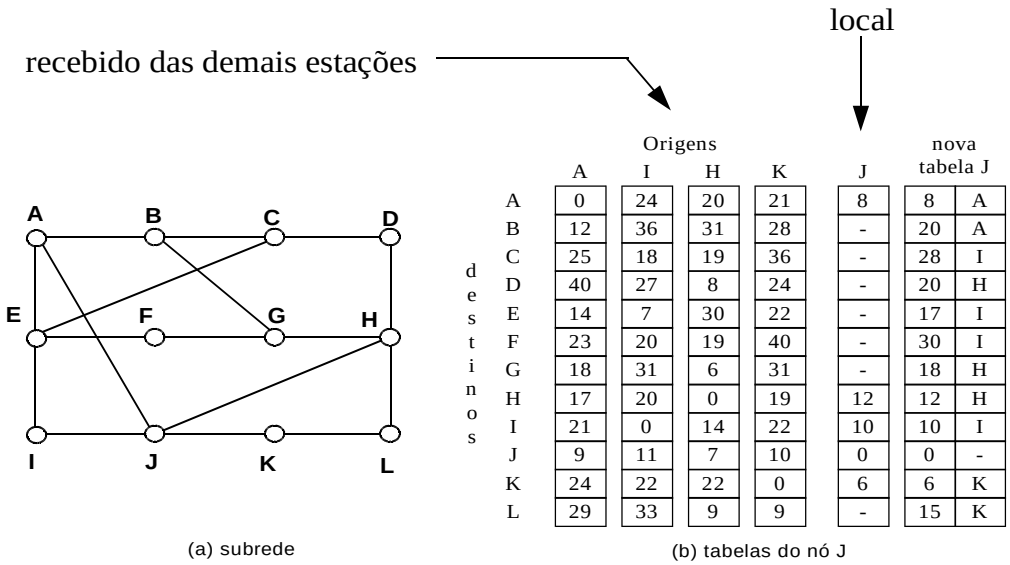


Figura 2.3.29 - Roteamento dinâmico distribuído

No exemplo acima, uma mensagem do nó J para o nó B será roteada através do nó A, enquanto uma mensagem destinada ao nó C passará pelo nó I (ver última tabela a direita, na figura 2.3.39). Vale lembrar que esta escolha de rota pode mudar, pois o roteamento é dinâmico, isto é, as tabelas são periodicamente atualizadas. Um algoritmo baseado neste principio, denominado RIP (Routing Information Protocol) foi implementado originalmente na ARPANET e posteriormente na Internet (parte do protocolo IP até 1990, quando foi substituido pelo protocolo OSPF, Open Shortest Path First), e na rede novell (protocolo IPX).

2.3.4.3. O controle de congestionamento

Durante o funcionamento de uma aplicação distribuída construída sobre uma rede, vão existir instantes em que o fluxo de mensagens sendo trocadas pode atingir valores bastante importantes, de tal forma que os nós intermediários, responsáveis do tratamento dos pacotes, não sejam mais capazes de tratar os pacotes para retransmissão. Isto, naturalmente, vai ter como conseqüência uma degradação no funcionamento da rede, podendo trazer prejuízos (lentidão, perdas de pacotes) ao desempenho da aplicação e comprometendo o seu correto funcionamento. As causas desta sobrecarga, conhecida por *congestionamento*, podem ser de várias naturezas. Um exemplo disto pode ser a lentidão dos nós na realização do roteamento ou um mau funcionamento do mecanismos de controle de fluxo.

O congestionamento consiste, normalmente de um processo a realimentação positiva, o número de mensagens tendendo a crescer se a rede está congestionada.

Sendo assim, a camada de Rede deve também fazer este papel, através da implementação de funções de controle de congestionamento, alguns dos quais serão descritos a seguir.

a) Pré-alocação de buffers

Uma primeira forma de controlar o congestionamento da rede é através da *pré-alocação de buffers*, particularmente se o serviço é orientado à conexão. Isto significa que, no momento do estabelecimento do circuito virtual que vai caracterizar a conexão, um determinado número de buffers deve ser alocado em cada nó para permitir o armazenamento dos pacotes a serem retransmitidos. Evidentemente, o número de buffers a ser alocado vai depender do protocolo implementado entre cada par de nós intermediários (IMPs). Um algoritmo do tipo "envia-espera" vai exigir um número de buffers evidentemente menor do que um algoritmo que autorize o envio de diversos pacotes antes da retransmissão.

b) A destruição de pacotes

Um outro mecanismo que é adotado para o controle de congestão é o da *destruição de pacotes*. Neste caso, não existe reserva prévia de buffers, de modo que, se um pacote chega num IMP e este não dispõe de buffer para o seu armazenamento, este é simplesmente destruído (ou descartado). Se o serviço oferecido é do tipo "datagrama", não há mais nada a fazer; por outro lado, se este é orientado à conexão, o pacote deverá ser armazenado em algum nó para uma possível retransmissão. Ainda, a destruição de pacotes deve seguir uma certa disciplina: por exemplo, destruir um pacote de reconhecimento pode não ser uma boa solução, uma vez que este pacote poderia permitir ao nó o apagamento de um pacote de informação e, por conseqüência, a liberação de um buffer. Uma solução para isto é a reserva, para cada linha de chegada, de um buffer que possibilite a recepção de pacotes de reconhecimento endereçados àquele nó.

c) O controle de fluxo

O *controle de fluxo* consiste em outra técnica de controle de congestionamento, embora não muito eficiente nesta tarefa. O problema do controle do fluxo é o fato que os limites do tráfego não podem ser estabelecidos a valores muito baixos, pois isto pode provocar problemas de eficiência na aplicação se um pico de tráfego é requerido. Por outro lado, a escolha de um limite alto de tráfego pode resultar num controle medíocre do congestionamento.

d) outras técnicas

Outras técnicas de controle de congestão podem ser ainda implementadas, como por exemplo:

- o *controle isarítmico*, baseado na existência em cada nó de um certo número de fichas. O nó que tiver um pacote a transmitir, deve obter uma ficha, se existir alguma disponível. Isto permite manter constante o número de pacotes em circulação na rede;

- os *pacotes de estrangulamento*, enviados por um nó ao usuário do serviço de rede, indicando que determinadas linhas de saída estão no limite da saturação. Isto faz com que o usuário reduza o envio de pacotes para o destino utilizando aquela linha até que a situação retome a normalidade.

2.3.4.4. Exemplos de Protocolos de Rede

a) Protocolo X.25 PLP

O protocolo X.25 PLP (*Packet Layer Protocol*) é o protocolo de nível 3 largamente adotado no modelo OSI fazendo parte do padrão X.25. O X.25 é o resultado dos esforços efetuados a partir de 1974 pelo CCITT para a definição de um padrão de comunicação para os níveis Físico, Enlace e Rede.

O X.25 define a interface entre um *DTE* (*Data Terminal Equipment*) e um *DCE* (*Data Circuit-terminating Equipment*). Um nó intermediário da subrede (um IMP, na terminologia OSI) é definido aqui como um *DSE* (*Data Switching Exchange*).

O padrão X.25 define o formato e o significado da informação trocada através da interface DTE-DCE para as camadas 1, 2 e 3. No que diz respeito ao nível 1, X.25 faz referência a outros padrões, no caso o *X.21* para as transmissões digitais e *X.21bis*, uma versão «analógica» do X.21, bastante similar a *RS-232-C*.

Na camada 2, X.25 define protocolos de enlace com o objetivo de oferecer um meio de transmissão confiável, definindo protocolos como LAP e LAPB.

No nível de Rede, X.25 implementa as conexões entre dois DTEs (figura 2.3.29), oferecendo dois tipos de conexão:

- a *chamada virtual* (*virtual calls*), que é similar a uma chamada telefônica comum, a qual é caracterizada pelo estabelecimento de uma conexão, a troca de dados e a liberação de conexão;
- o *circuito virtual permanente* (*permanent virtual circuits*), que funciona como uma linha especializada, disponível a todo momento, de modo que basta aos DTEs efetuar a transferência dos dados (uma conexão permanente é estabelecida na inicialização).

No caso das chamadas virtuais, como mostra a figura 2.3.30, quando um DTE quer dialogar com um outro DTE, ele deve, inicialmente, estabelecer a conexão; ele constrói, então, um pacote *CALL REQUEST* enviando-o ao DCE. Em seguida, a subrede encaminha o pacote ao DCE destinatário que vai passá-lo ao DTE correspondente. Se a conexão pode ser estabelecida, o DTE destinatário vai enviar um pacote *CALL ACCEPTED* indicando a aceitação da chamada.

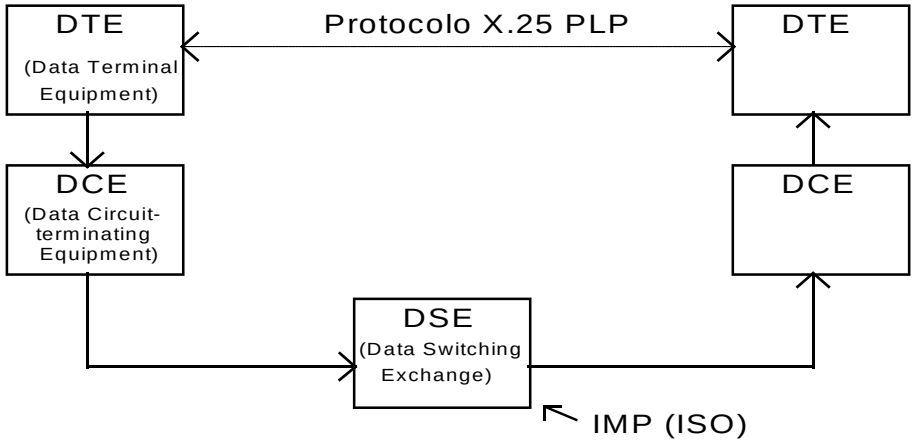


Figura 2.3.29 - O protocolo de rede X.25 PLP

Com a conexão estabelecida, os dois DTEs podem, então trocar pacotes de dados através de uma linha de comunicação bidirecional.

Quando um dos DTEs deseja terminar o diálogo, ele envia um pacote *CLEAR REQUEST* ao seu par que vai responder com um pacote *CLEAR CONFIRMATION* em reconhecimento.

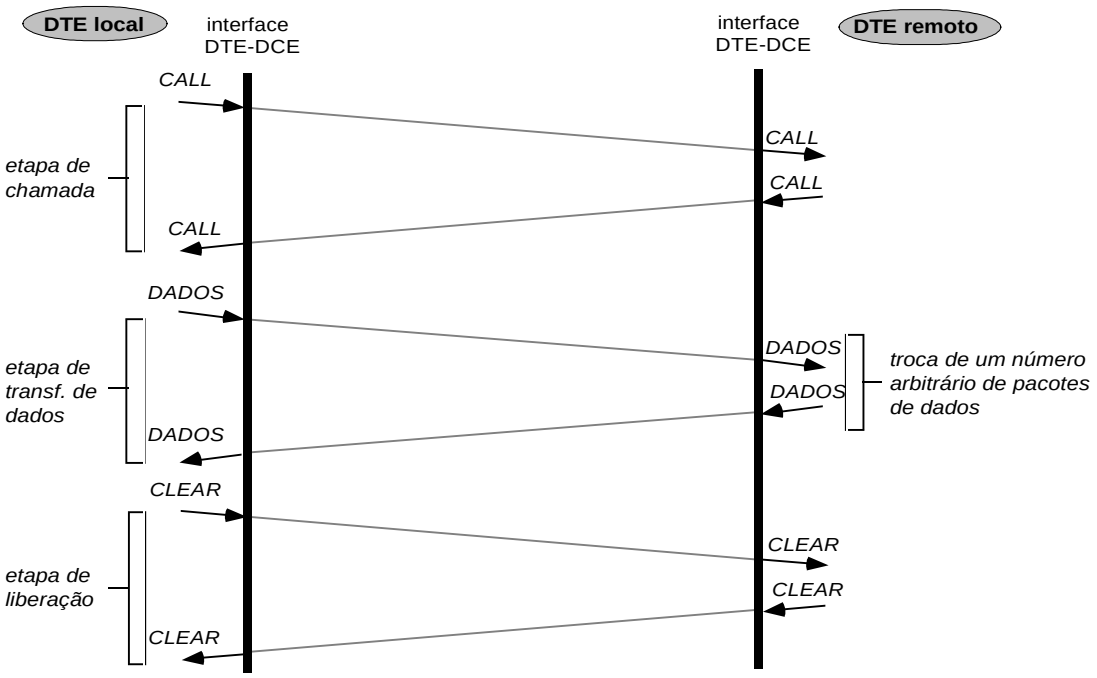


Figura 2.3.30 - Comunicação entre dois DTEs via X.25.

A figura 2.3.31 apresenta o formato de um pacote *CALL REQUEST*. Este pacote, assim como os demais do protocolo X.25, inicia por um cabeçalho de 3 bytes. Os campos *GRUPO* e *CANAL* compõem um número de circuito virtual a ser utilizado no diálogo, este número codificado em 12 bits, o que significa que pode-se utilizar até 4096 circuitos virtuais distintos, sendo que o número 0 é reservado pelo protocolo. O campo *TIPO* permite identificar o tipo do pacote enviado, o bit *CM* estando a 1 nos pacotes de controle e a 0 nos pacotes de dados. O cabeçalho é, de fato, comum a todos os pacotes do X.25. Já os campos a seguir são específicos ao pacote *CALL REQUEST*. Os dois campos seguintes, *TAM. END. INIC.* e *TAM. END. RESP.* representam os tamanhos dos endereços do iniciador e do respondedor, respectivamente. Eles são codificados na forma de dígitos decimais, em 4 bits por campo.

O sistema de endereçamento do X.25 é similar ao do sistema telefônico, onde cada usuário é identificado por um número decimal formado pelo código do país, um código de rede e um endereço interior da rede correspondente, este endereço podendo conter até 14 dígitos decimais. O campo *TAM. FACILIDADES*, permite indicar o tamanho do campo seguinte, *FACILIDADES*, que por sua vez permite ao usuário requisitar serviços especiais (*facilidades*) necessários à comunicação. Um exemplo destas «facilidades» pode ser a chamada a cobrar (PCV). Finalmente, o campo *DADOS USUÁRIO* permite ao DTE enviar até 16 bytes de informação com o pacote *CALL REQUEST*.

O pacote de dados do X.25 é mostrado à figura 2.3.32. O bit *Q* indica *dado qualificado*. Este bit não tem grande utilidade no protocolo de Rede, mas ele permite aos

protocolos das camadas superiores fazer a distinção entre suas mensagens de controle e suas mensagens de dados. O campo *CM* é sempre 0 para os pacotes de dados.

Os campos *SEQÜÊNCIA* e *SUPERPOS.* são utilizados no controle de fluxo com janela de antecipação. O bit *D* indica o significado do campo *SUPERPOS.*: se *D = 0*, o reconhecimento vai indicar que o DCE local recebeu o pacote; se *D = 1*, o reconhecimento vai indicar que o DTE remoto recebeu o pacote. O campo *M* (*More*) indica a pertinência de um pacote a um grupo de pacotes considerado.

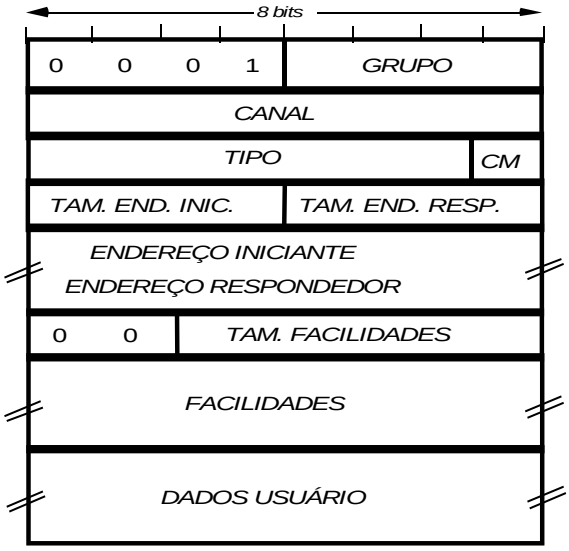


Figura 2.3.31 - Pacote *CALL REQUEST* do X.25.

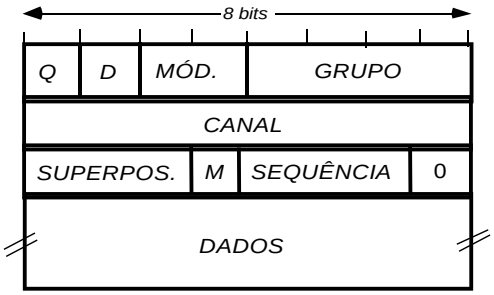


Figura 2.3.32 - Pacote de dados do X.25.

Os demais pacotes de controle do X.25 são apresentados na figura 2.3.33, alguns deles contendo apenas o cabeçalho e outros contendo informações adicionais (até 2 bytes). No caso de um pacote *CLEAR REQUEST*, por exemplo, o quarto byte indica a causa da liberação da conexão.

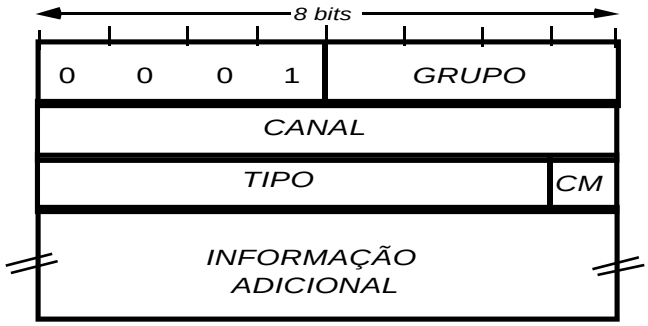


Figura 2.3.33 - Pacotes de controle do X.25.

b) O Protocolo IP (Internet Protocol)

O protocolo IP foi projetado para redes de comutação de pacotes. Assume-se que a rede é composta de várias sub-redes de tipos diferentes, interligadas por "routers" (roteadores). Cada estação na rede é denominada um "host" (hospedeiro).

O IP é um protocolo sem conexão e sem reconhecimento (opera, portanto, com datagrama não confiável). Ele só faz controle de erros do cabeçalho das mensagens, para garantir a correção do endereço de destino, vital para a função de roteamento. O IP também não faz controle de fluxo. Como o protocolo IP é usualmente utilizado em conjunto com um protocolo de transporte chamado TCP, estas funções ficam a cargo deste outro protocolo, como veremos na seção seguinte.

O IP usa endereçamento hierárquico e suporta roteamento dinâmico distribuído nos routers. Um router pode também descartar pacotes se não houver espaço em buffer para armazena-los.

O esquema de endereçamento adotado no protocolo IP é o seguinte: cada estação na rede possui um endereço único, composto de 32 bits (4 bytes), subdivididos em 3 classes mais utilizadas:

- classe A: byte 1 => msb a zero, demais 7 bits identificam sub-rede (de 1 a 126); 24 bits restantes usados para definir o endereço local do host (até 16 milhões de hosts por sub-rede)
- classe B: 2 bytes para sub-rede e 2 bytes para host (até 65.536 hosts por sub-rede)
- classe C: 3 bytes para sub-rede e 1 byte para host (até 254 hosts, valores 0 e 255 reservados)

O esquema completo de endereçamento é apresentado na figura 2.3.34.

Usa-se uma *máscara de endereçamento* para definir a partir de que byte começa o endereço do host, separando-o do endereço da subrede. Por exemplo, a máscara 255.255.255.0 indica que os 3 primeiros bytes fazem parte do endereço da subrede, enquanto o quarto byte contém o endereço do host.

Desta forma, uma estação de rede teria um endereço IP do tipo: 150.162.14.1 (host *atlas* no LCMI-UFSC) ou 150.162.14.4 (host *polaris*, também no LCMI). Observe que as duas estações do exemplo acima estão situadas na mesma subrede. Os bytes nos exemplos acima são separados por pontos para facilitar a conversão do formato string para endereço IP.

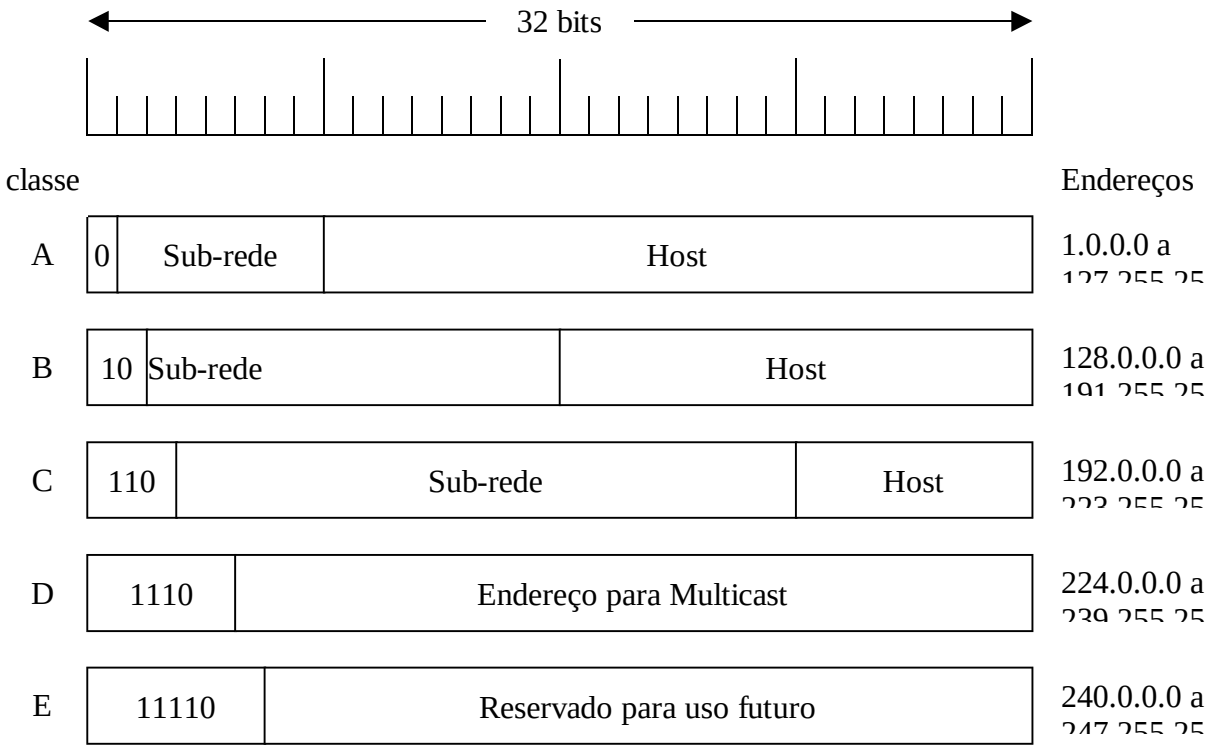


Figura 2.3.34 – Endereçamento IP

2.3.5. A CAMADA DE TRANSPORTE

A função da camada de Transporte é permitir a transferência de informações do sistema emissor ao sistema receptor de forma confiável e econômica, independentemente da natureza da informação ou das redes suportando a comunicação.

A importância dos serviços fornecidos por esta camada está no fato que muitas aplicações existentes podem funcionar simplesmente com a existência de um serviço confiável de transporte de informação, o que quer dizer que os serviços que poderiam ser fornecidos pelas camadas superiores são dispensáveis. Um exemplo disto é a interconexão de estações no sistema UNIX.

Nesta parte do documento serão apresentadas as principais definições relacionadas aos serviços e protocolos de transporte, particularmente do ponto de vista do modelo OSI.

2.3.5.1. O serviço oferecido à camada de Sessão

Segundo o modelo OSI, os usuários da camada de Transporte são as entidades de Sessão, às quais deve ser oferecido o serviço confiável de transporte dos bits de informação fim-a-fim, este serviço sendo fornecido através de uma entidade de software ou de hardware denominada *entidade de transporte*.

De maneira similar à camada de Rede, a de Transporte pode fornecer duas classes de serviço, ou seja, *sem conexão* e *orientados à conexão*.

Os serviços de Transporte orientados à conexão são caracterizados pelas três etapas já descritas para outros níveis do modelo OSI, isto é, *estabelecimento de conexão, transferência de dados* e *liberação da conexão*.

Estes serviços são bastante similares aos serviços oferecidos pela camada de Rede, o que poderia colocar em dúvida a necessidade desta camada. No entanto, a sua existência se justifica pela necessidade de serviços de supervisão da camada de Rede do ponto de vista das entidades efetivamente envolvidas na comunicação. Trata-se aqui de uma supervisão fim-a-fim, uma vez que, até o nível de Rede, as comunicações se fazem ponto-a-ponto.

Outra contribuição importante da camada de Transporte é que ela permite a utilização de primitivas de serviço padrão pelas diversas aplicações construídas sobre a rede efetuando um perfeito «isolamento» em relação às camadas superiores e tornando transparentes as possíveis alterações tecnológicas que poderiam ocorrer nos níveis inferiores. Por esta razão, constuma-se fazer uma distinção entre os níveis de 1 a 4 e os de 5 a 7. Os primeiros quatro níveis seriam mais orientados ao transporte efetivo das informações e os três níveis superiores, mais orientados às aplicações que serão construídas sobre a rede.

Poderíamos sintetizar o serviço fornecido pela camada de Transporte como de supervisor da *qualidade de serviço* oferecido pela camada de Rede. Isto significa que, se a camada de Rede é confiável, a camada de Transporte não terá muito a fazer.

Por outro lado, se o serviço de Rede é deficiente, a camada de Transporte assume a função de suprir as diferenças entre a qualidade de serviço que a camada de Sessão necessita e aquilo que a camada de Rede pode oferecer.

Para isto, o conceito de *qualidade de serviço* (*QOS, Quality Of Service*) é um aspecto importante na concepção da camada de Transporte, baseado sobre um certo conjunto de parâmetros, entre os quais:

- o tempo de estabelecimento de uma conexão;
- a probabilidade de falha de um estabelecimento;
- a taxa de débito da conexão;
- o tempo de trânsito;
- a taxa de erro residual;
- a probabilidade de incidente de transferência;
- o tempo de desconexão;

- a prioridade, etc...

No momento do pedido de um estabelecimento de conexão, o usuário iniciante encaminha estes parâmetros nas primitivas de serviço. Se a camada de Transporte julga certos parâmetros longe da realidade, ela pode sinalizar isto ao usuário iniciante, sem mesmo ter tentado estabelecer a conexão, através de uma mensagem de erro que vai, também, indicar a natureza do erro sinalizado. Outra possibilidade é a camada de Transporte julgar que um certo valor para um parâmetro seja impossível de oferecer, mas que um valor não muito longe daquele poderia ser oferecido. Neste caso, ela pode modificar os valores dos parâmetros enquadrados e encaminhar o pedido de conexão à máquina remota.

Ainda, se a máquina distante verifica que ela não pode oferecer determinados valores especificados nos parâmetros do pedido, ela pode modificar aqueles parâmetros. Se ela verificar que não pode atender determinados parâmetros nos valores mínimos permitidos, a conexão será rejeitada.

2.3.5.2. As primitivas de serviço de Transporte

As primitivas de serviço de Transporte do modelo OSI são apresentadas na tabela a seguir, existindo para os serviços orientados à conexão e sem conexão.

SERVIÇO ORIENTADO Á CONEXÃO
T_CONNECT.request (called, calling,exp_data,qos,user_data) *T_CONNECT.indication (called, calling, exp_data,qos,user_data)* *T_CONNECT.response (qos, responder, exp_data, user_data)* *T_CONNECT.confirm (qos, responder, exp_data, user_data)*
T_DISCONNECT.request (user_data) *T_DISCONNECT.indication (reason, user_data)*
T_DATA.request (user_data) *T_DATA.indication (user_data)* *T_EXPEDITED_DATA.request (user_data)* *T_EXPEDITED_DATA.indication (user_data)*
SERVIÇO SEM CONEXÃO
T_UNITDATA.request (called, calling,qos,user_data) *T_UNITDATA.indication (called, calling,qos,user_data)*

Como podemos ver, as primitivas de serviço de Transporte se assemelham bastante àquelas do serviço de Rede. Existe, porém, uma diferença fundamental entre os dois níveis. No caso do nível Rede, considera-se que o serviço oferecido corresponde ao funcionamento real do sistema, representando inclusive suas falhas. Desta forma, podem ocorrer perdas de pacotes ou a emissão de comandos *N_RESET* de sua própria iniciativa.

Já no caso do nível Transporte, os comandos *N_RESET* não são propagados aos usuários (as camadas superiores), uma vez que o objetivo principal desta camada é o tratamento de todos os problemas de comunicação evitando que os usuários do serviço tomem conhecimento das más condições de funcionamento da rede de comunicação (se tal é o caso!).

Uma outra diferença entre os dois serviços são os usuários destes. No caso do serviço de Rede, os usuários são as entidades de Transporte, normalmente elementos associados ao sistema operacional considerado ou a uma placa específica instalada nos sistemas.

Já os usuários do serviço de Transporte podem ser programas escritos pelos programadores de aplicações, uma vez que, como já foi dito, muitas aplicações podem comunicar-se diretamente através do uso direto das primitivas de serviço de Transporte (não fazendo uso dos serviços de Sessão, de Apresentação ou de Aplicação).

As possíveis relações entre as primitivas de serviço da camada de Transporte são apresentadas na figura 2.3.5.1. As duas extremidades das ilustrações caracterizam os usuários do serviço de Transporte, o fornecedor sendo representado pelo espaço separando os dois usuários.

Em 2.3.5.1(a) é ilustrada um estabelecimento normal de conexão, onde o usuário da esquerda envia uma primitiva *T_CONNECT.request* à camada de transporte. O usuário da direita vai receber então uma primitiva *T_CONNECT.indication*, cujos parâmetros vão conduzir os valores, particularmente, da qualidade de serviço a serem negociados. Este aceita o estabelecimento da conexão, retornando à camada de Transporte uma primitiva *T_CONNECT.response*, que será refletida, via serviço de Transporte, no usuário da esquerda na forma de uma primitiva *T_CONNECT.confirm*.

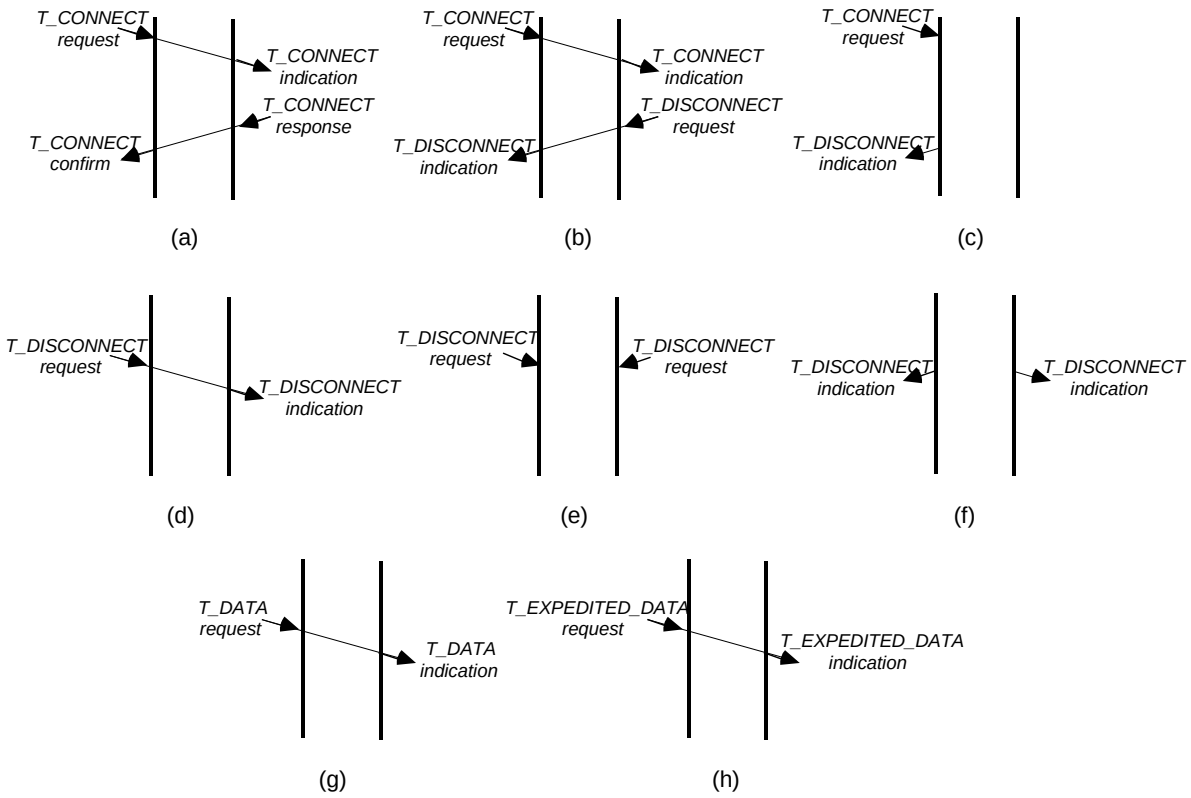


Figura 2.3.5.1 - As diversas interações entre as primitivas de serviço de Transporte.

A figura 2.3.5.1(b) apresenta uma tentativa de estabelecimento de conexão, rejeitada, porém, pelo usuário da direita, que utiliza as primitivas de serviço *T_DISCONNECT.request* e *T_DISCONNECT.indication* para sinalizar a rejeição.

O caso é similar na figura 2.3.5.1(c), porém, desta vez, é o próprio fornecedor do serviço de Transporte quem rejeita a conexão, enviando uma primitiva *T_DISCONNECT.indication* ao usuário da esquerda.

No que diz respeito à liberação da conexão, pode-se considerar três diferentes maneiras: a primeira, ilustrada em 2.3.5.1(d), considera o caso em que a liberação é iniciada por um dos usuários (no caso o da esquerda). A conexão é liberada no momento em que o usuário da direita recebe uma primitiva *T_DISCONNECT.indication*, reflexo da primitiva *T_DISCONNECT.request* emitida pelo usuário que iniciou a liberação; em 2.3.5.1(e) é considerado o caso em que os dois usuários iniciam, simultaneamente, a liberação da conexão, neste caso, a conexão é liberada sem a emissão de outras primitivas de serviço; finalmente, em 2.3.5.1(f), a liberação é iniciada pelo próprio serviço de Transporte, que vai emitir primitivas *T_DISCONNECT.indication* aos dois usuários conectados.

Já as figuras 2.3.5.1(g) e 2.3.5.1(h), representam, respectivamente, a etapa de transferência de dados, para dados normais e expressos.

2.3.5.3. Os protocolos de Transporte

Como definido pelo modelo OSI, o serviço oferecido por uma camada *N* é o resultado da implementação de um protocolo *N* regendo a comunicação entre duas entidades *N*. Em princípio, os protocolos de Transporte apresentam preocupações similares à dos protocolos de Enlace, ou seja, o controle de erros, o controle de fluxo, o sequenciamento dos dados.

No entanto, algumas diferenças podem ser levantadas:

- no caso da camada de Enlace, dois IMPs comunicam-se via um canal de comunicação; a nível de Transporte, a comunicação é feita via uma subrede;
- no que diz respeito ao endereçamento, a nível de Enlace, um IMP não necessita referir-se explicitamente com qual IMP ele quer dialogar, uma vez que dois IMPs são conectados por uma via única (exceto no caso de redes de difusão); já, no nível Transporte, o endereço do destinatário deve ser explicitamente definido;
- um outro problema é a questão do controle de fluxo; ao contrário do nível de Enlace, onde cada conexão pode alocar um determinado número de buffers para o armazenamento dos quadros, no nível Transporte isto fica mais difícil dado o número de conexões que pode estar sendo gerenciado num dado instante.

Um aspecto a ser levado em conta no momento da concepção da camada de Transporte é o serviço oferecido pela camada de Rede, que pode ser classificado da seguinte maneira:

- serviços do tipo A, que caracteriza os serviços "perfeitos", onde a fração de pacotes perdidos, duplicados ou corrompidos é desprezível; é o tipo de serviço dificilmente encontrado em redes públicas, mas algumas redes locais são bastante próximas deste tipo de serviço;
- serviços do tipo B, onde a perda de pacotes também é rara, mas a comunicação é freqüentemente interrompida (serviço N_RESET) devido a problemas de congestão, de hardware ou software;
- serviços do tipo C, não confiáveis, sendo caracterizados normalmente pelas redes a longa distância do tipo datagrama.

Para cada tipo de serviço, de A a C, o nível do serviço de Transporte caminha do mais simples ao mais complexo, ou seja, quanto pior o serviço oferecido pela camada de Rede, melhor (e mais complexo) deverá ser o serviço oferecido pela camada de Transporte.

Para permitir a definição de vários níveis de oferecimento de serviço, o modelo OSI, a nível da camada de Transporte, define cinco classes de serviço:

- *classe 0*, que são os serviços mais simples, capazes de estabelecer uma conexão, mas baseados na hipótese que o serviço de Rede não gera erros de transmissão; não existe tratamento de erros, controle de fluxo nem sequenciamento;
- *classe 1*, corresponde à classe 0, mas permite implementar a retomada de diálogo, considerando que pode ocorrer quebra do diálogo a nível de Rede (serviço N_RESET); ele permite a retomada da comunicação entre duas entidades de Transporte após a interrupção de uma conexão de Rede;
- *classe 2*, que torna mais sofisticada a classe 0 pela introdução da possibilidade de manutenção de diversas conexões de Transporte sobre uma única conexão de Rede (multiplexação);
- *classe 3*, que agrupa os mecanismos de retomada de diálogo da classe 1 com os de multiplexação da classe 2;
- *classe 4*, definida para operar sobre os serviços de Rede do tipo C cuja falta de confiabilidade é conhecida, devendo tratar então os erros, perdas, duplicações, retomada de diálogo e todos os possíveis problemas não resolvidos pela camada de Rede.

No momento do estabelecimento de uma conexão, são as entidades pares que devem tomar a decisão sobre qual classe de serviço adotar, uma classe sendo proposta pelo iniciante e negociada para permitir o estabelecimento da conexão.

As funções do protocolo de Transporte vão depender do ambiente no qual ele vai operar assim como da natureza dos serviços que devem ser supridos. Um mínimo de funções deve, todavia, ser oferecido por um protocolo de Transporte, entre as quais:

- o estabelecimento de conexão;
- transferência de TPDUs;
- segmentação de mensagens, etc...

O termo *TPDU* indicado acima serve para definir a unidade de dados trocada entre duas unidades de Transporte que é a *unidade de dados de protocolo de transporte* (em inglês, *Transport Protocol Data Unit*). Por outro lado, a unidade de dados emitida por um usuário do serviço Transporte será referenciado como sendo uma *mensagem*.

Uma mensagem a ser transmitida pode ter um tamanho qualquer, a *segmentação das mensagens* devendo ser assumida por esta camada. Isto significa que, se uma unidade de dados apresenta um tamanho superior ao de um pacote, ela deve ser segmentada em tantos pacotes quantos sejam necessários para efetuar a sua completa emissão.

Ainda, se uma entidade de Transporte gerencia várias conexões num dado instante, ela deve numerar as conexões e inserir o número correspondente nas unidades de dados, a fim de

permitir à entidade receptora de identificar corretamente a conexão à qual esta unidade de dados deve ser associada. Independente da classe de protocolo considerada, deve sempre existir um mecanismo de liberação de conexão, esta liberação sendo, porém, diferente dependendo da classe de protocolo considerada.

2.3.5.4. O gerenciamento de conexões

O gerenciamento de conexões é uma das importantes funções assumidas pela camada de Transporte. Como já foi dito, embora esta função possa parecer similar àquela suprida pela camada de Enlace, o gerenciamento das conexões de Transporte é bem mais complexo do que das conexões de Enlace.

Gerenciar uma conexão a nível de enlace ou rede é relativamente fácil, pois envolve sempre estações adjacentes. A situação se complica a nível de transporte, pois a conexão envolve estações finais (origem e destino), com várias estações intermediárias não consideradas como parte da conexão. Discutiremos a seguir vários aspectos ligados ao gerenciamento de conexões de transporte.

a) O Endereçamento

Um endereço a nível de transporte (TSAP, *Ponto de Acesso ao Serviço de Transporte o*u *Transport Service Access Point*),) é análogo ao NSAP de rede, mas o NSAP endereça nós de rede, enquanto o TSAP endereça um processo de aplicação dentro do nó.

Cada Processo de aplicação deve se associar a um TSAP para enviar dados. A forma de associação entre um processo e um TSAP não é definida pelo modelo OSI e dependente do sistema operacional de cada estação (ex.: no UNIX, via sockets).

A comunicação entre processos aplicativos requer a indicação tanto do NSAP quanto do TSAP. Quando é usado um esquema de endereçamento hierárquico, o NSAP é usualmente um campo dentro do TSAP.

Se endereçamento não é hierárquico ou o NSAP não é um campo de TSAP, é necessário mapear TSAPs em NSAPs por meio de um Servidor de Nomes (“Name Server”) ou pedindo por difusão que a máquina com um dado TSAP se identifique, i.é, envie seu NSAP como resposta (ex.: protocolos ARP e RARP).

Quando um usuário da camada de Transporte quer dialogar com um outro usuário, ele deve primeiramente, identificar, de alguma forma, o usuário (ou processo de aplicação) que vai estar envolvido no diálogo. Isto é feito normalmente pela definição de um *TSAP*. A relação entre os TSAPs, NSAPs e uma conexão de Transporte é apresentada na figura 2.3.5.2 a seguir, onde o processo de conexão pode ser implementado da seguinte forma:

- um processo servidor localizado na máquina B se conecta ao TSAP 122 e espera a recepção de uma indicação de conexão;

- um processo situado em A requisita uma conexão ao servidor em B, enviando uma primitiva de serviço *T_CONNECT.request* especificando a fonte, o TSAP 6 e o endereço do processo destinatário, o TSAP 122.

- a entidade de Transporte A seleciona um NSAP disponível na máquina e estabelece uma conexão de Rede, através da qual ela vai poder dialogar com a entidade de Transporte em B;

- a entidade de Transporte em B gera então uma primitiva de serviço *T_CONNECT.indication*; se o servidor em B está pronto a aceitar a conexão, esta será, então, estabelecida.

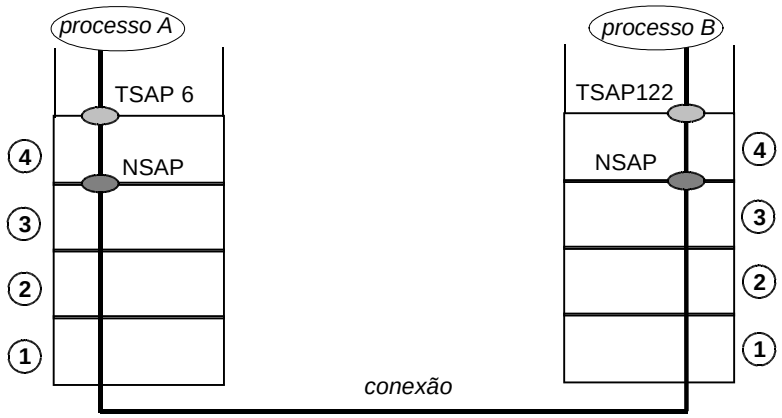


Figura 2.3.5.2 - Relação entre TSAP, NSAP e conexão de Transporte.

O problema do esquema acima é o seguinte: como o processo cliente na estação A sabe o TSAP do processo servidor na estação B (no caso, 122) ? Uma solução possível seria a seguinte: processos servidores muito usados recebem um endereço TSAP fixo e conhecido por todos os demais processos no sistema. Esta solução, no entanto, é pouco flexível e não resolve o problema mais geral de 2 processos de aplicação quaisquer querendo se comunicar entre si.

As formas mais usuais de realizar isto na prática são as seguintes:

- Uso de um ***Servidor de Processos*** (Process Server ou logger): cada nó tem um logger, cujo TSAP é conhecido, e todo pedido de serviço é encaminhado a ele. O Logger cria uma instância de um processo servidor cujo serviço foi requisitado, dá a ele um TSAP e estabelece conexão entre este e o processo cliente que requisitou o serviço.

- Uso de um ***Servidor de Nomes*** (Name Server): todos os processos servidores são cadastrados no Name Server, que é contatado pelos clientes e devolve TSAP do servidor para o serviço requisitado (como um serviço de informação da telefônica). O TSAP do Name Server é fixo.

b) O estabelecimento de uma conexão

Este é um processo relativamente complexo, particularmente se o serviço de Rede fornecido é relativamente pobre (tipo C, por exemplo) e não se resume no envio de uma primitiva de pedido de conexão (*Connection Request — CR*) e na recepção de uma confirmação (*Connection Confirm — CC*).

O problema está simplesmente associado ao fato que, num serviço de rede pobre, pode ocorrer perda ou duplicação de pacotes. Numa rede onde ocorrem problemas de congestionamento os reconhecimentos podem não chegar nunca em tempo hábil, provocando a retransmissão dos pacotes duas ou mais vezes. Ainda, se o serviço de rede é do tipo «datagrama», os pacotes vão seguir caminhos diferentes, alguns deles ficando presos em caminhos bloqueados e reaparecendo um período de tempo após.

Este aspecto pode ser relativamente prejudicial numa aplicação distribuída, uma vez que um comando pode, sem o conhecimento do usuário, ser efetuado diversas vezes (se este comando for o depósito na conta de alguém, o usuário que requisitou o depósito pode ter muito a perder na operação, particularmente se o proprietário da conta para onde o depósito foi requisitado não for muito honesto). Um cuidado especial deverá, então, ser tomado para eliminar o efeito da duplicação dos pacotes a nível do serviço de Transporte.

Uma forma de fazê-lo é através da definição de um parâmetro (número) de referência a ser incluído em cada unidade de dados, inclusive naquelas conduzindo o pedido de conexão. A cada liberação de conexão, uma tabela contendo as referências das conexões obsoletas será atualizada. Desta forma, a cada vez que um pedido de conexão for recebido, a entidade vai verificar, através do parâmetro de referência associado, se o pedido corresponde ou não a uma conexão já liberada. Em caso positivo, o pedido de conexão será ignorado. O problema com esta técnica é o fato que, caso uma estação entre em pane, os valores constantes na tabela de conexões podem ser perdidos e, assim, na retomada de execução, a entidade de transporte vai perder a informação sobre que conexões teriam sido efetuadas ou não.

Outra forma de fazê-lo é através da eliminação dos pacotes duplicados após um certo período de tempo. Isto pode ser implementado de diferentes maneiras:

- através da limitação do tamanho da subrede, diminuindo o risco de que pacotes fiquem circulando na rede e reduzindo, conseqüentemente, os riscos de congestionamento;

- a associação de um contador de «saltos» (um salto sendo a passagem por um nó intermediário) a cada pacote, de tal modo que, quando o contador ultrapassa um determinado valor, o pacote é destruído;
- a associação de uma data (de envio) a cada pacote de modo que o pacote será destruído no momento em que a sua «idade» tiver ultrapassado um certo valor.

Esta terceira solução apresenta uma dificuldade relativamente grande que é a manutenção da sincronização dos relógios ao longo das diferentes estações da rede.

Uma das soluções mais freqüentemente adotadas para a solução deste problema é o chamado “three-way handshake”, cuja operação é a seguinte: duas TPDUs com numeração idêntica não podem estar pendentes (esperando confirmação) ao mesmo tempo. A numeração das TPDUs não pode se repetir dentro de um intervalo de tempo T. A operação se dá em 3 passos (three-way):

- nó A envia pedido de conexão com número de seqüência inicial x
- nó B envia confirmação do pedido contendo valor de x e seu próprio número de seqüência inicial y
- nó A envia primeira TPDU contendo confirmação do valor y

Nenhuma nova TPDU com número de seqüência x é aceita antes de um intervalo de tempo T. A figura 2.3.5.3 ilustra a operação do protocolo em três situações distintas.

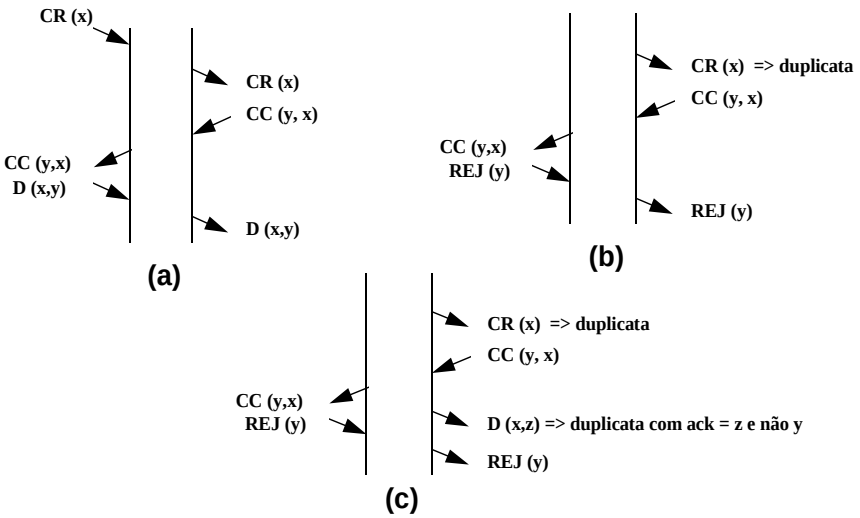


Figura 2.3.5.3 - Three-Way hadshake: (a) operação normal; (b) CR duplicado; (c) CR e D duplicados

c) A liberação da conexão

A liberação de uma conexão, embora seja mais simples do que o estabelecimento, exige certos cuidados a fim de evitar a ocorrência de certas situações indesejáveis. A figura

2.3.5.1 apresentava, em (d), (e) e (f), três situações típicas da liberação de uma conexão, a primeira situação revelando-se a mais clássica e mais freqüente.

Estas três situações são caracterizadas pela interrupção brutal da conexão, podendo inclusive ocorrer perda de dados que teriam sido transmitidos. Isto é ilustrado na figura 2.3.5.4, no primeiro caso descrito.

O protocolo de liberação de conexão deve, então, ser melhor elaborado a fim de evitar este tipo de problema. Uma solução seria a definição de um diálogo que permitisse assegurar que ambos os usuários terminaram efetivamente a transmissão de dados. Isto, no entanto, pode ser ineficiente, pois não é evidente definir como este diálogo poderia ser conduzido, particularmente se o transporte utiliza um serviço de Rede não confiável.

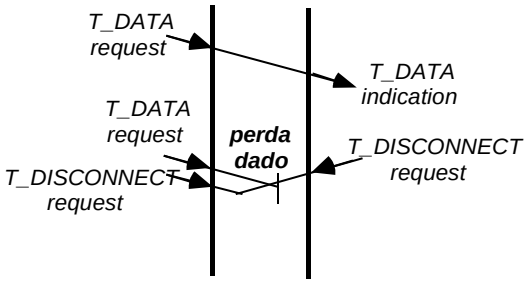


Figura 2.3.5.4 - Perda de dados no caso de uma liberação de conexão.

A liberação pode ser mais eficiente se um procedimento em três etapas é definido (three-way handshake), como mostrado na figura 2.3.5.5. Em 2.3.5.5(a), é mostrado o caso normal, onde o usuário que quer liberar a conexão envia um *DR* (*Disconnect Request*)para indicar a sua intenção. Após o envio, este dispara um temporizador (simbolizado na figura por *+ t*). O usuário receptor responde com um *DC* (*Disconnect Confirm*), disparando também um temporizador. Quando o *DC* chega ao usuário iniciador, este emite finalmente um *ACK* e se desconecta. Na seqüência, o usuário que recebe o *ACK* vai também se desconectar.

Já em 2.3.5.5(b), é mostrado como o protocolo vai se comportar em caso da perda do *ACK*. Neste caso, o temporizador disparado pelo usuário da direita vai resolver o problema, efetuando a desconexão na sua expiração.

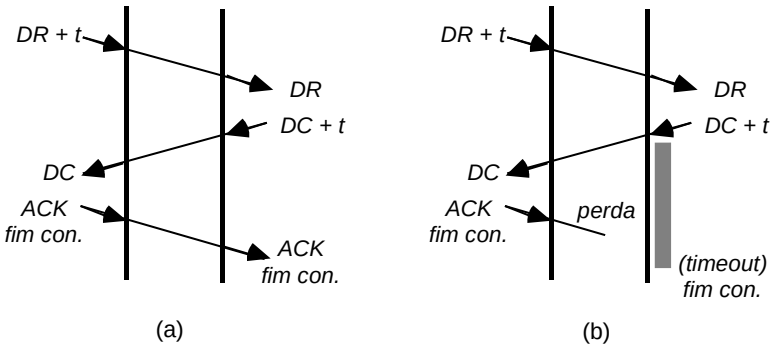


Figura 2.3.5.5 - Liberação de conexão em três etapas: (a) caso normal; (b) caso de perda do reconhecimento.

d) O controle de fluxo e o armazenamento

Um problema que deve ser tratado durante a manutenção de uma conexão ativa é aquele ligado ao controle de fluxo. Este problema é, de um lado, bastante similar àquele tratado pela camada de Enlace.

Por outro lado, a nível de Transporte, o número de conexões existentes é grande, o que significa uma necessidade de armazenamento muito maior que no caso do nível Enlace (que gerencia uma única conexão de cada vez).

Como foi visto na parte destinada à camada de Enlace, certos protocolos, por questões de eficiência, exigiam uma memorização dos quadros nas duas extremidades da conexão. Isto significava, também, a alocação de um número suficiente de buffers nas duas extremidades para assegurar o armazenamento dos quadros.

No caso de Transporte, este tipo de solução torna-se mais difícil a implementar, necessitando de soluções mais adaptadas ao problema. A solução a implementar pode depender fortemente do serviço oferecido pela camada de Rede.

Se o serviço oferecido é do tipo “datagrama” com reconhecimento, a entidade receptora sabe que a emissora é obrigada a armazenar as unidades de dados enviadas que ainda não foram reconhecidas. Assim, ele pode simplesmente não fazer nenhuma alocação de buffer específico para a conexão, todos os buffers disponíveis podendo ser compartilhados por todas as conexões existentes. Desta forma, quando uma TPDU é recebida, se não existe buffer disponível ela é simplesmente rejeitada. Isto não gera nenhum problema, uma vez que a entidade emissora, após um certo tempo, vai retransmitir a TPDU que não foi reconhecida.

Já, no caso de um serviço de rede confiável, o procedimento pode ser diferente. Num primeiro caso, se o emissor sabe que o receptor terá sempre condições de armazenar as TPDUs por ele emitidas, então este não vai necessitar armazená-las (pois sabe que não haverá necessidade de retransmissão). Caso contrário, ele vai ter de armazená-las de todo modo, uma vez que os reconhecimentos da camada de Rede significam apenas que o pacote foi recebido mas não que a TPDU foi aceita.

Se o armazenamento de TPDUs é feito do lado do receptor, o problema a ser resolvido é como definir o tamanho dos buffers de armazenamento. Se as diversas TPDUs são de tamanhos aproximadamente equivalentes, a solução é alocar um determinado número de buffers de igual tamanho, como mostra a figura 2.3.5.6(a), onde cada buffer vai conter uma TPDU. Se os tamanhos da TPDUs variam muito, esta solução pode não ser a melhor, mesmo se o tamanho dos buffers é estabelecido ao tamanho máximo das TPDUs. Neste caso, alocar um buffer para armazenar uma TPDU de tamanho reduzido é desperdiçar memória. Assim,

uma outra solução é a alocação de buffers de tamanho variável, como mostra a figura 2.3.5.6 (b), onde o desperdício é menor, ou, ainda, alocar um único buffer circular, como mostrado em 2.3.5.6(c).

Estas duas últimas soluções são eficientes no caso de conexões com grande tráfego de TPDUs. Caso contrário, podem não ser as melhores.

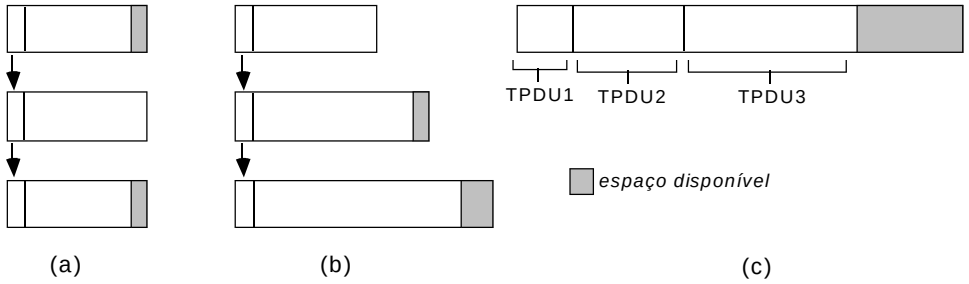


Figura 2.3.5.6 - As diferentes soluções de armazenamento: (a) buffers de tamanhos iguais; (b) buffers de tamanhos variáveis; (c) buffer circular.

O tráfego é importante também na definição da melhor solução no que diz respeito à localização do armazenamento — no emissor ou no receptor. No caso de múltiplas conexões, mas com pouco tráfego em cada uma delas (terminais interativos, por exemplo), é melhor a implementação de um mecanismo de alocação dinâmica de buffers nas duas extremidades. Como o emissor não tem certeza de que o receptor poderá alocar o buffer, ele mantém com ele uma cópia da TPDU até receber o reconhecimento. Por outro lado, no caso de tráfegos relativamente altos, é melhor ter uma alocação estática de buffers associados à conexão na extremidade receptora a fim de permitir a transmissão numa taxa máxima.

Outro ponto importante a ser coberto pela camada de Transporte, é a capacidade de um usuário (emissor) alocar à distância (no receptor) os buffers de maneira dinâmica, os buffers podendo ser alocados para cada conexão ou para um conjunto de conexões existentes entre dois usuários.

e) Multiplexação e Splitting

O mecanismo de multiplexação aplicado sobre uma mesma conexão, um circuito virtual ou uma mesma linha é um aspecto bastante importante no que diz respeito às camadas constituindo a arquitetura de um sistema de comunicação.

Na camada de Transporte, a multiplexação pode ser necessária particularmente em duas situações típicas:

- num primeiro tempo, no caso de uma rede utilizando circuitos virtuais, cada conexão estabelecida vai utilizar-se de recursos presentes nos IMPs durante toda a sua duração; assim, pode ser interessante, para otimizar a utilização dos recursos de rede (e, evidentemente, o custo de uma comunicação!), associar diversas conexões de Transporte a uma mesma conexão de Rede, como mostrado na figura 2.3.5.7(a), onde 4 conexões de Transporte são multiplexadas em uma de Rede — é a *multiplexação para cima*;
- no caso de uma aplicação onde o usuário envia uma quantidade relativamente grande de mensagens, dependendo do protocolo implementado a nível de Rede, uma única conexão pode ser insuficiente para suportar a conexão de Transporte; assim, a camada de Transporte pode decidir efetuar a *multiplexação para baixo*, ou *Splitting*, onde uma conexão de Transporte pode ser mapeada em várias conexões de Rede, como mostrado na figura 2.3.5.7(b).

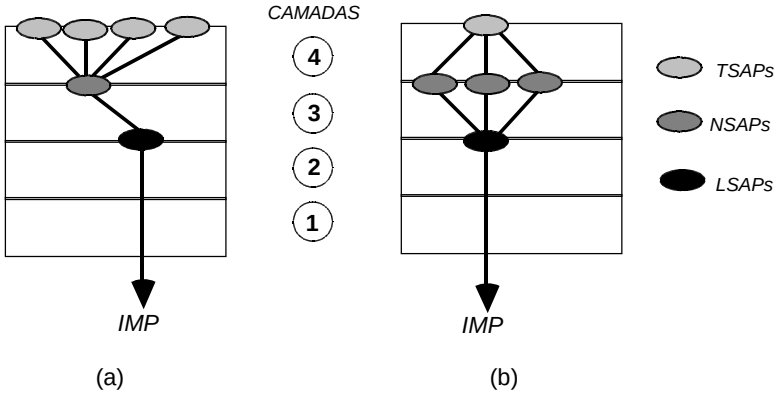


Figura 2.3.5.7 - Multiplexação: (a) para cima; (b) para baixo.

f) A retomada de uma conexão

Uma vez que durante a operação de uma conexão, incidentes podem ocorrer, a camada de Transporte deve tratar também deste problema. No caso da camada de Rede, as primitivas *N_RESET*, foram definidas para o tratamento de incidentes.

As entidades de Transporte, quando da ocorrência de um incidente, devem dialogar para trocar informações sobre quais unidades de dados teriam sido recebidas ou não.

O problema maior ocorre, porém, se a pane ocorre num dos sistemas envolvidos na comunicação. Após a retomada da conexão, todas as informações são reinicializadas de modo que o usuário acaba perdendo a informação de qual era o seu estado antes do incidente.

Uma forma de recuperação do estado é enviar uma unidade de dados a todos os outros usuários a fim de obter as informações sobre todas as conexões existentes. O emissor pode estar em dois estados: unidades à espera de reconhecimento ou não. Assim, o emissor pode

decidir a retransmissão das unidades. Esta solução, porém, não é assim tão simples, uma vez que a recepção do reconhecimento pode não garantir o armazenamento da unidade correspondente se, por exemplo, a pane ocorreu imediatamente após o envio do reconhecimento, quando a unidade recebida não tinha ainda sido armazenada. Neste caso, o emissor poderia retransmitir aquelas que, por ventura, não tivessem sido recebidas no lado do receptor.

Por outro lado, se a unidade é escrita antes do envio do reconhecimento, isto pode causar o problema inverso, o que vai provocar uma duplicação de TPDUs.

2.3.5.5. Exemplos de Protocolo de Transporte

a) Os protocolos OSI

Como já foi dito, o protocolo OSI é organizado em 5 classes, onde cada uma delas permite cobrir uma classe de confiabilidade da camada de Rede. O protocolo de Transporte OSI é baseado na definição de 10 TPDUs, estas compostas de quatro partes:

- um campo de 1 byte, denominado *LI* (*Length Indicator*), indicando o tamanho dos cabeçalhos fixos e variáveis;
- uma parte fixa do cabeçalho, cujo tamanho vai depender da TPDU considerada;
- uma parte variável do cabeçalho, cujo tamanho vai depender dos parâmetros definidos;
- dados do usuário.

A figura 2.3.5.8 apresenta os 10 tipos de TPDUs do modelo OSI. As quatro primeiras TPDUs, *CR*, *CC*, *DR* e *DC* (*CONNECTION REQUEST, CONNECTION CONFIRM, DISCONNECT REQUEST* e *DISCONNECTION CONFIRM*) correspondem aos pacotes *CALL REQUEST*, *CALL ACCEPTED*, *CLEAR REQUEST* e *CLEAR CONFIRM* definidos no protocolo X.25.

Uma entidade de Transporte, quando deseja estabelecer uma conexão, envia uma TPDU *CR* e deverá receber uma *CC*. No momento da liberação da conexão, as TPDUs *DR* e *DC* serão trocadas entre duas entidades.

As TPDUs *DT* (*DATA*) e *ED* (*EXPEDITED DATA*) são utilizadas para transferir, respectivamente, dados normais e dados expressos. Os reconhecimentos relativos a estas duas TPDUs serão, respectivamente, *AK* (*DATA ACKNOWLEDGEMENT*) e *EA* (*EXPEDITED DATA ACKNOWLEDGEMENT*).

Finalmente, as TPDUs *RJ* (*REJECT*) e *ER* (*ERROR*) são utilizadas para o tratamento de erros.

Como se pode ver na figura, todas as TPDUs são compostas do campo *LI*, já descrito. O byte seguinte é utilizado, particularmente no protocolo de classe 4, para implementar um mecanismo de crédito, utilizado para controlar o tráfego de unidades de dados, substituindo o mecanismo com janela de antecipação.

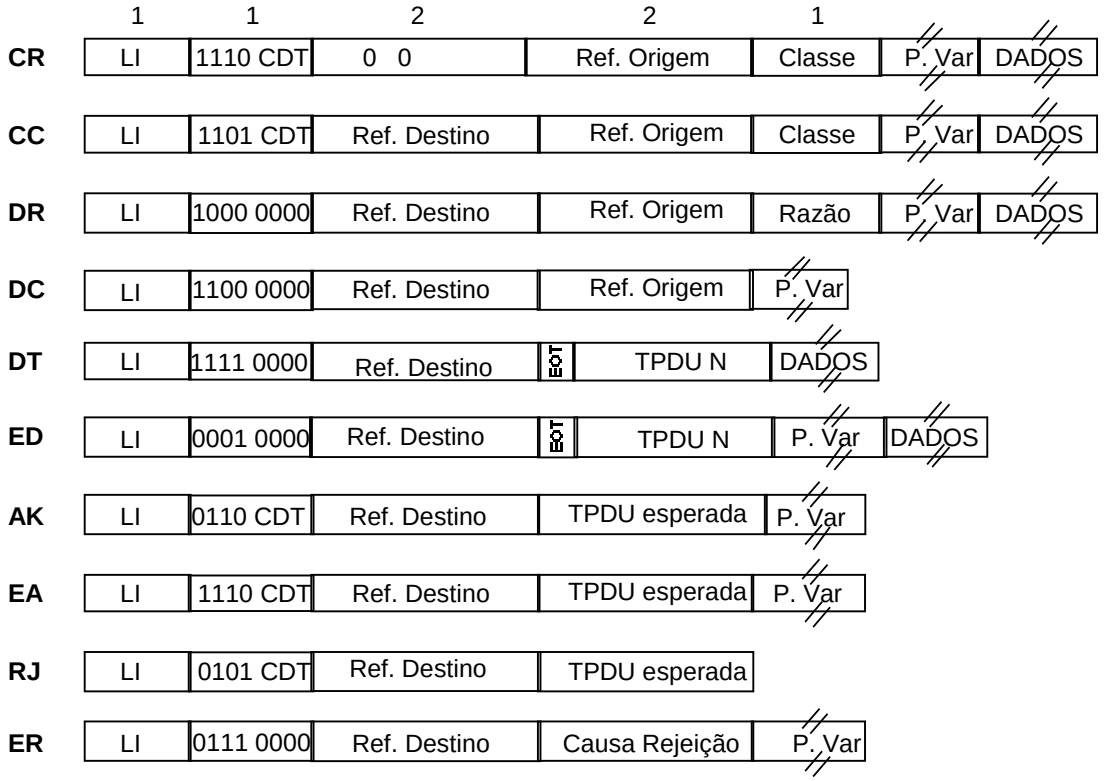


Figura 2.3.5.8 - TPDUs do protocolo de Transporte OSI.

Os campos *Ref. Destino* e *Ref. Origem* permitem identificar as conexões de Transporte. Eles são utilizados nos protocolos de classe 2 a 4, onde pode ocorrer a multiplexação de várias conexões. Quando um pacote é recebido através da camada de Rede, a entidade de Transporte vai analisar estes campos para definir a que conexão ele pertence.

O campo *Classe* é utilizado pelas entidades de Transporte para a negociação da classe do protocolo a ser implementado na conexão.

A parte variável da TPDU vai conter informações, tais como: TSAP devendo ser conectado no usuário local, TSAP devendo ser alocado no usuário remoto, classes de protocolo aceitas em segunda opção, etc.

O campo *DADOS* pode conter até 32 bytes de informações nas classes 1 a 4, não sendo utilizado na classe 0.

No caso das TPDUs *DT* (*DATA*) e *ED* (*EXPEDITED DATA*), o campo *EOT* (*End Of Transmission*) é utilizado para marcar o final de uma transmissão, ele é setado a 1, no caso da TPDU enviada ser a última. Isto é necessário para possibilitar os procedimentos de montagem.

As TPDUs *AK* (*DATA ACKNOWLEDGE*) e *EA* (*EXPEDITED DATA ACKNOWLEDGE*) são utilizadas para reconhecer, respectivamente as TPDUs *DT* e *ED*. O campo *TPDU esperada* funciona de maneira similar ao campo *ack* apresentado na camada de Enlace. A diferença é que ele indica que todas as TPDUs anteriores foram recebidas, exceto aquela referenciada pelo campo.

Finalmente, o campo *Causa Rejeição* presente na TPDU *ER* (*ERROR*) permite indicar a causa do erro sinalizado, como *código de parâmetro inválido, tipo de TPDU inexistente, referência de TPDU inválida*, etc.

No caso de um serviço de Transporte sem conexão, a camada de Transporte OSI, pode se utilizar de um serviço de Rede com ou sem conexão. Como já foi dito na parte relativa à camada de Rede, a definição de um serviço sem conexão que utilize um serviço orientado conexão não é lá muito interessante, mas, às vezes, o único serviço de Rede disponível é orientado à conexão, como por exemplo o X.25.

No caso da utilização de um serviço sem conexão, cada TPDU é associada a um pacote, sem reconhecimento nem garantia de sequenciamento.

O protocolo do serviço de Transporte sem conexão define o formato da TPDU como ilustrado na figura 2.3.5.9, bastante similar à utilizada no caso do serviço orientado à conexão. A parte variável da TPDU conduz os endereços de TSAP (origem e destinatário) e outras informações.

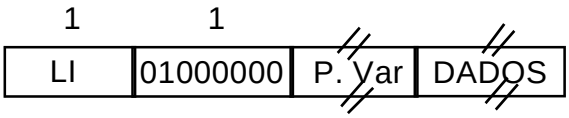


Figura 2.3.5.9 - Formato da TPDU do serviço de Transporte sem conexão.

b) O protocolo TCP

O protocolo *TCP* (*Transmission Control Protocol*) foi definido como resultado dos trabalhos realizados a nível da rede americana *ARPANET*, para poder permitir a inclusão de redes locais cujo serviço de Rede fosse menos confiável do que o da própria ARPANET.

Associado a TCP, foi definido também um novo protocolo de Rede, o *IP* (*Internet Protocol*), sendo que hoje, o par *TCP/IP* é bastante utilizado na interconexão de computadores (padrão UNIX e INTERNET).

Uma entidade de Transporte TCP aceita mensagens de tamanho variável, fragmentando-as em unidade de tamanho inferior ou igual a 64 Kbytes, estas unidades sendo

enviadas como datagramas isolados, a camada de Rede não oferecendo nenhuma garantia de entrega dos datagramas. O TCP deve, então, definir um temporizador e retransmitir as unidades perdidas. O TCP deve ainda tratar o problema de sequenciamento das unidades, uma vez que não é garantida a chegada na ordem correta a nível de Rede.

Cada unidade transmitida via TCP tem um número de seqüência particular (de 32 bits) permitindo tratar o problema de seqüências idênticas. O TCP trata também o problema de unidades atrasadas no estabelecimento de conexão implementando um procedimento a três etapas.

O TCP utiliza um cabeçalho fixo para as unidades de dados, o formato sendo apresentado na figura 2.3.5.10, cujos campos têm as funções seguintes:

- *porta fonte* e *porta destino* realizam o endereçamento em TCP (endereços dos TSAPs);

- *número de seqüência* e *reconhecimento*, efetuam as funções de sequenciamento e reconhecimento, respectivamente; eles ocupam, cada, 32 bits, pois cada byte é numerado no TCP;

- *tamanho cabeçalho* permite indicar o número de palavras de 32 bits que farão parte do cabeçalho; é necessário, pois o campo *opção*, vai ser composto de um número variável de palavras;

- *URG* (*URGent pointer*) bit setado a 1 se o *apontador de prioridades* está ativado; este será explicado mais tarde;

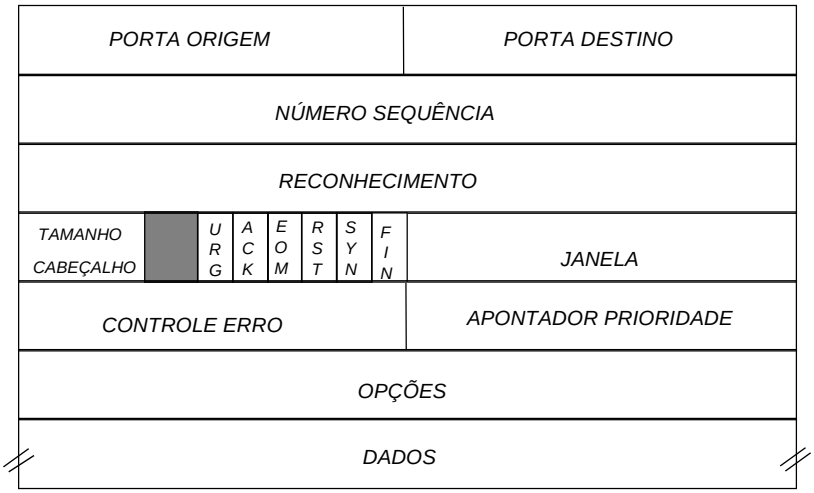


Figura 2.3.5.10 - Formato do cabeçalho da TPDU do TCP.

- *ACK* (*ACKnowledge*) serve para indicar se o campo *reconhecimento* é utilizado (1) ou não (0);
- *EOM* (*End Of Messages*) permite indicar o final de uma transmissão;
- *RST* (*ReSTart*) serve à reinicialização de uma conexão quando a sua operação torna-se comprometida;
- *SYN* é setado a 1 para indicar que a unidade contém um pedido de conexão;
- *FIN* é utilizado para a liberação de conexão; especifica que o emissor não possui mais dados a enviar;
- *janela* é utilizado para implementar o controle de fluxo; ele é composto de 16 bits pelo fato de que o controle é feito a nível de byte e não de TPDU;
- *controle de erro* permite efetuar o controle do erro; este é definido por um algoritmo simples onde os bytes de dados são somados como palavras de 16 bits; este campo apresenta o complemento 1 do resultado da soma;
- *apontador de prioridades* é utilizado para permitir o envio de unidades urgentes, indicando o offset (deslocamento) em relação ao número de seqüência ao qual o dado urgente pode se localizar;
- *opções* é utilizado para definir os parâmetros da comunicação (por exemplo, o tamanho dos buffers);
- *dados* contém efetivamente os dados que serão transmitidos.

No UNIX, os processos usuários do TCP são identificados por uma “porta” (ou “porto”). As portas são locais e podem se repetir em nós diferentes. Os processos são identificados univocamente na rede pelo identificador da sua porta concatenado ao endereço IP do nó, definindo um “socket” (equivalente a um TSAP para o RM-OSI).

Processos servidores muito usados recebem portas fixas, conhecidas dos demais processos (ex.: servidor FTP, Telnet, etc.). O estabelecimento de uma conexão entre processos de aplicação requer um par de sockets, que operam de forma full-duplex. Um mesmo socket pode participar de várias conexões. Uma conexão é definida através dos identificadores dos sockets envolvidos, dos números de seqüência, da largura de janela, etc.

O TCP estabelece e desfaz conexões por meio do protocolo three-way-handshake, já descrito anteriormente.

Os processos de aplicação transmitem dados fazendo chamadas ao TCP e passando como parâmetros os buffers onde os dados se encontram. No caso do UNIX, o TCP foi concebido como módulo do sistema operacional.

A interface com o usuário (ou processos de aplicação) é feita por meio das seguintes chamadas (system calls) ao TCP:

- open (conexão)

- close (conexão)
- send (dados)
- receive (dados)
- status (conexão)

c) Interfaces para Programas de Aplicação - APIs

APIs (Application Program Interfaces) são interfaces para programadores usando diretamente os serviços da camada de transporte. Seu objetivo é facilitar a implementação de programas de aplicação que fazem uso dos serviços de comunicação da rede. APIs são usualmente oferecidas para alguns sistemas operacionais (ex.: UNIX) e algumas linguagens (ex.: C).

Exemplos de APIs bastante difundidas são:

- TLI (Transport Layer Interface): API para linguagem C com TCP em UNIX System V.
- NetBIOS: interface para programação de aplicações distribuídas da IBM. Oferece serviço de transmissão confiável orientado à conexão, serviço de nomes para identificar usuários e serviço de datagrama não confiável (opcional). Existem versões para TCP, SPX (Sequenced Packet Protocol, protocolo de transporte da novell) ou para operar sobre a camada de enlace (LLC classe 1). No último caso, inclui protocolos de rede e transporte.
- Berkeley Sockets: API para linguagem C com TCP em UNIX BSD e compatíveis. Os serviços são acessíveis por meio dos system calls mostrados na tabela a seguir:

System Call	**Função**
socket	cria um socket (TSAP)
bind	associa um nome ASCII a um socket já existente
listen	cria uma fila de espera para requisições de serviço
accept	remove um pedido de conexão de uma fila ou espera por um novo pedido
connect	estabelece uma conexão com outro socket
shutdown	encerra uma conexão entre sockets
send	envia mensagem por uma conexão existente
recv	recebe mensagem por uma conexão existente
select	testa um grupo de sockets para ver se tem mensagens

Para algumas arquiteturas de rede, a subdivisão em camadas termina aqui. Este é o caso da arquitetura TCP/IP (cujo nome deriva do par de protocolos TCP e IP, já vistos). Para a arquitetura TCP/IP, assume-se que os aplicativos operem sobre os serviços de transporte (no caso, os serviços do protocolo TCP ou UDP, que é uma versão sem conexão e não confiável do TCP). Estudaremos alguns destes aplicativos mais a frente. Já no caso do RM-OSI, foram definidas mais três camadas acima da camada de transporte, que estudaremos a seguir.

2.3.6. A CAMADA DE SESSÃO

Como foi mencionado anteriormente, o modelo hierárquico de comunicação proposto no modelo OSI pode fazer distinção em duas classes de camadas: as *camadas inferiores*, concretizadas pelas camadas de 1 a 4, já vistas, cujos serviços são orientados ao transporte de informação própriamente dito, tratando essencialmente de problemas de comunicação como, por exemplo, codificação e transmissão de bits, controle de fluxo e de erros, sequenciamento, roteamento, controle de tráfego e gerenciamento de conexões; as *camadas superiores*, concretizadas pelas camadas de 5 a 7, cujos serviços se orientam mais às aplicações escritas pelos usuários no sentido de facilitar ou simplificar as suas tarefas, fornecendo serviços padronizados os mais diversos.

A camada de Sessão é a primeira camada (no sentido *bottom-up*) enquadrada na segunda classe. Ao contrário de outras camadas já estudadas e outras que serão vistas mais adiante, a camada de Sessão foi introduzida no momento da definição do modelo OSI. As demais camadas presentes no OSI foram, de certo modo, inspiradas de modelos pré-existentes na concepção de redes já existentes na época como, por exemplo, ARPANET.

Esta é, na verdade, uma das camadas mais simples do modelo OSI, oferecendo uma quantidade relativamente limitada, longe dos serviços oferecidos por camadas como a de Transporte, por exemplo.

Nesta parte do documento, veremos então os serviços oferecidos pela Sessão e como estes são implementados através dos protocolos.

2.3.6.1. Serviços oferecidos à camada de apresentação

A principal função desta camada é oferecer aos seus usuários meios para o estabelecimento das conexões, denominadas *sessões*, de modo que estes possam trocar dados. Uma sessão pode ser utilizada para permitir a conexão à distância a um computador, por exemplo, através de um terminal, para uma transferência de arquivo, para o carregamento de programas à distância, etc.

Segundo o modelo OSI, os usuários dos serviços de Sessão são as entidades de Apresentação, a posição desta camada estando ilustrada na figura 2.3.43.

Apesar de que, ao nível de Sessão, são oferecidas primitivas de serviço para a comunicação sem conexão, neste modo, não é possível explorar os serviços orientados aos usuários, disponíveis nesta camada.

No que diz respeito à conexão de Sessão (ou à *sessão*, como definido acima), pode-se estabelecer as diferentes possíveis relações entre uma conexão de Sessão e uma conexão de Transporte, como mostra a figura 2.3.44.

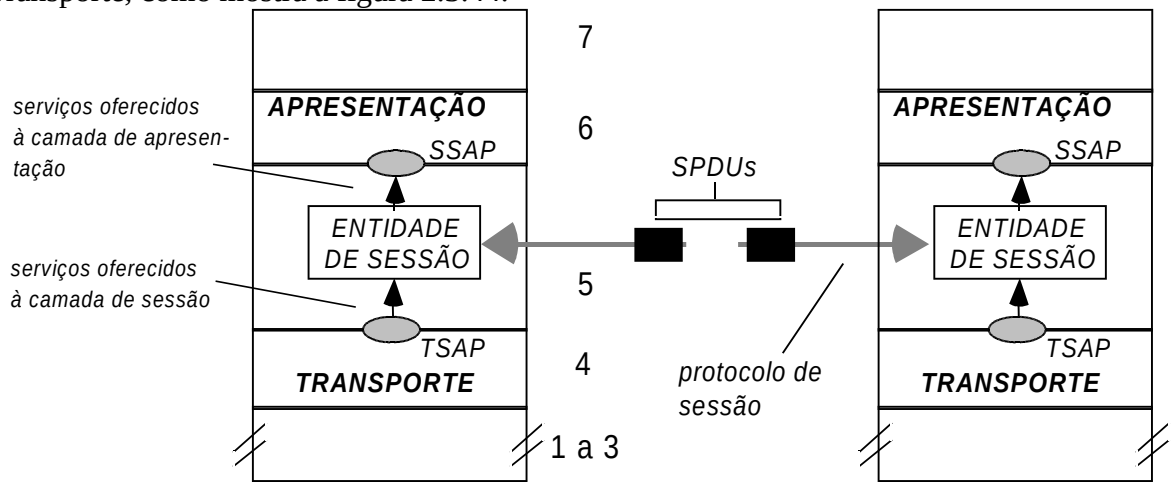


Figura 2.3.43 - Posição da camada de Sessão no modelo OSI.

Em 2.3.44(a), é mostrada uma correspondência de 1 a 1 entre uma conexão de Sessão e uma de Transporte. Por outro lado, como mostrado em 2.3.44(b), pode-se utilizar uma mesma conexão de Transporte para suportar diferentes *sessões* consecutivas. Ainda, pode-se ter o quadro inverso, onde, pela quebra de uma conexão de Transporte, a abertura de uma nova é providenciada para garantir a continuidade de uma mesma *sessão*. Este último cenário, ilustrado em 2.3.44(c), se caracteriza, por exemplo, quando as entidades de Transporte assumem a tarefa de retomada de diálogo após uma pane.

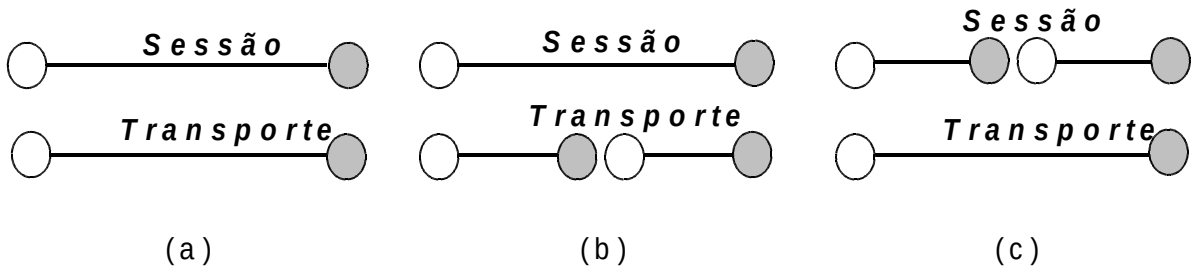


Figura 2.3.44 - Diferentes relações entre conexão de Sessão e de Transporte: (a) correspondência 1 a 1; (b)várias conexões de Transporte para uma única *sessão*; (c) uma conexão de Transporte para várias *sessões* .

2.3.6.2. Sessão x Transporte: similaridades e diferenças

Da mesma forma como foi visto para camadas inferiores, o diálogo via esta camada é caracterizado pelas três etapas a seguir: *estabelecimento de sessão*, *transferência de dados* e *liberação da sessão*.

As primitivas de estabelecimento e liberação de sessão oferecidas à camada de Apresentação são similares àquelas oferecidas à própria camada de Sessão pelo Transporte. Esta similaridade ocorre a tal ponto que, em muitos casos, a recepção de uma primitiva de serviço da camada de Apresentação reflete-se imediatamente no envio de uma primitiva de serviço equivalente da camada de Transporte. Um exemplo disto é a primitiva de estabelecimento de Sessão, *S_CONNECT.request*, que é refletida no envio de uma primitiva *T_CONNECT.request*.

Na fase de estabelecimento de uma conexão, da mesma forma que na camada de Transporte, ocorre a negociação, entre as duas entidades envolvidas (de Apresentação) sobre que parâmetros vão definir a comunicação. Dentre estes parâmetros, encontram-se parâmetros já conhecidos de outros níveis do modelo OSI (por exemplo, a necessidade de transferência de dados expressos) e também parâmetros específicos à camada de Sessão (por exemplo, na abertura de uma sessão para troca de informações via correio eletrônico, um parâmetro pode ser qual usuário vai tomar a iniciativa do diálogo).

Apesar das semelhanças já levantadas, pode-se destacar também algumas diferenças entre uma sessão e uma conexão de Transporte, essas diferenças estando particularmente associadas ao procedimento de liberação das conexões. No caso da camada de Transporte, a primitiva *T_DISCONNECT.request* , na forma adotada no modelo OSI, causa a terminação abrupta da conexão, podendo ocorrer inclusive perda dos dados ainda em trânsito (este problema foi levantado quando da apresentação da camada de Transporte e se aplica, por exemplo, ao protocolo TP4 da ISO). Já, no caso das sessões, a primitiva responsável da liberação é *S_RELEASE.request* que permite terminar, de maneira ordenada a conexão, sem ocorrência de perda dos dados (*liberação negociada*). É possível, no entanto, em caso de necessidade, promover uma liberação abrupta da sessão, isto, graças à utilização da primitiva *S_ABORT.request*. A diferença entre as duas formas de liberação de uma conexão é mostrada na figura 2.3.45. Em 2.3.45(a) é apresentada a liberação abrupta de uma conexão de Transporte; em 2.3.45(b), é apresentada a liberação negociada de uma sessão.

Como se pode ver em 2.3.45(b), ao contrário dos serviços de liberação de conexão apresentados até o momento, o serviço de liberação negociada *S_RELEASE* é um serviço confirmado, caracterizado pelas primitivas *request*, *indication*, *response* e *confirm*.

Como mostra a figura, mesmo após ter emitido o pedido de liberação, a entidade usuária pode continuar a receber primitivas de serviço de indicação de transferência de dados (no caso, *S_DATA.indication*), sendo que a desconexão só será efetivada após a recepção da primitiva *S_RELEASE.confirm*.

Um outro ponto de bastante similaridade entre as camadas de Sessão e Transporte é o endereçamento. Da mesma forma que no Transporte, a nível de Sessão é necessário indicar um *SSAP* (*Ponto de Acesso ao Serviço de Sessão* ou *Session Service Access Point*) na fase de estabelecimento de sessão. Normalmente, um endereço de SSAP nada mais é que um endereço de TSAP enriquecido com outras informações.

Ainda, uma diferença entre Sessão e Transporte está nos tipos de dados transmitidos. Como foi visto no caso da camada de Transporte, existem dois tipos de fluxos de dados — os dados normais e os dados expressos (ou urgentes). Já na camada de Sessão, além destes dois tipos de dados, podem ocorrer outros dois — os *dados tipados* (*typed data*) e os *dados de capacidade* (*capacity data*). Estes tipos de dados serão explicados mais adiante, neste documento.

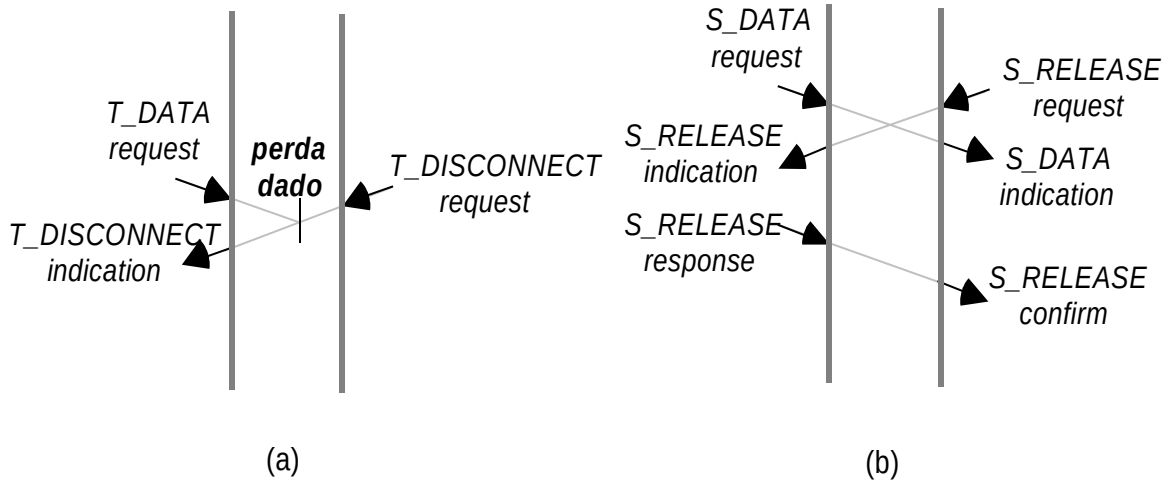


Figura 2.3.45 - (a) liberação abrupta; (b) liberação negociada.

2.3.6.3. A gestão do diálogo

Segundo o modelo OSI, todas as conexões estabelecidas são, a princípio, do tipo *full duplex* (ou seja, bidirecionais simultâneas, conforma mostrado na figura 2.3.46(a)). Por outro lado, existem muitas aplicações nas quais a possibilidade de operação em *half duplex* pode ser mais interessante. Este serviço é implementado a nível da camada de Sessão graças à definição de uma *ficha de dados* (*data token*), como ilustrado na figura 2.3.46(b). A comunicação em *half duplex* é uma opção negociada no momento do estabelecimento de uma sessão. Se esta opção é adotada, deverá ser definido também qual dos usuários envolvidos no diálogo poderá tomar a iniciativa (possuidor da ficha). Quando este usuário terminar a sua transmissão, ele pode ceder a ficha ao usuário par para que ele possa efetuar a sua transmissão. A passagem de ficha de um usuário a outro é implementada através da primitiva *S_TOKEN_GIVE*.

Ainda, se um dos usuários querendo efetuar uma transmissão não possui a ficha naquele instante, ele pode requisitá-la a seu par através da emissão de uma primitiva *S_TOKEN_PLEASE.request*. O usuário receptor da primitiva *S_TOKEN_PLEASE.indication* pode ou não querer ceder a ficha.

Em caso negativo, o usuário que a requisitou não fará outra coisa senão esperar a «boa vontade» do seu interlocutor (ou ainda, enviar dados urgentes que não necessitam a possessão da ficha). A ficha só passa a ter algum sentido no caso de comunicação *half duplex*. Se *full duplex* é a opção adotada, ela não é levada em conta.

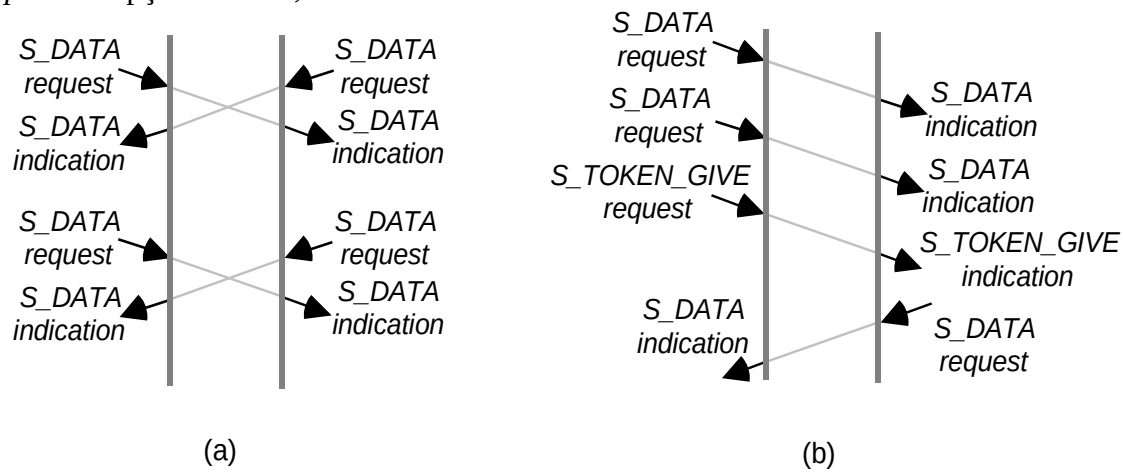


Figura 2.3.46 - (a) comunicação *full duplex*; (b) comunicação *half duplex* com ficha.

2.3.6.4. A sincronização

A camada de Sessão deve cobrir igualmente os problemas relacionados à *sincronização*. Esta tarefa é útil para a manutenção da coerência do estado entre dois usuários interlocutores em caso de erro ou outro problema.

Apesar de que a camada de Transporte tenha, por princípio, a função de cobrir todos os problemas relacionados à retomada após erros ou panes, esta camada cobre unicamente os problemas relacionados à transmissão de dados propriamente dita, não levando em conta os problemas que podem ocorrer nas camadas superiores. Estes erros podem ocasionar perdas de dados que a camada de Transporte é incapaz de detectar (uma vez que esta não é a sua função!).

A camada de Sessão vem, então ao socorro do sistema efetuando a tarefa de sincronização, através da inserção a nível dos dados de *pontos de sincronização*, que permitem manter a sessão num estado correspondendo a um antigo ponto de sincronização. A figura 2.3.47(a) ilustra o procedimento de inserção de pontos de sincronização.

Um exemplo disto pode ser ilustrado na transmissão de um documento via rede, no qual este pode ser decomposto em páginas às quais se pode associar os pontos de sincronização. Neste caso, a *resincronização* vai consistir na retransmissão do documento a partir de uma dada página que estava sendo transmitida quando o problema ocorreu.

A sincronização é implementada da seguinte forma: o usuário emissor insere, nas suas mensagens, pontos de sincronização, cada ponto contendo um número de série. Quando um usuário envia uma primitiva (*request*) para inserir um ponto de sincronização, o outro usuário vai receber uma primitiva de *indication* correspondente, isto ocorrendo de igual maneira no caso de uma resincronização.

É importante notar aqui que a camada de Sessão oferece unicamente as ferramentas para a solução dos problemas de erros e incoerência por sincronização/resincronização. Na realidade, quem ativa estas ferramentas quando da ocorrência de um problema são as entidades das camadas superiores.

O mecanismo de sincronização define dois tipos distintos de pontos de sincronização: os pontos de sincronização *máximos* e *mínimos*.

Os pontos de sincronização *máximos* são utilizados para delimitar trechos da informação denominados *diálogos*, que representam uma decomposição lógica da informação (capítulos de um livro, por exemplo).

Já os pontos de sincronização *mínimos* são utilizados para separar porções menores da informação. No caso de um livro, esta porção poderia ser as páginas dos capítulos. A figura 2.3.47(b) ilustra os pontos de sincronização máximos e mínimos.

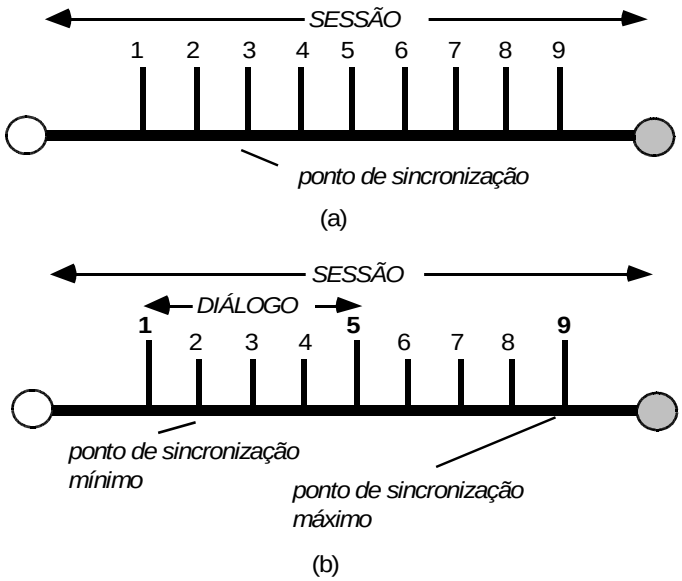


Figura 2.3.47 - (a) Pontos de sincronização; (b) Pontos de sincronização máximos e mínimos.

Uma diferença fundamental entre os pontos máximos e mínimos reside na capacidade de resincronização. No caso de um ponto de sincronização máximo, após uma pane, é possível resincronizar, no máximo, de um ponto de sincronização máximo para trás. A partir deste ponto, é impossível recuperar informação. Isto ocorre porque um ponto máximo é visto como uma fronteira de proteção, cuja informação anterior é garantida ter sido recebida, o que

significa que ela pode ser apagada da extremidade emissora. Isto já não ocorre no caso dos pontos de sincronização mínimos.

Além disso, pelo fato de que os pontos de sincronização máximos são tidos como fronteiras de proteção da informação, estes devem ser explicitamente reconhecidos pelo receptor, o que não é necessário no caso dos pontos de sincronização mínimos.

2.3.6.5. Gerenciamento de atividades

Outra função importante da camada de Sessão é o controle de atividades. Esta tarefa é baseada no conceito de decomposição do fluxo de dados em *atividades*, independentes umas das outras. O conceito de atividade vai depender da aplicação considerada, o usuário sendo o responsável desta definição.

Um exemplo típico de utilização do conceito é a transferência de arquivos, onde cada arquivo deve ser separado, de alguma forma dos demais. A forma de fazê-lo é através da definição, de cada arquivo, como sendo uma atividade, como mostra a figura 2.3.48. Para fazê-lo, antes da emissão de cada arquivo, o usuário deve enviar uma primitiva *S_ACTIVITY_START.request* para marcar o início de uma atividade; isto vai gerar, no lado do receptor, uma primitiva *S_ACTIVITY_START.indication* sinalizando o início do envio de uma nova atividade (neste caso particular, o arquivo).

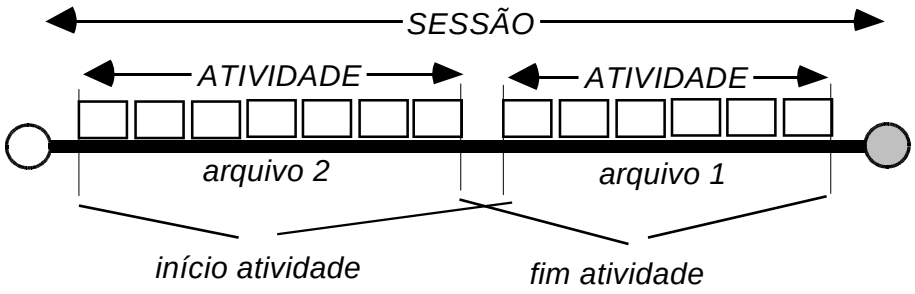


Figura 2.3.48 - Conceito de atividade aplicado a uma transferência de arquivo.

O fim da transferência de um arquivo é marcado, então, pelo envio, por parte do emissor, de uma primitiva *S_ACTIVITY_END.request*. O interesse da definição de atividade é o fato que certas aplicações podem ter a garantia de atomicidade, evitando erros devido a panes ocorridas entre ações de uma mesma atividade.

Um exemplo de aplicação é o de uma operação bancária realizada através de um terminal informatizado. A operação normal se desenrola da maneira seguinte:

- o proprietário da conta emite um comando informando o seu número de conta (e a senha correspondente) — o computador do banco verifica o número e a senha, e bloqueia o código correspondente a fim de evitar o acesso concorrente;
- o proprietário da conta envia um novo comando informando o número da conta para onde o dinheiro deve ser transferido — o computador verifica o número da conta e bloqueia o código correspondente pela mesma razão já apresentada;
- o proprietário, finalmente, emite um terceiro comando informando o montante a ser transferido — o computador efetiva, então, a transferência para a conta destinatária.

Um caso típico deste cenário é aquele em que uma pane (falta de energia, por exemplo) ocorre no terminal sendo utilizado pelo cliente proprietário da conta imediatamente após o primeiro comando. O computador vai bloquear o código correspondente à conta, mas os demais comandos não serão efetuados.

A aplicação do conceito de atividade via camada de Sessão permite solucionar este problema, garantindo a atomicidade da operação. A operação completa pode ser vista como uma atividade. Assim, após a recepção de uma primitiva *S_ACTIVITY_START.indication,* o computador do banco ficaria armazenando as mensagens (de comando) até a recepção de uma primitiva *S_ACTIVITY_END.indication*. Só neste momento, os comandos seriam efetivados pelo computador.

Uma outra propriedade interessante das atividades é a capacidade de ser interrompida e retomada sem a perda das informações. Isto pode ser feito através da primitiva *S_ACTIVITY_INTERRUPT* que permite iniciar uma nova atividade e retomar aquela ao final desta segunda atividade a partir do ponto onde ela tinha sido interrompida.

Um exemplo típico é aquele da transferência de um arquivo de tamanho relativamente grande, no qual, durante a transferência, seja necessário, com relativa urgência, efetuar a consulta a uma base de dados (anuário telefônico, agenda, por exemplo). Este exemplo é ilustrado na figura 2.3.49.

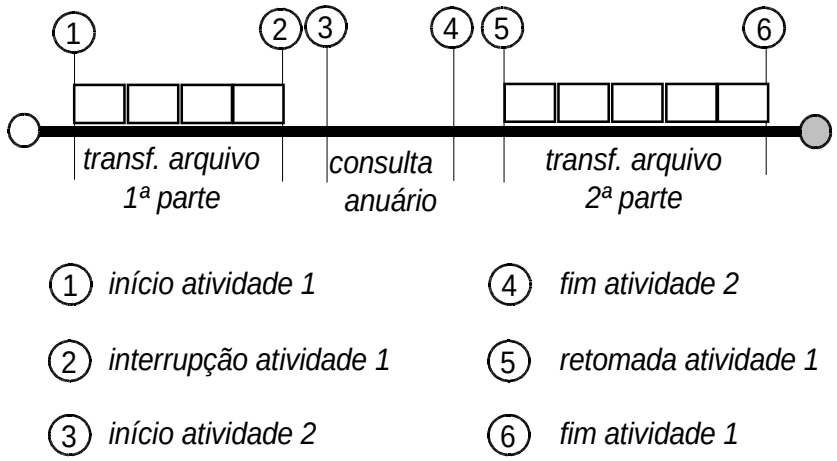


Figura 2.3.49 - Exemplo da interrupção de um atividade para a realização de outra.

2.3.6.6. As primitivas de serviço de Sessão

O quadro a seguir apresenta a lista das primitivas de serviço oferecidas pela camada de Sessão, indicando as classes de primitivas compondo cada serviço (*R - request*, *I - indication*, *Rs - response* e *C - confirm*).

São 58 primitivas organizadas em 7 grupos: estabelecimento de conexão, liberação de conexão, transferência de dados, gerenciamento de fichas, sincronização, gerenciamento de atividades e relatório de anomalias.

Os dois primeiros grupos são relacionados, respectivamente, à inicialização e término das sessões. As primitivas *S_CONNECT*, conduzem, em seus parâmetros, informações tais como: SSAPs dos interlocutores, qualidade do serviço, números iniciais dos pontos de sincronização, etc.

Já os serviços de liberação de sessão podem ser de três tipos: o primeiro, caracterizado pela primitivas *S_RELEASE*, especificando um serviço confirmado de término negociado de sessão (sem perda de dados); os dois outros, para e liberação abrupta de sessão (com eventual perda de dados), caracterizados pelas primitivas *S_U_ABORT* e *S_P_ABORT*, indicando, respectivamente, terminação de iniciativa do usuário (*U - user*) e do fornecedor do serviço (*P - provider*).

O terceiro grupo é caracterizado pelas quatro classes de primitivas para a transferência de dados (*S_DATA*, *S_EXPEDITED_DATA*, *S_TYPED_DATA* e *S_CAPABILITY_DATA*) cada uma para um dos quatro tipos de dados (normais, urgentes, tipados e de capacidade). Os dados normais e urgentes tendo sido discutidos em outras oportunidades, cabe aqui uma apresentação dos dois outros tipos de dados.

Os *dados tipados* são similares aos dados normais do ponto de vista da transferência, a diferença fundamental sendo a liberdade de transmissão sem a necessidade de posse da ficha de dados. Este tipo de dados são utilizados, principalmente, para a troca de mensagens de controle entre os dois usuários.

Os *dados de capacidade* são utilizados para a transferência de mensagens de controle da própria camada de Sessão. O serviço *S_CAPABILITY_DATA*, como se pode ver na tabela, é um serviço confirmado, fornecendo um reconhecimento dos dados de capacidade. Ainda, ao contrário dos dados tipados, estes dados só podem ser transmitidos se a entidade possui a ficha de transferência de dados.

O quarto grupo de serviços é relacionado ao gerenciamento das fichas, que são em número de 4: as fichas de dados (para a transferência em *half-duplex*), as fichas de liberação (para o início de uma liberação negociada), as fichas de sincronização mínima (para a inserção de pontos de sincronização mínimos) e as fichas de sincronização máxima e atividade (para o

gerenciamento das atividades e da sincronização). *S_TOKEN_GIVE* permite efetuar a transferência de uma ou mais fichas à entidade remota, as fichas a transferir sendo indicadas nos parâmetros das primitivas.

O quinto grupo contém os serviços relativos à sincronização, seja para inserção de pontos de sincronização (através dos serviços *S_SYNC_MAJOR* e *S_SYNC_MINOR*, dedicados, respectivamente, à introdução de pontos de sincronização máximos e mínimos), seja para a resincronização a partir de um ponto dado (*S_RESYNCHRONIZE*).

Primitiva	**R**	**I**	**Rs**	**C**	***Função***
ORIENTADO CONEXÃO					
S_CONNECT	•	•	•	•	*estabelecimento de conexão*
S_RELEASE	•	•	•	•	*liberação negociada de conexão*
S_U_ABORT	•	•			*liberação abrupta (usuário)*
S_P_ABORT		•			*liberação abrupta (fornecedor)*
S_DATA	•	•			*transferência de dados normais*
S_EXPEDITED_DATA	•	•			*transferência de dados urgentes*
S_TYPED_DATA	•	•			*transferência de dados tipados*
S_CAPABILITY_DATA	•	•	•	•	*transf. de dados de capacidade*
S_TOKEN_GIVE	•	•			*passagem de ficha de dados*
S_TOKEN_PLEASE	•	•			*pedido de ficha*
S_CONTROL_GIVE	•	•			*passagem de todas as fichas*
S_SYNC_MAJOR	•	•	•	•	*inserção de pto. de sincr. máx.*
S_SYNC_MINOR	•	•	•	•	*inserção de pto. de sincr. mín.*
S_RESYNCHRONIZE	•	•	•	•	*pedido de resincronização*
S_ACTIVITY_START	•	•			*início de uma atividade*
S_ACTIVITY_END	•	•	•	•	*fim de uma atividade*
S_ACTIVITY_DISCARD	•	•	•	•	*abandono de uma atividade*
S_ACTIVITY_INTERRUPT	•	•	•	•	*interrupção de uma atividade*
S_ACTIVITY_RESUME	•	•			*retomada de uma atividade*
S_U_EXCEPTION_REPORT	•	•			*relatório de anomalia- user*
S_P_EXCEPTION_REPORT		•			*relatório de anomalia- provider*
SEM CONEXÃO					
S_UNITDATA	•	•			*transferência de dados (s/ conexão)*

O sexto grupo é dedicado ao gerenciamento das atividades (início, fim, abandono, interrupção e retomada), através dos serviços *S_ACTIVITY_START*, *S_ACTIVITY_END*, *S_ACTIVITY_DISCARD*, *S_ACTIVITY_INTERRUPT* e *S_ACTIVITY_RESUME*.

O sétimo grupo é dedicado à geração de relatórios de anomalia, tanto da parte de um dos usuários (*S_U_EXCEPTION_REPORT*) quanto da camada de Sessão (*S_P_EXCEPTION_REPORT*).

2.3.6.7. Primitivas de Serviço e SPDUs

Os protocolos de Sessão implementados a nível das redes públicas são, normalmente complexos em função da grande diversidade de serviços oferecidos. A cada primitiva de serviço existente, corresponde normalmente uma SPDU, transmitida em função do protocolo implementado. No caso de primitivas com resposta, existe uma SPDU gerada para a resposta. Do conjunto de primitivas apresentadas anteriormente, duas primitivas apenas não são acessíveis aos usuários da Sessão, no caso, aquelas iniciadas pelo fornecedor de serviço (*S_U_ABORT* e *S_U_EXCEPTION_REPORT*).

A cada vez que a camada de Sessão recebe uma primitiva de serviço, ela constrói uma SPDU que vai ser emitida ao destinatário. Ao recebê-la, a entidade de Sessão receptora vai gerar uma primitiva de serviço de indicação correspondente à primitiva de request que gerou a SPDU, os parâmetros da indicação sendo herdados da primitiva de serviço de request.

Quando o serviço é confirmado, como, por exemplo, o serviço *S_CONNECT* ou *S_RELEASE*, a entidade usuária poderá emitir uma resposta positiva ou negativa (aceitação ou não da requisição). Neste caso, a nível da camada de Sessão, vão existir duas SPDUs orientadas à condução das duas possíveis respostas — *ACCEPT* ou *REFUSE* no caso do serviço *S_CONNECT* e *DISCONNECT* ou *NOT FINISHED* no caso do serviço *S_RELEASE*.

A figura 2.3.50 apresenta os formatos das SPDUs, o formato geral sendo apresentado em 2.3.50(a), onde os campos apresentados têm o seguinte significado:

- *SI, Session Identifier*, é um campo de 1 byte que permite identificar o tipo da SPDU;
- *LI, Length Identifier*, ocupa também um byte e é utilizado para indicar o tamanho da SPDU em número de bytes, indo de 0 a 254; caso a unidade de dados possua um tamanho superior a 254, *LI* indica o valor 255 e os dois bytes seguindo este campo vão apresentar o verdadeiro comprimento da SPDU;
- *PARÂMETROS*, é o campo através do qual serão conduzidos os parâmetros associados a uma dada SPDU, a forma de codificação dos parâmetros podendo ser como apresentado em 2.3.50(b), que será descrita mais adiante; ainda, este campo pode ser codificado como mostrado em 2.3.50(c) e 2.3.50(d); estes mostram a possibilidade de reagrupar os parâmetros;
- *DADOS*, é o campo no qual os dados de usuário podem ser conduzidos.

No que diz respeito ao campo *PARÂMETROS*, os diferentes formatos são utilizados, como mostrado em 2.3.50(b) a 2.3.50(d). Em 2.3.50(b), um parâmetro é caracterizado por um *identificador do parâmetro (PI, Parameter Identifier)*, um campo indicando o tamanho do parâmetro (*LI, Length Identifier)* e um campo contendo o valor do parâmetro (*PV, Parameter Value)*. Em 2.3.50(c) e 2.3.50(d) são apresentados formatos mais gerais, onde o primeiro campo identifica o grupo de parâmetros (*PGI, Parameter Group Identifier)*, este campo sendo seguido de um identificador de comprimento do grupo, *LI*. Finalmente, são apresentados os parâmetros de maneira individual, como já apresentados em 2.3.50(b).

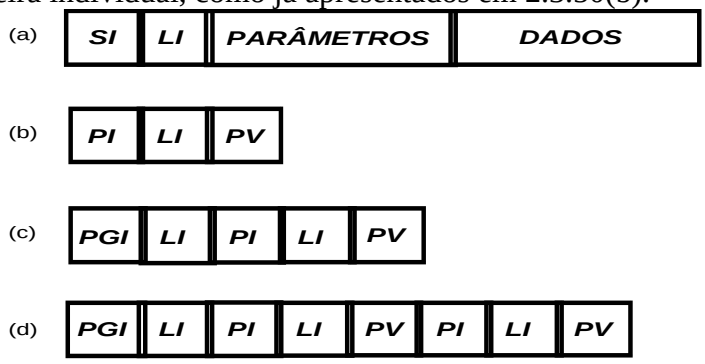


Figura 2.3.50 - Formatos das SPDUs: (a) Formato geral da SPDU; (b) a (d) Formatos do campo parâmetros.

Quando uma SPDU é gerada na camada de Sessão, esta é enviada no forma de uma mensagem à camada de Transporte. É possível, na camada de Transporte, enviar um conjunto de SPDUs a nível de uma mesma mensagem, o que permite otimizar a quantidade de primitivas de Transporte utilizadas — a este mecanismo dá-se o nome de *concatenação*.

Na extremidade de recepção, a entidade de Sessão deve efetuar o processo inverso, quando da recepção de alguma primitiva de serviço de Transporte, recuperando as diversas SPDUs da mensagem recebida. A este mecanismo, dá-se o nome de *segmentação*.

2.3.7. A CAMADA DE APRESENTAÇÃO

Quando foi concebida, a camada de Apresentação era responsável unicamente pelas funções de conversão entre dados representados em ASCII e EBCDIC. Em seguida, novos serviços foram introduzidos de forma que, atualmente, pode-se descrever a tarefa desta camada por tudo o que possa estar relacionado à representação dos dados a serem transmitidos, particularmente as funções de:

- conversão de dados;
- criptografia;
- compressão de dados.

As seções que seguem apresentarão os principais problemas e as soluções encontradas na realização destas funções.

2.3.7.1. O problema da representação dos dados

a) A conversão de dados

A grande diversidade de equipamentos existentes, de diferentes modelos e, principalmente, fabricantes distintos gerou uma série de opções no que diz respeito à forma como os dados poderiam ser representados a nível de cada computador.

Assim nasceram os inúmeros códigos de representação de dados, onde os mais conhecidos são o código ASCII e EBCDIC.

Estas diferenças, que não oferecem nenhum inconveniente quando as diversas máquinas estão operando de maneira isolada, podem representar um verdadeiro quebra cabeças para um engenheiro de sistemas quando duas destas máquinas devem trocar informações.

Se a rede de computadores envolvesse somente computadores de um mesmo tipo, não haveria maiores problemas de representação de dados, exceto que se utilizassem linguagens diferentes para os programas aplicativos. No entanto, as redes de computadores devem suportar a interconexão de equipamentos heterogêneos.

Infelizmente, a forma de representação dos dados não é igual em todos os computadores (inclusive no que se refere aos processadores fabricados pela Motorola, Intel, IBM, etc). Por exemplo, dados na forma alfanumérica seguem em geral uma das duas formas mais usuais de representação já citadas: computadores de grande porte usam código EBCDIC enquanto PCs e Workstations usam código ASCII. Valores numéricos podem ser representados em forma binária (complemento 1, complemento 2 ou sinal-magnitude), em forma alfanumérica (código ASCII ou EBCDIC), em forma BCD (Binary Coded Decimal), etc. Os exemplos seguintes ilustram esta situação:

- números inteiros podem ser codificados de forma binária como:
  - complemento 1: valor negativo é computado por simples inversão de bits

    +127 = 0111 1111 (msb = 0 => +)

    -127 = 1000 0000 (msb = 1 => -)
  - complemento 2: valor negativo é computado por inversão de bits e somando 1 ao resultado

    +127 = 0111 1111

    -127 = 1000 0001 (que seria -126 em complemento 1)

- sinal-magnitude: msb define sinal, demais bits definem valor
  +127 = 0111 1111 (msb = 0 => +)
  -127 = 1111 1111 (msb = 1 => -)
- caracteres alfanuméricos podem ser representados em ASCII ou EBCDIC:
  '534' em ASCII = 0011 0101 0011 0011 0011 0100
  '534' em EBCDIC = 1111 0101 11110011 1111 0100
- IBM 370 usa BCD para decimais:
  534 em BCD = 0000 0101 0011 0100
- Decimal binário Motorola x Intel:
  534 (binário Intel) = 0001 0110 0000 0010
  534 (binário Motorola) = 0000 0010 0001 0110

Isto significa que, quando dois ou mais computadores devem ser conectados via uma rede, um trabalho de conversão deverá ser realizado. No caso de arquiteturas de comunicação que adotem como referência o modelo OSI, a solução situa-se a nível da camada de Apresentação.

Para compatibilizar diferenças de representação de dados, é necessário realizar uma conversão do formato local de cada equipamento para um formato padrão com as características seguintes:

- o mecanismo deve permitir a descrição de estruturas de dados complexas a serem intercambiadas, tais como arrays, records, structs, unions, etc;
- o mecanismo deve suportar uma codificação não ambígua para as instâncias destas estruturas.

Para tratar o problema, a ISO definiu os conceitos de ***sintaxe abstrata***, ***sintaxe concreta*** e ***sintaxe de transferência.***

Por ***Sintaxe Abstrata*** entende-se a especificação da organização de uma estrutura de dados de forma independente da codificação interna utilizada pela máquina. Uma Sintaxe Abstrata torna possível a definição de tipos de dados e atribuição de valores. Por exemplo: em C, a definição de números inteiro (*int*), números reais de ponto flutuante (*float*) e estrutruras complexas (*struct*) são sintaxes abstratas, que correspondem em PASCAL as sintaxes abstratas *integer*, *real* e *record*, respectivamente.

Em lugar da adotar a sintaxe abstrata de uma linguagem já existente, a ISO optou por definir uma linguagem própria para a especificação de estruturas de dados, conhecida como ASN.1 (Abstract Sintax Notation One), da qual veremos exemplos mais a frente.

Já a ***Sintaxe Concreta*** é a especificação de um formato para a codificação de instâncias de estruturas de dados a partir de uma sintaxe abstrata. Os exemplos vistos anteriormente (BCD, ASCII, EBCDIC, Complemento 1, Complemento 2, BCD, etc) representam sintaxes concretas.

A exemplo do que ocorreu com a sintaxe abstrata, a ISO definiu sua própria sintaxe concreta sob o padrão ASN.1 BER (Basic Encoding Rules). O padrão ASN.1 BER adota uma abordagem denominada TLV, com 3 campos:

- **T**ag: rótulo do tipo de dado
- **L**ength: número de bytes do dado
- **V**alue: valor efetivo do dado em sintaxe concreta (padrão ISO 8825)

Finalmente, a ***Sintaxe de Transferência*** é a sintaxe concreta adotada para realizar uma dada transferência de informações entre duas maquinas. Assume-se que, no emissor, a camada de apresentação deve converter os dados a enviar da sintaxe concreta local para a sintaxe de transferência, enquanto, no receptor, os dados recebidos são convertidos da sintaxe de transferência para a sintaxe concreta local. Por exemplo: dados convertidos da sintaxe abstrata PASCAL e sintaxe concreta X para ASN.1 BER na máquina A, são transferidos para a máquina B e convertidos de ASN.1 BER para sintaxe abstrata C e sintaxe concreta Y.

Uma possível sintaxe de transferência é a própria ASN.1 BER, que foi concebida pela ISO com a intenção de exercer este papel.

Uma dada combinação de sintaxe abstrata e sintaxe de transferência constitui um ***Contexto de Apresentação***. No exemplo da figura 2.3.7.1 temos duas máquinas que trocam dados em 3 contextos de apresentação diferentes.

Observe que a sintaxe de transferência adotada é sempre a ASN.1 BER. Note também que uma mesma estação de rede pode fazer uso de diferentes sintaxes abstratas locais em diferentes processos de aplicação (no exemplo da figura 2.3.7.1 são as sintaxes abstratas fornecidas pelas linguagens C e FORTRAN) , de forma que podemos ter mais de um contexto de apresentação para cada máquina.

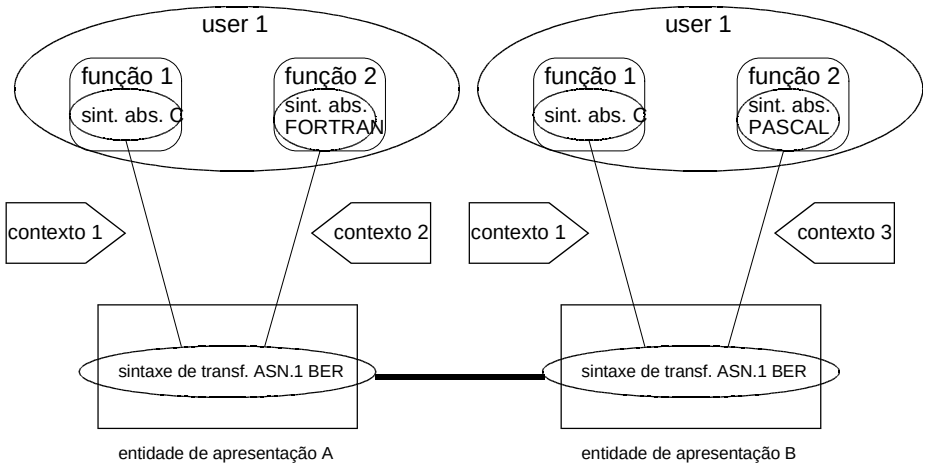


Figura 2.3.7.1 - Contextos de apresentação

A camada de apresentação amplia os serviços de sessão com uma negociação das sintaxes de transferência a usar. No estabelecimento de uma conexão de apresentação são

definidos os Identificadores de Contexto de Apresentação (PCI = Presentation Context Identifier). Cada PCI relaciona uma sintaxe abstrata a uma sintaxe de transferência (define um Contexto de Apresentação). Observe que pode-se ter vários contextos de apresentação em uso em uma mesma conexão, como no exemplo visto anteriormente.

O conjunto de contextos negociados no estabelecimento da conexão formam um DCS (Defined Context Set). Novos contextos podem ser incluídos no DCS durante a operação.

A camada de apresentação converte a representação abstrata de dados vista pela aplicação (por ex.: um texto com figuras) e recebida na forma de PSDU em uma seqüência de bytes na forma de SSDU. Esta transformação de representação é realizada conforme a seqüência mostrada na figura 2.3.7.2.

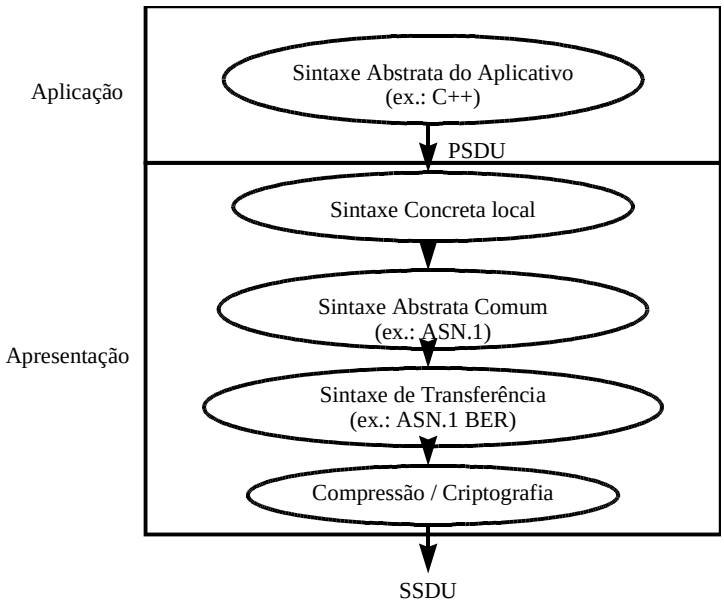


Figura 2.3.7.1 - Transformação de representação

b) A compressão de dados

A utilização de uma rede de comunicação representa normalmente um custo que pode tornar-se não desprezível quando o volume de informação a ser transferida é relativamente grande (quem recebe as contas telefônicas no final do mês sabe disso!).

Numa comunicação telefônica, se pudéssemos evitar ficar tanto tempo conversando e falássemos através de códigos que permitissem representar uma maior quantidade de informação, certamente, uma porção menor dos nossos salários seriam encaminhadas, todo mês, à nossa companhia telefônica.

Da mesma forma, no caso da comunicação entre computadores, se, através da aplicação de técnicas de compressão de dados, o volume de informação puder ser reduzido, o custo da comunicação também o será.

As técnicas de representação estão fortemente ligadas à forma de representação dos dados. Um bom exemplo disto é a representação de números inteiros em binário. Se, por

exemplo, queremos representar números inteiros em 32 bits, uma forma de fazê-lo é através de 4 bytes.

Por outro lado, se 95% dos números inteiros a transmitir estão na faixa de 0 a 250, pode ser mais interessante representa-los por um único byte, utilizando o valor 255 para indicar quando um número tiver de ser representado verdadeiramente em 32 bits. Neste caso, seriam necessário 5 bytes (e não mais 4) para representar o número. Entretanto, o ganho em termos de redução de informação seria altamente compensador, considerando que a quase totalidade da informação poderia ser representada unicamente em 1 byte.

As técnicas de compressão de dados não são unicamente úteis neste caso. Um outro aspecto de aplicação importante destas técnicas é a economia de espaço em disco e fitas magnéticas no caso do armazenamento de arquivos. Algumas destas técnicas serão estudadas mais adiante.

c) A confidencialidade dos dados

O aparecimento das redes de comunicação para a interconexão de computadores vem introduzindo uma série de benefícios nos mais diversos setores da sociedade. Entretanto, as redes de computadores trouxeram também preocupações que não existiam antes do seu aparecimento. Um exemplo disto é o problema da segurança dos dados armazenados num computador.

Quando não existiam as redes, a única forma de se ter acessos a dados confidenciais de uma empresa, laboratório, universidade, etc, seria entrar no prédio onde se localizava o computador e copia-los num disco ou fita magnética ou tirar uma listagem. Dependendo da importância das informações armazenadas (uma base militar ou o computador da Brasil Jet), o prédio poderia ser vigiado dia e noite com direito de acesso a apenas as pessoas de confiança.

Com as redes de comunicação, o acesso aos dados num computador fica bastante facilitado, apesar dos esforços de bloqueio, como por exemplo as *senhas* (*passwords*). Que o digam os computadores do Pentágono, que já foram «visitados» por adolescentes em microcomputadores.

Assim, uma série de cuidados devem ser tomados no que diz respeito à segurança ou à confidencialidade das informações armazenadas num computador e mesmo daquelas trocadas entre computadores, dentre elas:

- a proteção contra leitura de dados por pessoas não autorizadas;
- a proteção contra a inserção de mensagens por elementos não autorizados;
- a verificação da «identidade» do emissor da mensagem;
- tornar possível o envio de documentos com «assinatura eletrônica».

A técnica de criptografia é a solução que permite implementar os pontos acima mencionados. Algumas das diversas técnicas de criptografia conhecidas serão apresentadas ao longo deste capítulo.

2.3.7.2. As primitivas de serviço da camada de apresentação

A tabela a seguir apresenta a lista de primitivas de serviço de Apresentação. Como se pode notar, quase todas as primitivas são idênticas àquelas apresentadas na parte anterior, relativa à camada de Sessão.

No caso dos serviços orientados à conexão, por exemplo, os usuários desta camada, ou seja, as entidades da camada de Aplicação podem abrir uma sessão utilizando a primitiva *P_CONNECT.request*, que se reflete, na camada de Apresentação, pelo envio de uma primitiva *S_CONNECT.request*, já descrita quando da apresentação da camada de Sessão.

De fato, a maior parte das primitivas de serviço de Apresentação atravessam desta forma a camada.

A novidade mostrada na camada são as primitivas *P_ALTER_CONTEXT*, que compõem o serviço de alteração de contexto de Apresentação.

Um contexto é a unidade na qual são reagrupadas as estruturas de dados utilizadas pelas entidades de Aplicação. Ao longo de uma sessão, é possível utilizar um contexto durante um determinado tempo e utilizar um outro num outro momento. O serviço *P_ALTER_CONTEXT* permite fazer a modificação.

Primitiva	**R**	**I**	**Rs**	**C**	***Função***
ORIENTADO Á CONEXÃO					
P_CONNECT	•	•	•	•	*estabelecimento de conexão*
P_RELEASE	•	•	•	•	*liberação negociada de conexão*
P_U_ABORT	•	•			*liberação abrupta (usuário)*
P_P_ABORT		•			*liberação abrupta (fornecedor)*
P_DATA	•	•			*transferência normal de dados*
P_EXPEDITED_DATA	•	•			*transferência urgente de dados*
P_TYPED_DATA	•	•			*transferência de dados tipados*
P_CAPABILITY_DATA	•	•	•	•	*transf. de dados de capacidade*
P_TOKEN_GIVE	•	•			*passagem de ficha de dados*
P_TOKEN_PLEASE	•	•			*pedido de ficha*
P_CONTROL_GIVE	•	•			*passagem de todas as fichas*

P_SYNC_MAJOR	•	•	•	•	*insere pto. de sincr. máximo*
P_SYNC_MINOR	•	•	•	•	*insere pto. de sincro. mínimo*
P_RESYNCHRONIZE	•	•	•	•	*pedido de resincronização*
P_ACTIVITY_START	•	•			*início de uma atividade*
P_ACTIVITY_END	•	•	•	•	*fim de uma atividade*
P_ACTIVITY_DISCARD	•	•	•	•	*abandono de uma atividade*
P_ACTIVITY_INTERRUPT	•	•	•	•	*interrupção de uma atividade*
P_ACTIVITY_RESUME	•	•			*retomada de uma atividade*
P_U_EXCEPTION_REPORT	•	•			*relatório de anomalia (user)*
P_P_EXCEPTION_REPORT		•			*idem (provider)*
P_ALTER_CONTEXT	•	•	•	•	*alteração de contexto*
SEM CONEXÃO					
P_UNITDATA	•	•			*transferência de dados*

A camada de Apresentação viabiliza a negociação sobre que forma de representação será adotada para os dados manipulados pela aplicação. A negociação é baseada no fornecimento, por parte de um usuário, de todas as estruturas de dados que ele deseja utilizar, sendo que o usuário par pode ou não aceitar a proposta.

Durante o diálogo, ainda é possível que novas propostas de estruturas de dados sejam negociadas pelos dois usuários.

2.3.7.3. A notação ASN.1

ASN.1 (*Abstract Syntax Notation One*) é uma sintaxe abstrata que foi definida na ISO para solucionar a problemática da representação dos dados. Isto foi motivado pela necessidade da existência de uma forma de representação de dados que fosse suficientemente flexível e genérica para que pudesse ser utilizada pelas mais diversas aplicações existentes.

ASN.1 é definida pela norma ISO 8824, que apresenta as estruturas de dados previstas na sintaxe; outra norma, ISO 8825, define a *sintaxe de transferência*, ou seja, a forma como as estruturas de dados ASN.1 podem ser mapeadas em binário.

As estruturas de dados ASN.1 foram definidas para prever as complexas estruturas de dados que poderiam ser manipuladas a nível de uma dada aplicação. Um registro ou uma estrutura de dados utilizada numa aplicação bancária pode conter os mais diversos campos, como por exemplo o nome, o endereço, os números de conta, as taxas de câmbio do dia, a data e hora, etc. Em outras aplicações, pode existir a necessidade de troca de estruturas de dados as mais diversas possíveis, sendo que cada aplicação pode ter as suas próprias estruturas de dados. Ainda, algumas estruturas de dados podem ser típicas de certas aplicações, outras específicas a uma determinada empresa ou instituição.

Nos estudos apresentados das camadas inferiores, particularmente na apresentação dos protocolos existentes, muitas estruturas de dados foram apresentadas na forma de figuras, com seus campos sendo apresentados por retângulos. Estas estruturas podem ser formalizadas por registros, records, vetores, etc.

A camada de Aplicação, cujas entidades são os usuários da camada de Apresentação, é caracterizada por muitas estruturas de dados, transmitidas na forma de *APDUs* (*Application Protocol Data Units*), compostas de diferentes campos, cada campo tendo um determinado *tipo* (booleano, inteiro, etc).

A notação ASN.1 permite formalizar os diversos tipos de dados podendo compor os campos de uma APDU, reagrupando-os inclusive num módulo de biblioteca para cada aplicação típica. Desta forma, é possível a uma aplicação enviar uma estrutura de dados à camada de Apresentação, dando a ela o nome da estrutura ASN.1 correspondente. A camada de Apresentação poderá, então, conhecer o tipo e o tamanho de cada campo compondo a estrutura de dados e, desta forma, codifica-los para a transmissão.

A entidade de Apresentação receptora, por sua vez, conhecendo a estrutura ASN.1 dos dados recebidos, pode fazer todas as conversões que sejam necessárias para adaptar à máquina receptora.

O interesse da existência de ASN.1 é que esta notação serve como uma linguagem comum para todas as máquinas. Isto evita que todas as máquinas tivessem de conhecer todas as possíveis formas de representação de cada uma das máquinas com as quais elas teriam de se comunicar. Com ASN.1, basta apenas que cada máquina conheça como codificar seus dados nesta notação.

A norma ISO 8824 define a notação de sintaxe abstrata ASN.1, cujos principais pontos serão apresentados a seguir. Em um primeiro exemplo de como um dado pode ser descrito em ASN.1, vamos fazer um paralelo entre uma estrutura de dados Pascal e uma estrutura de dados ASN.1.

Vamos considerar que pretendemos representar por uma estrutura em Pascal as principais características dos dinossauros. Isto poderia ser feito através de uma estrutura do tipo *record*, onde cada campo teria seu tipo específico e representando uma característica dos dinossauros. Um exemplo disto é mostrado a seguir:

```
type dinossauro = record
    nome       :  array [1..12] of char;
    tamanho    :  integer;
    carnívoro  :  boolean;
    ossos      :  integer;
    descoberta :  integer;
end;
```

No caso de ASN.1, a mesma estrutura de dados poderia ser representada da seguinte forma:

```
dinossauro    ::= SEQUENCE {
      nome OCTET STRING, --12 chars
      tamanho INTEGER,
      carnívoro BOOLEAN,
      ossos INTEGER,
      descoberta INTEGER
}
```

Como se pode notar, não existem muitas diferenças entre a forma de especificar um record em Pascal e uma seqüência em ASN.1. De fato, tipos complexos em ASN.1 são gerados a partir de tipos de base, denominados *tipos primitivos*. Os tipos primitivos da ASN.1 são apresentados na tabela a seguir.

Tipo	***Significado***
INTEGER	*inteiro de valor arbitrário*
BOOLEAN	*verdadeiro / falso*
BIT STRING	*lista de bits*
OCTET STRING	*lista de bytes*
ANY	*conjunto de todos os tipos*
NULL	*nenhum tipo*
OBJECT IDENTIFIER	*nome de um objeto*

Como foi dito, estes tipos podem ser combinados para a definição de estruturas de dados mais complexas, isto pela utilização de construtores de ASN.1, onde os mais utilizados são:

Construtor	***Significado***
SEQUENCE	*seqüência de elementos de diversos tipos*
SEQUENCE OF	*seqüência de elementos de um mesmo tipo*
SET	*conjunto (não ordenado) de elementos de diversos tipos*
SET OF	*conjunto (não ordenado) de elementos de um mesmo tipo*
CHOICE	*escolha de um tipo entre uma lista de tipos*

O construtor *SEQUENCE* permite construir uma estrutura de dados cujos campos sejam de tipos diferentes. Ele é bastante similar a um record em Pascal, como já foi visto no exemplo anteriormente apresentado. *SEQUENCE OF* permite construir vetores de elementos de um mesmo tipo. O construtor *SET* é similar ao *SEQUENCE*, com uma diferença apenas —

os elementos de um *SET* não são ordenados como no caso do *SEQUENCE*. Da mesma forma, *SET OF* é similar a *SEQUENCE OF*, a mesma diferença sendo válida para este caso.

O construtor *CHOICE* permite especificar um dado cujo formato pode assumir diversas opções. Um exemplo da utilização deste construtor é mostrado abaixo.

```
PDU_Comando ::= CHOICE {
    Comando_local,
    Comando_remoto
}
```

Além dos tipos primitivos e construtores, existem ainda os tipos pré-definidos de ASN.1, como por exemplo:

- *NumericString*, que define uma lista de numerais de 0 a 9 mais os brancos;
- *PrintableString*, que define uma lista cujos elementos podem ser as letras maiúsculas, as minúsculas, os números e os caracteres seguintes: **( ) ‘ + - . , / : = ?**
- *GeneralizedTime*, para definir datas e horas.

Existe ainda, em ASN.1, a possibilidade de expressar estruturas de dados onde alguns dos campos possam ou não estar presentes em tempo de execução — são os campos *opcionais*. Isto é representado através da cláusula *OPTIONAL*.

Ainda, pode-se atribuir a um dado um valor inicial (por *default*) através da cláusula *DEFAULT*.

Finalmente, ASN.1 define ainda o conceito de *etiqueta*, que permite identificar um valor associando-o a um dado campo. Para mais detalhes sobre ASN.1, aconselha-se uma consulta à norma ISO 8824 e outros documentos disponíveis sobre esta notação.

2.3.7.4. A compressão de dados

De um ponto de vista genérico, os dados transmitidos através de um canal podem ser vistos como uma seqüência de símbolos $S_1, S_2, ..., S_N$, pertencentes a um dado alfabeto (bits, dígitos decimais, letras, palavras de uma língua, etc).

As técnicas de compressão de dados existentes são baseadas em três diferentes aspectos da representação de dados: a limitação do alfabeto, as freqüências relativas dos símbolos, o contexto de aparecimento dos símbolos. Estes três pontos de vista serão analisados a seguir.

a) Codificação de um alfabeto finito de símbolos

Muitas aplicações são baseadas na representação dos dados através de alfabetos finitos de símbolos. Um exemplo disto são as aplicações relacionadas à utilização de terminais, onde os dados são representados pelo alfabeto ASCII.

Consideremos o exemplo do gerenciamento de uma biblioteca, onde um livro poderia ser identificado pelo seu título, o título de um livro podendo ser expresso em 20 caracteres (ou 140 bits).

Isto significa que, com uma palavra de 140 bits, poderíamos identificar até 2^{140} livros. No entanto, não existem bibliotecas no mundo que possuam esta quantidade de livros. Assim, se, ao invés de identificarmos os livros pelo seu título escolhermos um sistema de numeração, poderemos diminuir em muito a quantidade de bits necessária para a identificação de um livro de qualquer biblioteca.

b) Codificação dependente da freqüência

Em determinadas aplicações, muitos símbolos de um alfabeto são muito mais utilizados que outros. Nos textos em Inglês, por exemplo, a letra “E” aparece 100 vezes mais freqüentemente que a letra “Q”, a palavra “THE” aparecendo 10 vezes mais que “BE”.

Baseados nestas informações, os dados podem ser codificados de tal forma que símbolos ou seqüências de símbolos sabidamente mais freqüentes que outros sejam representados de uma forma especial, simplificada no que diz respeito à quantidade de informação necessária.

As técnicas consistem normalmente em codificar os símbolos mais freqüentes por códigos curtos e os menos freqüentes por códigos mais longos.

c) Codificação baseada no contexto

A técnica apresentada acima é baseada na ocorrência de símbolos ou determinadas seqüências de maneira isolada, como se a probabilidade de ocorrência dos símbolos ou seqüências fosse independente daquela que os precede. Na verdade, estas probabilidades podem ser altamente dependentes da informação precedente. Por exemplo, no caso de um texto em Português, a probabilidade de ocorrer a letra “T” após a letra “Q” é muitas vezes inferior que a probabilidade de ocorrer uma letra “U” após “Q”. A probabilidade de ocorrência de uma letra “P” após uma letra “N” numa palavra é nula, a menos que um erro de ortografia tenha sido cometido.

Uma técnica mais sofisticada que aquela é de levar em conta este aspecto calculando a probabilidade relativa de ocorrência de cada símbolo após os demais símbolos de um alfabeto.

No caso das letras, como o exemplo citado acima, isto vai significar na construção de 26 tabelas, cada uma contendo as probabilidades das letras seguindo uma letra do alfabeto. Se uma forte correlação existir entre os símbolos, a taxa de compressão vai ser muito melhor que aquela obtida com a utilização da técnica apresentada anteriormente.

Uma desvantagem desta técnica é a grande quantidade de tabelas requerida para a definição das probabilidades relativas. No caso de um alfabeto composto de *k* símbolos, serão necessárias k^2 entradas em tabela.

Um método possível de compressão é a organização dos símbolos em tipos, por exemplo, no caso de letras e números, pode-se classificá-los em *maiúsculas*, *minúsculas*, *caracteres numéricos* e *caracteres especiais*. Assim, quatro códigos podem ser associados aos quatro tipos e 28 códigos vão permitir codificar os caracteres. A idéia de base desta técnica é que, a princípio, todo símbolo após uma letra minúscula tem fortes possibilidades de ser também uma letra minúscula, todo símbolo seguindo um caracter numérico é também um caracter numérico, etc. Quando se quer chavear de um tipo de símbolo a outro, basta inserir o código associado ao novo tipo.

Uma técnica similar pode ser utilizada para codificar longas seqüências de dados binários contendo grandes quantidades de *zeros*. Cada símbolo de *k* bits indica quantos zeros separam dois 1's consecutivos. Assim, para tratar grandes faixas de zeros, o símbolo 1 indica que uma faixa de 2^k - 1 mais o valor do símbolo seguinte separa os dois uns. Por exemplo, para a seqüência

0001000001000000100000000000001000000100010000000110000010 1

que contém as faixas de zeros de comprimentos: 3,5,6,14,6,3,7,0,1,5 e 1, pode-se utilizar a codificação seguinte:

011 101 110 111 111 000 110 011 111 000 000 001 101 001

onde a faixa de 14 zeros é codificada por 111 111 000 (ou seja, 7 + 7 + 0 = 14). Neste caso, a compressão permitiu reduzir em 34% o tamanho da informação.

Uma aplicação desta técnica pode ser feita, com algumas modificações, na transmissão digital de imagens de televisão colorida. O sinal é constituídos de uma seqüência de quadros, de 25 a 30 por segundo, cada um contendo um vetor retangular de pontos (pixels) da imagem. Uma imagem de televisão pode ser composta de 1000 linhas de 600 pixels, cada pixel sendo codificado em 16 bits, 15 bits para a cor (com 32 níveis para cada cor) e 1 bit para informação de controle (enquadramento vertical ou horizontal).

Uma codificação direta desta informação necessitaria 600000 pixels por quadro, ou seja, uma taxa de transmissão de 240 a 288 Mbit/s. Com uma codificação binária, seria necessário dispor de uma faixa de freqüência de quase 600 MHz. Uma vez que a transmissão

analógica ocupa faixas de freqüência de apenas 6 MHz, a transmissão digital, sem compressão de imagem, poderia tornar-se inviável.

A solução está no fato que, em transmissão de imagens, existe uma forte probabilidade de que muitos quadros consecutivos sejam idênticos àqueles que os precederam. Desta forma, basta providenciar, a nível dos televisores, uma memória que possa armazenar o último quadro recebido e, assim, transmitir apenas as diferenças entre o quadro anterior e o atual.

2.3.7.5. A criptografia

A criptografia é uma técnica bastante antiga, introduzida principalmente pelas organizações militares, particularmente na troca de mensagens em tempo de guerra. Nessas ocasiões, um problema da criptografia era a dificuldade de decodificação das mensagens criptografadas, muitas vezes tendo de ser feitas nos campos de batalha em condições totalmente adversas. Outro problema era a dificuldade na mudança de uma técnica a outra, particularmente devido à dificuldade em informar o grande número de pessoas que deveria estar consciente da mudança.

O processo de criptografia de uma mensagem é ilustrado pela figura 2.3.51. As mensagens a serem criptografadas são codificadas graças a uma função parâmetrada por uma *chave*, gerando, na saída o texto criptografado ou o *criptograma*. Este texto é transmitido por um mensageiro ou rádio, este podendo ser «escutado» por um espião.

A diferença entre o espião e o destinatário é que, como o primeiro não tem a *chave* de decodificação, ele dificilmente vai poder decodificar a mensagem. Os espiões podem efetuar dois tipos de ação sobre os criptogramas, copiá-los unicamente para conhecer o seu conteúdo (*espião passivo*) ou modificá-los de modo a confundir o destinatário destes (*espião ativo*).

Um princípio das técnicas de criptografia é a suposição de que o espião conhece sempre as técnicas gerais da criptografia, ou seja, que pelo menos o esquema apresentado na figura 2.3.51 é conhecido. Por outro lado, criar a cada vez um método diferente de criptagem a cada vez que um outro é considerado conhecido representa um trabalho considerável. A forma mais eficiente de tentar «esconder o jogo» é através da introdução das *chaves*. As *chaves* consistem de seqüências de caracteres relativamente curtas que permitem escolher um método de criptagem em função do seu valor. A vantagem disto é que, enquanto uma técnica de criptagem pode ser modificada numa escala de anos, uma *chave* pode ser modificada quando se julgar necessário.

Os métodos criptográficos podem ser subdivididos em Métodos Simétricos (onde a chave de codificação é igual a chave de decodificação) e Métodos Assimétricos (onde a chave de codificação é diferente da chave de decodificação).

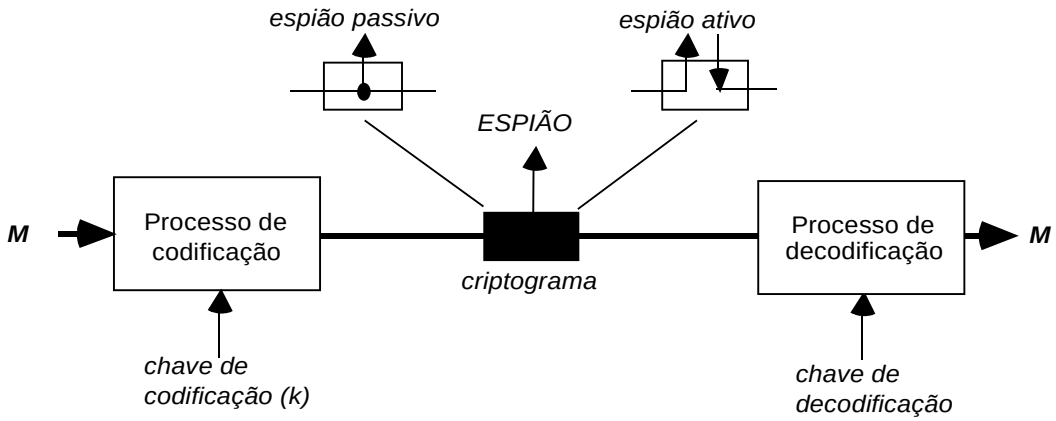


Figura 2.3.51 - O processo de criptografia.

Os métodos de criptografia podem ser organizados em duas principais classes: os *métodos por substituição* e os *métodos por transposição*, cujos princípios serão vistos a seguir.

a) A criptagem por substituição

Nestes métodos, cada letra ou grupo de letras é literalmente substituído por outra letra ou outro grupo de letras. A técnica mais antiga de criptagem é atribuída a Júlio César, que definiu uma técnica onde cada letra do alfabeto seria substituída por uma letra que estivesse 3 passos à sua frente ou seja, *a* seria codificada em *D*, *b* seria codificada em *E*, *c* seria codificada em *F*, etc, e isto de maneira rotativa, de modo que *z* seria codificado em *C*. Desta forma, a palavra *REDE* ficaria *UHGH*, portanto, irreconhecível (exceto, talvez, para um homem das cavernas...).

Esta técnica poderia ainda ser generalizada, pelo deslocamento do alfabeto de *k* passos ao invés de 3. A esta técnica generalizada, dá-se o nome de *permutação circular*.

Uma outra técnica, a *substituição monoalfabética*, é uma extensão à permutação circular, mas, desta vez, as letras podem ser substituídas por qualquer outra. Por exemplo, se o alfabeto

a b c d e f g h i j k l m n o p q r s t u v w x y z

é substituído por

N B V C X W M L K J H G F D S Q P O I U Y T R E Z A,

a palavra *REDE* torna-se, agora, *OXCX*. Com esta técnica, existem 26! (26 fatorial = 4.10^{26}) chaves possíveis. Mesmo que o espião conheça a técnica, como ele não conhece a chave, ele teria dificuldade para descobrir o significado do criptograma. No entanto, esta

dificuldade pode ser diminuída se forem consideradas as propriedades naturais da língua utilizada. Na língua inglesa, por exemplo, existem letras que são mais freqüentemente utilizadas, como por exemplo, as letras *e*, *t*, *o*, os digramas mais freqüentes são *th*, *in*, *er*, e os trigramas mais utilizados são *the*, *ing*, *and* e *ion*. E o trigrama mais utilizado da língua portuguesa? Nos últimos tempos, deve ter sido *CPI*.

Desta forma, a decodificação de um criptograma obtido com a substituição monoalfabética, é feita associando-se a letra mais freqüente à letra mais freqüente da língua considerada, os digramas mais freqüentes àqueles mais freqüentes da língua considerada, etc.

b) A criptagem por transposição

Na criptagem por substituição, apesar da substituição das letras, os códigos mantém a ordem correta das letras nas palavras e frases. Na codificação por transposição, as letras não são substituídas, mas de fato, transpostas de tal forma a esconder o verdadeiro significado do texto.

Uma técnica de criptagem é mostrada na figura 2.3.52, denominada *transposição por colunas*. Nesta técnica as linhas serão trocadas pelas colunas numeradas em função de uma *chave* que é, normalmente, uma palavra não contendo nenhuma letra repetida. No exemplo, a *chave* considerada é a palavra *PESCAR*, onde a ordem das colunas associadas a cada uma das letras vai ser 4-3-6-2-1-5. Assim, um texto como *TRANSFERIR UM MILHÃO PARA A CONTA DE COLLOR* vai ficar *SUAADONRHAALRRIANOTEMPOCFMOCERAILRTL*.

Figura 2.3.52 - Exemplo da transposição por colunas.

Seguem alguns exemplos de sistemas criptográficos muito difundidos na prática:

- DES (Data Encryption Standard): método simétrico desenvolvido pela IBM, com 19 estágios envolvendo transposição e substituição. Padrão adotado pelo governo dos EUA.

- RSA (Rivest, Shamir & Adleman): método assimétrico desenvolvido pelo MIT, onde as chaves de codificação e decodificação (diferentes) são obtidas a partir de operações com números primos grandes. A codificação é feita sobre o valor binário de cada símbolo.
- PEM (Privacy Enhanced Mail): permite realizar a autenticação da origem de uma mensagem recebida via mail na Internet. Trata-se aqui de um sistema de Assinatura Digital: o objetivo não é impedir a decodificação, mas atestar a identidade do emissor (autenticação). O emissor codifica uma parte da mensagem ("assinatura") usando uma chave somente conhecida por ele. A chave de decodificação (diferente) é pública. O receptor decodifica a assinatura e verifica a autenticidade da mesma.

2.3.7.6. O protocolo de apresentação

A camada de Apresentação definida a nível das redes públicas apresenta uma característica essencial que não é propriamente um serviço — é a utilização de ASN.1. Esta facilidade deve ser inteiramente suportada uma vez que, em princípio, muitas das aplicações existentes consideram esta notação.

O protocolo de Apresentação é relativamente simples, pois, como já foi dito, muitas das primitivas de serviço de Apresentação são mapeadas diretamente sobre primitivas de serviço de Sessão. A tabela a seguir apresenta as PDUs definidas no protocolo de Apresentação, estas, sendo reagrupadas em quatro classes distintas: *estabelecimento de sessão*, *desconexão anormal*, *transferência de dados* e *gestão de contexto.*

Na primeira classe de serviços, as PPDUs são trocadas entre entidades de Apresentação para o estabelecimento de uma conexão. O processo inicia pelo envio de uma PPDU CP. Esta PPDU, que foi enviada na forma de uma requisição (request), será recebida pela entidade par na forma de uma indicação (indication), o que justifica as duas marcas nas colunas R e I, respectivamente. Dentre os parâmetros encaminhados via esta PPDU, estão a lista de contextos de Apresentação, expressa por um identificador do contexto (um inteiro), o nome da notação de sintaxe abstrata (identificador de objeto) e a lista das sintaxes de transferência a utilizar. A aceitação ou a recusa da conexão são sinalizadas através das PPDUs CPA e CPR, respectivamente.

Na segunda classe, pode-se notar que não existem PPDUs para a desconexão (liberação negociada). As conexões são liberadas naturalmente quando as sessões são liberadas. *ARU* e *ARP* são PPDUs utilizadas para a liberação abrupta (*ABORT*) da conexão.

Para a classe de transferência de dados, cinco PDUs são previstas, quatro para o envio dos diferentes tipos de dados, *normais*, *urgentes*, *tipados* e *de capacidade*, as PDUs correspondentes, sendo, *TD*, *TE*, *TTD* e *TC*. A PDU *TCC* serve para o reconhecimento de dados de capacidade.

Finalmente, na classe de alteração de contexto, quatro PDUs são previstas, duas delas, AC e ACA, para a requisição e a aceitação de uma alteração de contexto, respectivamente. As PDUs RS e RSA são utilizadas para os serviços de sincronização de Sessão, como descritas na parte dedicada à camada de Sessão.

PPDU	***Significado***	***R***	***I***	***Rs***	***C***
CP	*Pedido conexão de Apresentação*	•	•		
CPA	*Aceitação da conexão*			•	•
CPR	*Recusa da conexão*			•	•
ARU	*Desconexão anormal (iniciativa do usuário)*	•	•		
ARP	*Desconexão anormal (iniciativa fornecedor)*		•		
TD	*Transferência de dados normais*	•	•		
TE	*Transferência de dados urgentes*	•	•		
TTD	*Transferência de dados tipados*	•	•		
TC	*Transferência de dados de capacidade*	•	•		
TCC	*Confirmação de transferência de capacidade*			•	•
AC	*Pedido de alteração de contexto*	•	•		
ACA	*Aceitação da alteração de contexto*			•	•
RS	*Pedido de resincronização*	•	•		
RSA	*Aceitação da resincronização*			•	•

2.3.7.7. Exemplos de camadas de apresentação

Os Protocolos de apresentação são padronizados nas normas ISO 8322 e 8323 e nas recomendações X.216 e X.226 do CCITT. Eles só existem em redes cuja arquitetura segue o modelo OSI. Por exemplo, as redes MAP e TOP usam a camada de apresentação proposta pela ISO, adotando ASN.1 e ASN.1 BER como sintaxe abstrata e sintaxe de transferência, respectivamente.

Inúmeras aplicações padronizadas em redes públicas assumem a disponibilidade da sintaxe abstrata ASN.1 em uma camada abaixo (por exemplo: os serviços FTAM, para transferência de arquivos entre estações de rede, e VTS, para emulação de terminais virtuais).

Já a Internet não tem camada de apresentação (não existe esta camada na arquitetura TCP/IP). A solução adotada para a troca de dados entre computadores com sintaxes concretas locais diferentes é a seguinte: os pacotes da camada de transporte tem um cabeçalho com uma sintaxe concreta fixa (números inteiros de 32 bits em complemento 2) e todos os aplicativos tem que ser escritos de forma a codificar ao menos o cabeçalho das TPDUs desta forma. A sintaxe do resto da informação é problema do software aplicativo (em outras palavras, é problema do desenvolvedor de software).

2.3.8. A CAMADA DE APLICAÇÃO

A camada de Aplicação tem por função o gerenciamento dos programas de aplicação do usuário, que executam em máquinas interligadas via rede e utilizam o sistema de comunicação para a troca de informações. Os programas de aplicação que possuam processos situados em máquinas fisicamente separadas e utilizam o sistema de comunicação vão utilizar-se dos serviços de comunicação oferecidos por esta camada.

Esta camada é a que mantém o contato direto com os usuários da arquitetura de comunicação, abrindo caminho para todos os serviços oferecidos pelas camadas inferiores.

Os elementos compondo a arquitetura da camada de Aplicação vão se utilizar das facilidades oferecidas pela camada de Apresentação para a manipulação e a representação de dados, e os mecanismos de controle de diálogo oferecidos pela camada de Sessão. As interações entre os programas aplicativos permitem modelizar a operação cooperativa entre os sistemas abertos reais, necessitando porém o compartilhamento de uma quantidade de informações que viabilize estas interações, a fim de que o tratamento das atividades seja feito de maneira coerente.

Dentro desta seção serão estudados os principais aspectos relacionados à camada de Aplicação, dando ênfase a sua estrutura e apresentando alguns exemplos de serviços oferecidos. Estes serviços podem ser de uso geral ou serviços específicos para certas classes de aplicações, como veremos mais a frente.

2.3.8.1. Estrutura da camada de Aplicação

a) Definições da norma ISO 9545

Uma aplicação é dita *distribuída* quando os processos de aplicação a ela relacionados residem em sistemas computacionais distintos. A camada de aplicação do RM-OSI cuida da comunicação entre processos cooperantes de uma mesma aplicação distribuida .

A grande diversidade das aplicações podendo ser construídas sobre uma arquitetura de comunicação e a questão da heterogeneidade dos sistemas, fator importante da concepção do modelo OSI, conduziu, no âmbito da ISO, à definição de uma arquitetura unificada para a camada de Aplicação, denominada *ALS* (*Application Layer Structure*) e definida pela norma ISO 9545. Nesta definição, a norma não propõe serviços de Aplicação, mas introduz um conjunto de conceitos relacionados à estrutura da camada, que pode servir como base para a definição de outras normas ou propostas de serviços de Aplicação. Os principais conceitos ali definidos são apresentados a seguir.

Um *Processo de Aplicação* ou *AP* (*Application Process*) representa, de forma abstrata, um elemento de um sistema aberto real que realiza o tratamento de informação no contexto de

uma aplicação particular. Dependendo da natureza da aplicação considerada, um processo de Aplicação pode ter necessidade de trocar informações com outros processos de Aplicação. A atividade de um Processo de Aplicação é caracterizada, conceitualmente por uma instância ou *Invocação de Processo de Aplicação* ou *API* (*Application Process Invocation*). Os processos de aplicação comunicam-se por meio de *Entidades de Aplicação*.

Uma *Entidade de Aplicação* ou *AE* (*Application Entity*) é composta de *Elementos de Serviço de Aplicação (ASE)* e executa os *Protocolos de Aplicação*. Um mesmo Processo de Aplicação pode reagrupar diversas Entidades de Aplicação de mesmo tipo ou de tipos diferentes. As capacidades de comunicação oferecidas por uma AE são ativadas por meio de uma *Invocação de Entidade de Aplicação* ou *AEI* (*Application Entity Invocation*).

Um *Elemento de Serviço de Aplicação* ou *ASE* (*Application Service Element*) compreende um par serviço-protocolo normalizado que pode constituir uma Entidade de Aplicação. Ele corresponde a um subconjunto das funções ou facilidades de comunicação oferecidas para o suporte de cooperação entre Entidades de Aplicação. Estas funções e facilidades são definidas pela especificação de um conjunto de *Unidades de Dados de Protocolo de Aplicação* ou *APDUs* (*Application Protocol Data Units*) assim como os procedimentos associados à sua utilização, definindo o *Protocolo de Aplicação* entre dois ASEs.

A comunicação entre entidades de Aplicação pares deve ser suportada por alguma forma de relação que permita a troca de informações de controle dos protocolos de Aplicação. Esta forma de relação, que corresponde a noção de conexão entre entidades pares, é definida como uma *Associação de Aplicação*. Uma *Associação de Aplicação* ou *AA* (*Application Association*) é uma relação cooperativa estabelecida entre duas AEIs para a troca de informações. Uma ou mais AAs são criadas quando duas AEIs devem trocar informações no contexto de uma dada aplicação. Isto significa que uma AEI pode manter, simultânea ou seqüencialmente, mais de uma AA com uma ou diversas AEIs.

Finalmente, um conceito importante é o de um *Contexto de Aplicação*, que corresponde ao conjunto de regras e funções a serem implementadas ao longo da duração de uma Associação de Aplicação para a gestão das trocas de informações entre AEIs. Este conceito corresponde a definição de uma semântica e sintaxe unificadas para os dados trocados entre duas AEIs.

Este conceitos estão ilustrados de forma esquemática na figura 2.3.8.1.

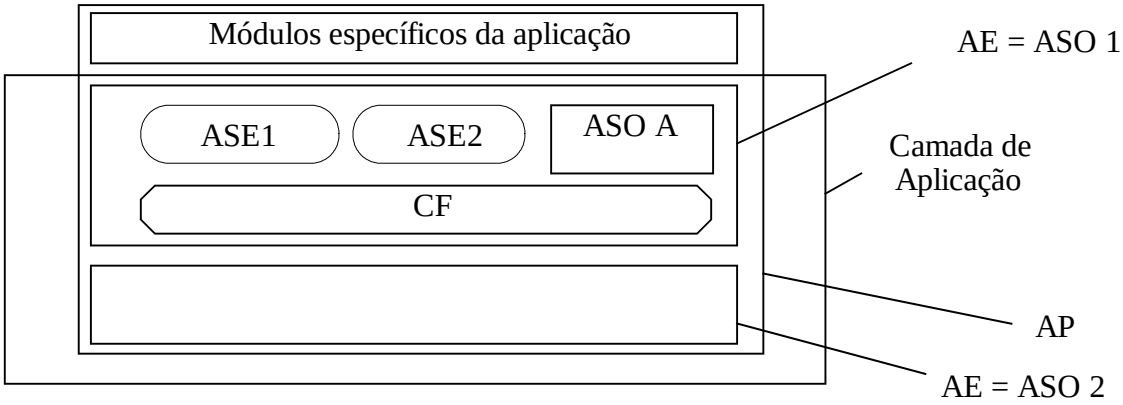


Figura 2.3.8.1 - Conceitos da camada de aplicação

b) Elementos da camada de aplicação

Na prática, a camada de aplicação é composta de uma série de *entidades de aplicação* (*Application Entities, AE*) , conhecidas como *elementos de serviço de aplicação* (*Application Service Elements, ASE*) e *elementos de usuário* (*User Elements, UE*).

Um UE fornece uma interface entre o usuário e os diversos serviços de processamento de informação da camada de aplicação. O UE pode ser visto como uma biblioteca de procedures e functions a ser linkada com os processos de aplicação (APs) e não costuma ser padronizado. Cada AE terá usualmente uma única UE.

Por outro lado, normalmente uma AE será composta de diversos ASEs. Existem ASEs que oferecem serviços gerais, que propiciam a transferência de informações entre APs independentemente da natureza da aplicação (ex: definição de contexto, sincronização entre APs, etc). Tais ASEs de uso geral são denominados *"Common" Application Service Elements* (*CASE*). Exemplos de serviços providos pelos CASEs atualmente disponíveis são:

- serviços de estabelecimento de conexões entre APs (ACSE = Association Control Service Element);
- serviços de disparo de operações em ASEs remotos (ROSE = Remote Operation Service Element);
- serviços de implementação de ações atômicas (CCR = Commitment, Concurrency and Recovery).

Outros ASEs oferecem serviços específicos para determinadas aplicações em particular e por isso são denominados *"Specific" Application Service Elements* (*SASE*). Exemplos de serviços providos pelos SASEs atualmente disponíveis são:

- serviços de terminal virtual (VTS = Virtual Terminal Services);

- serviços de transferência e manipulação de arquivos (FTAM = File Transfer, Access and Menagement);
- serviços de submissão remota de tarefas (JTM = Job Transfer and Manipulation);
- serviços de manuseio de mensagens (MHS = Message Handling Services);
- serviços de mensagens da manufatura (MMS = Manufacturing Message Services).

É importante observar que novos ASEs, diferentes dos citados acima, estão sendo atualmente definidos e padronizados pela ISO, CCITT e outras entidades.

A figura 2.3.8.2 ilustra a relação entre estes elementos de forma esquemática.

É usual dispor-se de um CASE e diversos SASEs em uma camada de aplicação tipica, como ilustrado na figura.

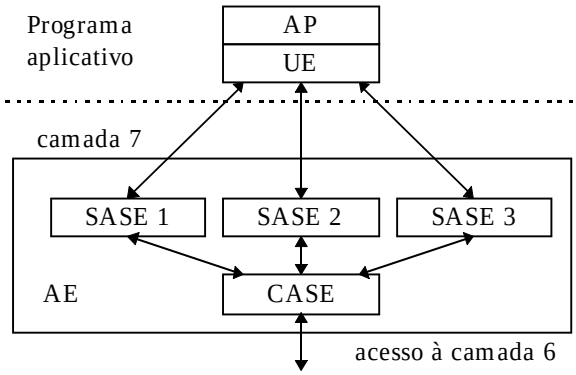


Figura 2.3.8.2 - Os elementos da camada de aplicação

A seguir, apresentamos alguns exemplos de ASEs e seus serviços.

2.3.8.2. Exemplos de CASE (ASEs para serviços comuns)

a) Serviços de gestão de Associações - ACSE

A gestão de conexões é um dos problemas a serem resolvidos pela camada de Aplicação, de forma a evitar que cada aplicação desenvolvida tenha de resolver esta questão à sua maneira. Assim, a nível desta camada, foi definido um *Elemento de Serviço de Controle de Associação,* o *ACSE* (*Association Control Service Element*) que assume esta tarefa.

Como a totalidade das aplicações conhecidas requer, normalmente os serviços de gestão de conexão, este elemento de serviços deve estar sempre presente a nível dos contextos de Aplicação.

Os serviços oferecidos pelo ACSE são definidos pelas primitivas apresentadas na tabela a seguir.

Primitiva	***Significado***

A_ASSOCIATE	*Estabelecimento de conexão*
A_RELEASE	*Liberação negociada de conexão*
A_U_ABORT	*Liberação abrupta de conexão (usuário)*
A_P_ABORT	*Liberação abrupta de conexão (provedor)*

A_ASSOCIATE é um serviço confirmado que permite estabelecer uma associação a partir de um elemento de serviço de aplicação. Dentre os parâmetros da primitiva de serviço, é enviada a proposição de contexto de Aplicação (isto é, a semântica e sintaxe dos dados a serem intercambiados). Em caso de aceitação por parte do receptor da indicação, a associação será, então, estabelecida.

O serviço *A_RELEASE* é um serviço confirmado que permite liberar, de maneira ordenada, uma associação estabelecida entre duas AEIs, caracterizando uma liberação negociada, como definido na parte relativa à camada de Sessão.

A_ABORT é um serviço não confirmado que permite «abortar», ou liberar de forma abrupta uma associação existente. Como já descrito anteriormente, neste caso podem ocorrer perdas de informações sendo trocadas através da associação. A ativação deste serviço é de iniciativa do usuário do ACSE. *A_ABORT* pode ser ativado pelo iniciador do serviço *A_ASSOCIATE*, no caso em que não exista concordância a respeito do contexto de aplicação a utilizar. *A_P_ABORT* é o serviço que efetua o mesmo que *A_ABORT*, sendo que, desta vez, a iniciativa é do fornecedor do serviço.

O ACSE tem acesso a serviços de duas das camadas inferiores, particularmente, a camada imediatamente inferior, de Apresentação, e a camada de Sessão.

No que diz respeito aos serviços de Apresentação, uma primeira correspondência que se pode estabelecer é o de uma *associação* com uma *conexão de Apresentação*. Deste ponto de vista, existem uma correspondência de 1 para 1, ou seja, uma associação de Aplicação corresponde a uma conexão de Apresentação.

O ACSE necessita utilizar unicamente os serviços de base da camada de Apresentação, ou seja, os serviços *P_CONNECT*, *P_RELEASE* e *P_U_ABORT*. O iniciador do serviço *A_ASSOCIATE* deve determinar o contexto de Apresentação pretendido.

b) Serviços para implementação de ações atômicas - CCR

O CCR (Commitment, Concurrency and Recovery) é outro elemento comum de serviço (CASE), que coordena a operação entre unidades distribuidas de forma confiável. Ele permite definir serviços compostos de um conjuntos de ações com a propriedade de “atomicidade”, isto é, o serviço só é considerado completo se todas as ações que o compõe foram executadas até o fim.

Um exemplo prático de situação onde um serviço atômico deve ser empregado seria uma transferência bancária. Suponhamos que João, que é cliente do banco A, pede para fazer

uma transferencia de uma quantia X para seu amigo José, que é cliente do banco B. Para completar a transação, algumas ações tem que ser executadas: o computador do banco A deve descontar a quantia X da conta de João e enviar uma mensagem para o banco B; quando receber a mensagem do banco A, o computador do banco B deve depositar a quantia X na conta de José e enviar uma mensagem de acknowledge ao banco A, indicando que a operação foi completada. Até ai tudo bem, mas suponha que um dos bancos faça a sua parte do serviço, mas o outro não, devido, digamos, a uma falha local ou a um problema na rede. Se, por exemplo, o banco B recebeu o pedido de depósito, executou-o, mandou a mensagem de acknowledge mas esta não é recebida, o banco A pode pedir novamente a execução do serviço, o que resultará, para alegria de José, em uma repetição da operação de depósito. Isto requer um protocolo que só execute a transação de forma completa ou absolutamente não execute. O CCR opera segundo uma politica cliente-servidor, com um protocolo conhecido como *two-phase commit* ("submissão em duas fases").

Na primeira fase, quando um cliente pede a um conjunto de servidores a execução de um serviço atômico, cada servidor verifica a disponibilidade local de atender ao pedido. Se o pedido puder ser atendido, ele coloca a ordem de serviço correspondente em uma memória não volátil local (que não se apaga se o sistema cair, como por exemplo um disco rígido) e envia ao cliente um acknowledge positivo. Se o servidor não puder atender ao pedido, ele envia ao cliente um acknowledge negativo e não faz mais nada.

O cliente espera os acknowledges de todos os servidores envolvidos no pedido. Se um deles for negativo, ele aborta o pedido e avisa a todos os demais servidores que o serviço foi cancelado. Se todos os acknowledges forem positivos, o cliente envia uma segunda ordem a todos os servidores confirmando que o serviço deve ser executado. Os clientes então executam o serviço e somente após isto apagam a ordem da memória não volátil local.

As primitivas de serviço do CCR são mostradas na tabela a seguir.

Primitiva	**emitida por**	**significado**
C-BEGIN	Cliente	Inicio de ação atômica
C-PREPARE	Cliente	Fim pedido de serviço
C-READY	Servidor	servidor pronto para ação
C-REFUSE	Servidor	servidor não pronto p/ação
C-COMMIT	Cliente	Submeter ação
C-ROLLBACK	Cliente	Cancelar ação
C_RESTART	Ambos	notificação de queda

Qualquer serviço atômico é iniciado com C-BEGIN. Quando o cliente terminou de enviar todos os pedidos de ações que fazem parte do serviço atômico, ele envia um C-PREPARE. Em resposta a este último comando, os servidores respondem com C-READY ou C-REFUSE, conforme sua capacidade de atender ao pedido ou não. Se todas as respostas

forem positivas, o cliente envia o comando C-COMMIT; em caso contrário, ele envia um comando C-ROLLBACK. C-RESTART só é usado para avisar aos demais participantes do serviço atômico que uma estação caiu mas está de volta a operação normal.

2.3.8.3. Exemplos de SASEs (ASEs para serviços específicos)

a) Serviços de terminal virtual - VTS

Os terminais são atualmente de grande importância para a interação entre usuário e computador. Existem vários tipos diferentes de terminais, totalmente incompatíveis entre si. Por exemplo, a apresentação dos dados na tela pode seguir 3 modos básicos:

- modo “rolo”: onde as linhas anteriores rolam para cima quando uma nova linha aparece, eventualmente desaparecendo da tela. Estes terminais usualmente não oferecem capacidade de edição e não tem inteligência local.
- modo “página”: que permite que o usuário movimente o cursor para cima e para baixo, editando qualquer parte da página. Estes terminais já dispõe de inteligência local (microprocessador interno dedicado).
- modo “formulário”, oferecem na tela um formulário com campos específicos que podem ser editados pelo usuário (usados por exemplo em caixas eletrônicos de bancos).

O problema que se coloca aqui é o de como trocar dados entre computadores que possuem terminais diferentes. Para resolver problemas de incompatibilidade entre terminais diferentes na conexão entre terminais e outros recursos computacionais via rede, foram propostas duas abordagens:

- serviço de terminal paramétrico: aqui as diferenças entre terminais são parametrizadas, isto é, cada terminal é associado a um conjunto de parâmetros que definem suas características operacionais. Este método requer um *concentrador de terminais*, cuja função é definir os parâmetros adequados a cada terminal ligado a ele. Existem padrões para estes parâmetros, tais como o PAD (Packet Assembler/Disassembler) da CCITT (conhecido também como conversor X.25). O inconveniente desta técnica é que hoje existe uma variedade muito maior de tipos de terminais do que havia na época da sua criação, alguns deles requerendo uma quantidade muito grande de parâmetros para definir todas as suas características operacionais.
- serviço de terminal virtual: aqui as funções empregadas para acessar um dado tipo de terminal são abstraídas em um modelo, usualmente um objeto ou classe, que recebe os dados brutos a serem apresentados na tela e realiza a conversão destes para o formato requerido pelo terminal local. Este método é mais geral e atende a todos os tipos de

terminais existentes. Ele poderá também acomodar todos os novos tipos de terminais que forem surgindo, o que requer apenas a criação de um novo objeto que herde as propriedades básicas de algum outro terminal virtual parecido.

A ISO propôs um conjunto de serviços de terminal virtual que visa atender as necessidades de acesso a terminais para aplicações relativamente simples (ex.: edição de texto, interação com o sistema operacional, entrada de dados, etc). Estes serviços são parte de um SASE denominado VTS (Virtual Terminal Service), que inclui também um protocolo denominado VTP (Virtual Terminal Protocol). Este SASE define uma série de objetos básicos, cujos métodos (primitivas de serviço) permitem a execução das operações indicadas na tabela a seguir.

Primitiva	**Operação**
VT-ASSOCIATE	estabelece conexão entre terminais ou entre um terminal e um processo de aplicação (AP) via ACSE
VT-RELEASE	desfaz conexão de forma ordenada via ACSE
VT-U-ABORT	desfaz conexão de forma abrupta pelo User via ACSE
VT-P-ABORT	desfaz conexão de forma abrupta pelo Provider via ACSE
VT-CONTEXT-SWITCH	altera contexto (semântica e sintaxe dos dados)
VT-START-NEG	inicia negociação de parâmetros com parceiro
VT-END-NEG	termina negociação de parâmetros com parceiro
VT-NEG-INVITE	convida parceiro a sugerir parâmetros de negociação
VT-NEG-OFFER	oferece uma sugestão de parâmetros ao parceiro
VT-NEG-ACCEPT	aceita parâmetros sugeridos pelo parceiro
VT-NEG-REJECT	rejeita parâmetros sugeridos pelo parceiro
VT-DATA	transfere dados estruturados de forma sequenciada
VT-DELIVER	possibilita controlar dados entregues ao parceiro por meio de uma marca de entrega
VT-ACK-RECEIPT	reconhece recebimento de marca de entrega
VT-GIVE-TOKEN	passa direito de envio da dados ao parceiro (token)
VT-REQUEST-TOKEN	requisita direito de envio de dados ao parceiro

b) serviços de manuseio de mensagens - MHS

O MHS (Message Handling Service) é um SASE que provê serviços para intercâmbio de mensagens eletrônicas entre APs. É, portanto, um serviço de mail eletrônico, semelhante ao usado na Internet (só que lá se utiliza o SMTP, Simple Mail Transfer Protocol). O MHS é definido na recomendação X.400 da CCITT.

Quando um AP deseja enviar uma mensagem a outro, ele submete a mensagem a um *Agente de Transferência de Mensagens* (*MTA, Message Transfer Agent*), que entrega a

mensagem a outro MTA no lado receptor. Este, por sua vez, entrega a mensagem ao AP destinatário. As mensagens tem um formato padronizado, composto de um *envelope* (contendo dados como o endereço do destinatário e do emissor) e o *conteúdo*, que é a mensagem em si.

Algumas das primitivas básicas colocadas a disposição dos APs pelo MHS são as seguintes:

Primitiva	**Operação**
M-LOGON	estabelece conexão do AP na rede
M-SUBMIT	submete mensagem para envio no lado emissor
M-DELIVER	entrega mensagem ao destinatário no lado receptor
M-NOTIFY	notifica emissor da recepção da mensagem
M-LOGOFF	desfaz conexão

c) Serviços de acesso e transferência de arquivos - FTAM

Em 1987, foi publicada pela ISO a versão definitiva da norma de serviços de acesso e transferência de arquivos, *FTAM* (*File Transfer, Access and Management*), com o número ISO 8571.

A norma cobre um dos assuntos mais importantes da interconexão de sistemas, uma vez que ela normaliza o protocolo que permite oferecer os serviços de acesso e transferência de arquivos.

FTAM permite a transferência do todo ou de parte de um arquivo, ao destinatário ou a uma estação (sistema) distante, oferecendo ainda serviços de controle de erros, retomada imediata ou não, controle de acesso a arquivos (acesso seletivo, proteção, etc.) e modificação dos atributos de um arquivo.

O tratamento de arquivos via FTAM é baseado no conceito de *arquivo virtual*, que permite representar, para os serviços FTAM, o arquivo real. A relação entre estes elementos é apresentada na figura 2.3.8.3.

A noção de arquivo virtual permite tornar transparente ao usuário as diferentes formas de armazenamento e os métodos de acesso dos sistemas de arquivo real. FTAM permite estabelecer as relações entre o sistema de arquivos real e os sistemas de arquivos virtuais.

Um *servidor virtual de arquivos* reagrupa todas as informações relativas aos arquivos, permitindo conhecer o estado de um arquivo num dado instante. O modelo definido para o servidor de arquivos virtuais contém atributos os mais genéricos possíveis quanto à representação dos aspectos de um servidor de arquivos reais.

Este é modelado como uma entidade endereçável com a qual um usuário remoto pode se comunicar, utilizando os serviços FTAM. A princípio, um número arbitrário de usuários pode ser associado a um mesmo servidor virtual de arquivos num dado instante. Ainda, um

servidor virtual de arquivos pode gerenciar um número arbitrário de arquivos, onde cada arquivo é caracterizado por atributos tais como: o nome, ações possíveis sobre o arquivo (leitura, inserção, substituição, etc...), controle de acesso (somente leitura, somente escrita, etc...), tamanho do arquivo, identidade do criador, data e hora de criação, identidade do último modificador, data e hora de modificação, etc.

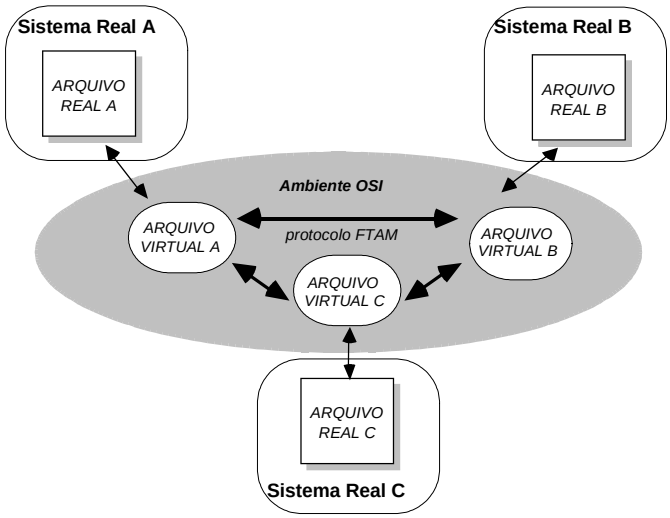


Figura 2.3.8.3 - Relação entre arquivos virtuais e arquivos reais.

A seleção de um arquivo é caracterizada por duas etapas: o estabelecimento de uma associação entre o usuário querendo selecioná-lo e o servidor virtual de arquivos; o fornecimento da identidade do arquivo a selecionar.

A estrutura de acesso aos arquivos num servidor virtual de arquivos assume a forma arborescente, coerente com as estruturas mais genéricas dos sistemas reais de arquivos. Esta estrutura é ilustrada pela figura 2.3.8.4, onde se pode ver um único nó raiz (**R**), nós internos (**A**, **E**) e folhas (**B**, **C**, **D**, **F**) conectadas por arcos dirigidos. Um nó da árvore pode pertencer a um único nível, podendo, por sua vez, servir de acesso à sua subárvore, conhecida como uma *Unidade de Dados de Acesso a Arquivo* ou *FADU* (*File Access Data Unit*). O conteúdo de um arquivo pertencente ao servidor de arquivos é mantido numa ou mais *Unidades de Dados* ou *DU* (*Data Unit*).

Uma *DU* é um objeto de dados tipado (escalar, vetor, conjunto, etc...), contendo elementos de dados atômicos denominados *Elementos de Dados*, aos quais é normalmente associada uma sintaxe abstrata (caracter, byte, inteiro, etc...).

Os serviços associados a um servidor virtual de arquivos são acessados via primitivas de serviço, que serão apresentadas a seguir.

Os serviços oferecidos pelo FTAM para o acesso e a transferência de arquivos são organizados em unidades denominadas *regimes*. Os regimes correspondem a etapas da manipulação de arquivos, a tabela a seguir apresenta o conjunto de primitivas associado a cada um deles.

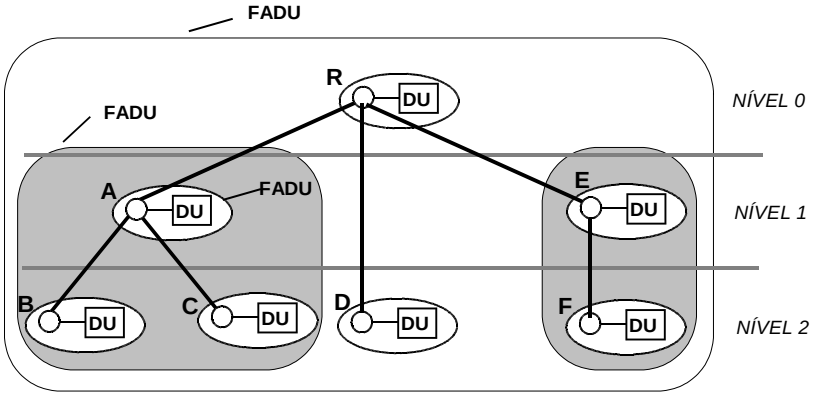


Figura 2.3.8.4 - Estrutura de acesso do servidor virtual de arquivos.

REGIME: CONEXÃO DE APLICAÇÃO	**estabelece a autorização e as informações necessárias à operação do servidor de arquivos**
F_INITIALIZE	*inicialização da associação (via ACSE)*
F_TERMINATE	*terminação negociada de associação (via ACSE)*
F_ABORT	*terminação abrupta da associação (via ACSE)*
REGIME: SELEÇÃO DE ARQUIVO	**identificação de arquivos que serão manipulados**
F_SELECT	*seleção de um arquivo existente*
F_CREATE	*criação de um arquivo*
F_DESELECT	*liberação de um arquivo*
F_DELETE	*eliminação de um arquivo*
F_READ_ATTRIB	*leitura dos atributos do arquivo*
F_CHANGE_ATTRIB	*modificação de atributos de um arquivo*
REGIME: ACESSO A ARQUIVO	**viabiliza as transferências de dados, incluindo as capacidades necessárias para a transferência**
F_OPEN	*abertura de um arquivo*
F_CLOSE	*fechamento de arquivo*
F_LOCATE	*localização de um arquivo*
F_ERASE	*apaga conteúdo de um arquivo*
REGIME: TRANSFERÊNCIA DE DADOS	**etapa de transferência de dados de e para arquivos**
F_READ	*leitura do conteúdo de um arquivo*
F_WRITE	*escrita sobre um arquivo*
F_DATA	*transferência de dados*
F_DATA_END	*transferência de fim de dados*
F_TRANSFER_END	*fim de transferência de dados*

2.3.8.4. Visão geral dos ASEs disponíveis

O FTAM, como foi apresentado, constitui-se em um importante elemento de serviço de aplicação (ASE) para as aplicações que exijam manipulação ou transferência de arquivos, sejam aplicações de automação de escritório ou de automação da manufatura. No que diz

respeito a esta segunda classe de aplicações, podemos destacar como um importante elemento de serviço de aplicação, o ASE denominado MMS (Manufacturing Message Services), proposto para a camada de aplicação da rede MAP, que será apresentado mais adiante, quando tratarmos de redes industriais.

Além dos ASEs vistos anteriormente, a ISO e a CCITT definiram uma série de outros elementos de serviço. A tabela a seguir apresenta, de forma resumida, os principais padrões utilizados a nível de aplicação no modelo OSI.

Nome do ASE	Norma ISO	Recomendação CCITT
ACSE (Association Control Service Element)	ISO 8649 / 8650	X.217 / X.227
RTSE (Reliable Transfer Service Element)	ISO 9066	X.218 / X.228
ROSE (Remote Operations Service Element)	ISO 9072	X.219 / X.229
CCR (Commitment, Concurrency and Recovery)	ISO 9804 / 9805	
MHS (Message Handling Services)	ISO 10021	X.400 / X.420
EDS (Electronic Directory Services)	ISO 9594	X.500 / X.521
FTAM (File Transfer, Access and Management)	ISO 8571	
VTS (Virtual Terminal Services)	ISO 9040 / 9041	
JTM (Job Transfer Manipulation)	ISO 8831 / 8832	
DTP (Distributed Transaction Processing)	ISO 10026	
MMS (Manufacturing Message Services)	ISO 9506	
CMIP (Common Management Information Protocol)	ISO 9505 / 9596	X.710 / X.711
RDA (Remote Database Access)	ISO DIS 9579	

2.4. A INTERCONEXÃO DE REDES

2.4.1. INTRODUÇÃO

O problema da interconexão aparece no momento em que dois usuários que necessitam dialogar não estão conectados necessariamente à mesma subrede. Um exemplo típico deste problema é encontrado nas propostas de comunicação em automação industrial. Como é sabido, tanto as empresas como as arquiteturas de comunicação de dados obedecem, geralmente a uma organização hierárquica. As fábricas podem ser divididas em células e áreas, enquanto o setor administrativo pode ser dividido em departamentos, sessões, divisões, coordenadorias, etc.

Estas divisões são normalmente baseadas no estabelecimento de critérios funcionais, tendo as funções de cada divisão significado especial na empresa.

As razões que podem conduzir a sistemas integrando diferentes subredes podem ser de naturezas as mais diversas:

- é muito mais **econômico** interligar computadores geográficamente próximos através de uma rede local e compartilhar uma interface única com uma rede externa do que conectá-los, cada um deles a esta mesma rede externa;
- é **tecnologicamente** limitante a interconexão (via uma rede local) de um grande número de computadores separados por grandes distâncias; por exemplo, os diversos computadores localizados em diferentes prédios de um campus de universidade. Neste caso, é mais interessante interligar os computadores de cada prédio por uma rede local sendo que as diversas redes locais serão interconectadas;
- o **desempenho** e a **confiabilidade** de um sistema podem ser fortemente aumentados se, ao invés de interligar um grande número de estações por uma única rede, esta for particionada em duas ou mais redes; cada rede local associaria aquelas estações que possuam maior tráfego entre elas, diminuindo assim o tráfego no suporte de transmissão, sendo que elementos de interconexão das diversas redes garantiriam a comunicação entre as estações conectadas a diferentes redes;
- é **funcionalmente** mais interessante interligar estações que realizem trabalhos pertencentes a atividades compatíveis por redes locais adequadas ao perfil destas atividades; as diferentes redes associadas a cada nível de atividade continuariam a permitir a comunicação entre estações pertencentes a diferentes atividades através dos elementos de interconexão.

O problema da interconexão de redes é derivado de três questões importantes:

- a primeira está relacionada à função de roteamento, dado que dois equipamentos envolvidos num diálogo podem não pertencer à mesma subrede;

- a segunda está relacionada com a possibilidade (bastante realista) de que duas subredes interconectadas, apesar de possuirem arquiteturas semelhantes, não implementem os mesmos protocolos (por ex.: Token-Ring x Ethernet), o que representa um problema não trivial a ser resolvido;

- a terceira, ainda mais complexa, está relacionada com a hipótese de que as subredes a serem interconectadas, além de possuirem protocolos diferentes em cada camada, não sejam baseadas na mesma arquitetura (por exemplo, uma das subredes tem arquitetura IBM-SNA, a outra segue a arquitetura TCP/IP e uma terceira é OSI).

A solução normalmente adotada para os problemas acima consiste na definição de um equipamento especial de rede, cuja função é oferecer suporte para a interconexão, como ilustra a figura 2.4.1. Levanta-se então a questão de que características e funções estes equipamentos devem ter.

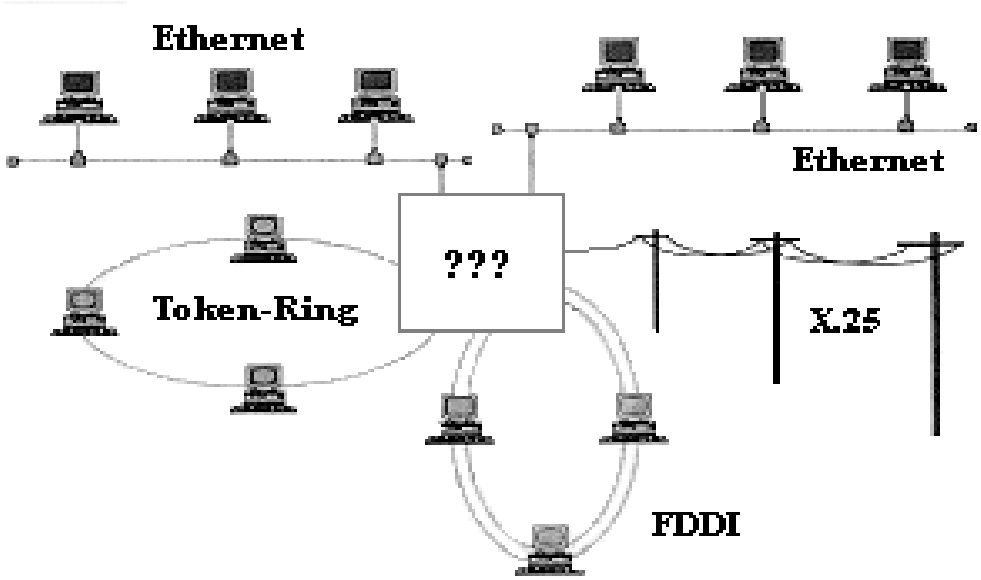


Figura 2.4.1 - O problema de interconexão de redes

O objetivo desta seção é discutir os principais aspectos associados à interconexão de subredes, abordando desde a forma como a interconexão é vista no modelo OSI, até a resolução das questões apresentadas acima. Apresentaremos também as funções principais dos equipamentos de interconexão de redes mais usados.

2.4.2. ASPECTOS DA CONECTIVIDADE

Retornando às questões levantadas acima, uma série de pontos técnicos devem ser resolvidos para possibilitar a interconexão de redes. Vamos discutir alguns destes pontos a seguir.

2.4.2.1. Endereçamento

A função essencial de uma rede de computadores é permitir a troca de informações entre os equipamentos a ela conectados. Dada a possibilidade de interligação de equipamentos heterogêneos, torna-se necessária a definição de uma política de representação dos nomes e endereços dos processos, que sejam reconhecidos em qualquer ponto da rede. Uma forma de

fazê-lo é permitir que os usuários possam ser identificados por *nomes* (ou cadeias de caracteres) em lugar de endereços numéricos, difíceis de memorizar. Isto significa que os nós da rede devem manter tabelas que efetuem o mapeamento entre os nomes dos processos usuários da rede e os seus endereços efetivos.

Entretanto, quando duas ou mais subredes são interconectadas, os padrões de endereçamento podem ser distintos e um meio de identificar (de maneira transparente) os processos situados nas diferentes subredes deve ser implementado. Além disso, no caso de adoção de nomes para os processos usuários, como os nomes locais podem se repetir em subredes diferentes, é necessário um esquema que os diferencie.

A política adotada para resolver esta questão é aquela do endereçamento hierárquico, como, por exemplo, a definida pelo CCITT (recomendação X.121), onde cada endereço apresenta três componentes: um código de país (3 dígitos), um código da rede (1 dígito) e um campo para o endereço dentro da rede (10 dígitos). Isto significa que cada país pode definir até 10 redes, cada rede tendo 10 dígitos para definir seus endereços, o formato dependendo de cada rede.

Um outro exemplo deste tipo de solução é o esquema de endereçamento oferecido pelo DNS (Domain Name System). Cada usuário é identificado de forma univoca na rede por um nome na forma “user@domínio1.domínio2”, que é convertido em endereço IP pelos servidores de nomes “domínio1” e “domínio2”, passando para algo na forma 200.24.120.5. Este é o esquema padrão adotado no TCP/IP (Internet).

2.4.2.2. Encaminhamento das mensagens

Este aspecto está diretamente ligado às funções de roteamento implementadas na camada de Rede. A questão colocada aqui está relacionada com a forma de encaminhamento de uma mensagem quando esta é trocada entre dois equipamentos situados em subredes diferentes.

Um aspecto a ser ressaltado aqui é o problema de roteamento, já abordado anteriormente (camada de Rede), sendo que alguns dos algoritmos de roteamento foram lá apresentados. Faz-se necessária aqui a disponibilização de equipamentos de interconexão de redes com a capacidade de realizar esta função entre subredes de tipos diferentes. Tal equipamento, como veremos mais a frente, recebe o nome de *router* (roteador).

2.4.2.3. A fragmentação de mensagens

Este é um outro problema importante, dado que diferentes subredes podem trabalhar com tamanhos distintos de mensagens (os tamanhos máximos). É óbvio que uma solução seria limitar o tamanho máximo das mensagens podendo transitar em todas as subredes como

sendo o menor tamanho máximo de todas as subredes, mas esta é uma solução pouco eficiente.

A solução mais interessante é a seguinte: quando uma mensagem deve transitar entre diferentes subredes, se ela ultrapassa o tamanho máximo da subrede pela qual ela vai transitar, esta é decomposta em várias mensagens (ou *fragmentos*), cujas dimensões serão adaptadas às limitações da subrede considerada. É evidente que, caso estes fragmentos sejam enviados a uma nova subrede que suporte o tamanho máximo da subrede de origem, estes poderão ser reagrupados para recompor a mensagem original.

Estes são alguns dos pontos a serem discutidos no momento em que um elemento deve ser projetado para a interconexão de duas subredes. Outros pontos não menos importantes são: o controle de erros, o tipo de serviço (orientado à conexão ou não), o nível da interconexão, o controle de fluxo, o controle de congestionamento, a segurança, a tarifação de serviços.

2.4.3. A INTERCONEXÃO SEGUNDO O MODELO OSI

Antes de estudarmos os elementos responsáveis da interconexão de redes, vamos analisar a abordagem do modelo OSI no que diz respeito a este problema particular. Segundo o que foi estabelecido no modelo OSI no que diz respeito à interconexão, a camada de Rede assume a resolução destes problemas, sendo para tal subdividida em três subcamadas, como mostra a figura 2.4.2: a *subcamada de acesso à subrede*, a *subcamada de adaptação da subrede* e a *subcamada de interconexão*.

A primeira subcamada, representada pelos níveis *3a* na figura 2.4.2, é responsável do gerenciamento do protocolo da subrede considerada, assumindo a recepção e o envio dos pacotes de controle e implementando as funções de base da camada de Rede.

A segunda subcamada, representada por *3b* em 2.4.2, é responsável de todas as adaptações entre serviços necessárias entre as subredes interconectadas. Caso uma das subredes não for compatível com o modelo OSI (caso nada raro), seus serviços terão nomes, funções e parâmetros diferentes, que precisam ser adaptados de forma a obter uma uniformização de sintaxe e semântica entre as subredes envolvidas.

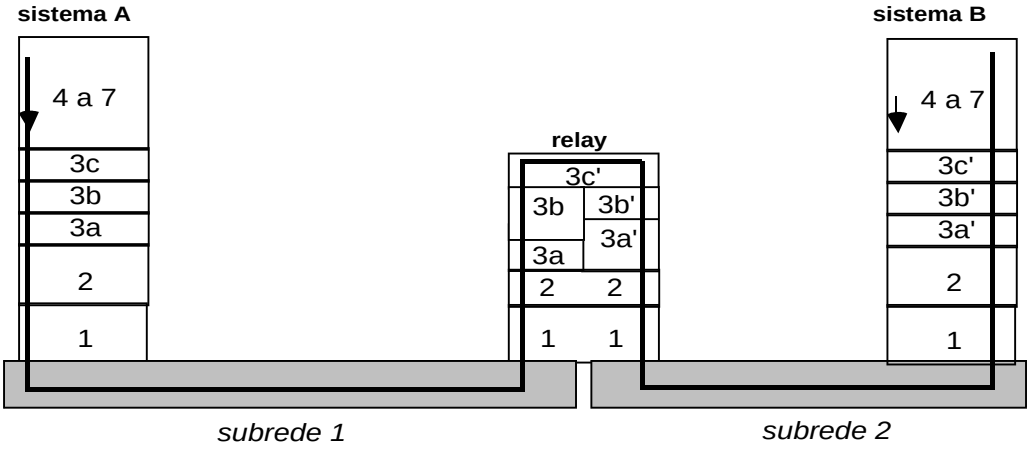


Figura 2.4.2 - Estrutura interna da camada de rede do ponto de vista da interconexão, segundo o modelo OSI.

Finalmente, a terceira subcamada, *3c* em 2.4.2, é a que implementa efetivamente a interconexão das subredes responsabilizando-se pelo roteamento entre as subredes.

As diferenças entre as subredes interconectadas podem ser de diferentes naturezas. No caso de subredes em conformidade com o modelo OSI, a interconexão é feita apenas através da transmissão, via subcamada de interconexão, dos pacotes às subcamadas de acesso à subrede da subrede considerada. Neste caso, a função da subcamada de adaptação é inexistente e esta camada fica então vazia.

Por outro lado, se uma rede não-OSI deve ser interconectada a uma baseada no modelo OSI, as adaptações deverão ser implementadas, isto sendo função daquela subcamada.

A tarefa de roteamento dentro de uma grande rede, composta de diversas subredes, é similar àquela a nível de uma única subrede, o que significa que as técnicas discutidas na seção referente à camada de rede podem ser utilizadas.

2.4.4. AS DIFERENTES POSSIBILIDADES DE INTERCONEXÃO

A figura 2.4.3 ilustra diferentes possibilidades de interconexão de redes (locais ou de longa distância). Em cada caso, é necessário introduzir um elemento intermediário ou *relay*, responsável das adaptações de protocolo que sejam necessárias, podendo ser uma *ponte* (ou *bridge*) ou *passarela* (*gateway*), dependendo do tipo de interconexão.

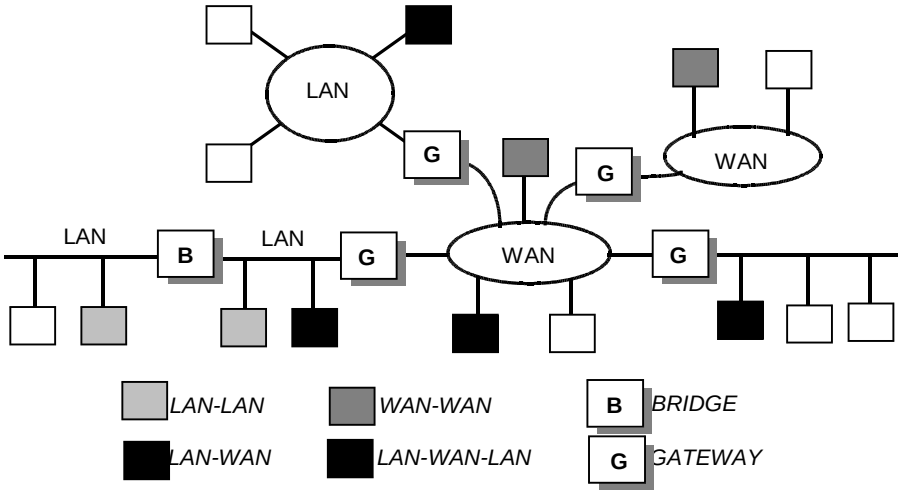


Figura 2.4.3 - Possibilidades de interconexão de redes.

Saindo um pouco do contexto do modelo OSI, no caso mais geral, a tarefa efetuada pelos relays pode ser implementada em qualquer camada, os quatro tipos de relays mais comuns sendo:

- os *repetidores* (*repeaters*), implementados no nível físico, que permitem unicamente amplificar e retransmitir os sinais elétricos representando os bits de dados entre dois segmentos de cabo;
- as *pontes* (*bridges*), implementadas no nível enlace, que efetuam o armazenamento e retransmissão dos quadros entre 2 redes locais; a retransmissão do quadro pode ser caracterizada por algumas modificações nos formatos dos quadros, se necessário;
- os *roteadores* (*routers*), implementados no nível rede, que retransmitem pacotes entre várias redes;
- as *passarelas* (*gateways*), implementadas ao nível aplicação, cuja tarefa é bem mais complexa que as dos elementos anteriores, utilizados para a interconexão de subredes incompatíveis até mesmo do ponto de vista da arquitetura (redes OSI x redes não-OSI).

Apresentaremos, a seguir, algumas características importantes destes elementos.

2.4.5. OS REPETIDORES (REPEATERS)

Os repetidores são usados para interligar subredes idênticas, produzindo basicamente o efeito de uma simples extensão. Eles atuam somente a nível físico, recebendo quadros de uma subrede, reforçando sinais elétricos e retransmitindo na outra subrede, conforme mostrado na figura 2.4.4. Sua implementação é usualmente feita somente em hardware.

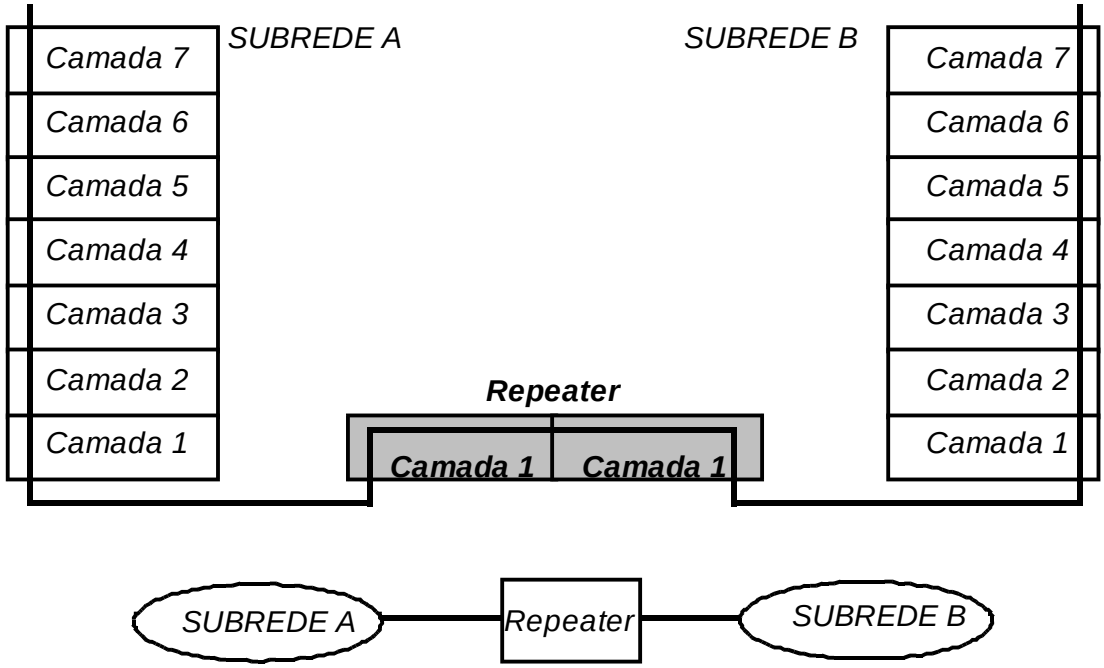


Figura 2.4.4 - Repetidor expandindo uma rede (nível OSI 1)

Um repetidor introduz sempre um pequeno retardo na entrega de uma mensagem, de modo que o número de repetidores que podem ser utilizados em uma rede é limitado.

Os repetidores tem uma função muito importante em redes com topologia em anel (ex.: Token-Ring), onde retransmitem mensagens entre segmentos de rede, de um nó do anel para o outro. Veremos o funcionamento deste tipo de rede mais a frente, na discussão sobre a norma IEEE 802.5.

Em redes com topologia em barramento, deve-se evitar caminhos fechados envolvendo repetidores, pois cada mensagem seria repetida infinitamente.

Em redes baseadas em contenção (ex.: CSMA/CD), o repetidor deve também detectar colisões em uma subrede e sinalizar sua ocorrência na outra (Figura 2.4.5).

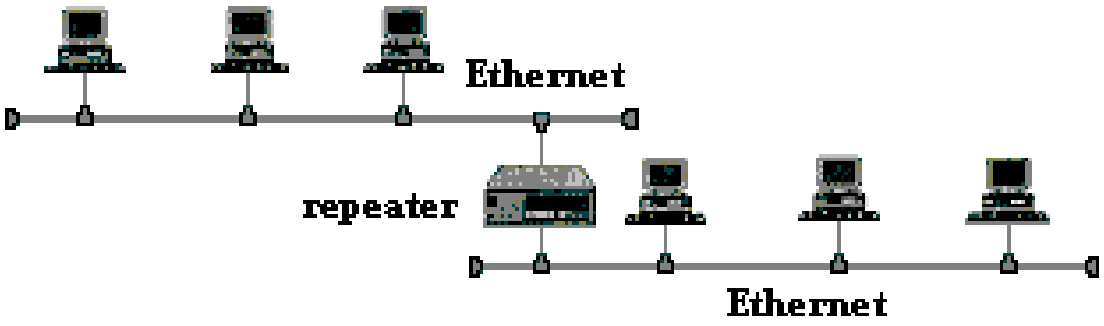


Figura 2.4.5 - Repetidor em redes CSMA/CD (Ethernet)

2.4.6. AS PONTES (BRIDGES)

Se duas subredes apresentam compatibilidade em relação à camada de enlace de dados, uma **ponte** pode ser utilizada para interconectá-las.

Uma ponte é um equipamento inteligente (baseado em microprocessador) conectado a duas subredes ou entre uma subrede e a backbone. Dado que as duas redes estão conectadas à ponte utilizam a mesma política de endereçamento na camada 2 do modelo OSI, ela examina os endereços de ambas as redes para definir que mensagens devem ser passadas de uma rede à outra. As pontes são bidirecionais por natureza, o que significa que elas são responsáveis do encaminhamento de todos os pacotes emitidos ao nível das duas redes. A figura 2.4.6 ilustra a forma de interconexão através de uma ponte.

A operação de uma ponte é baseada na manutenção de uma tabela contendo os endereços dos equipamentos compondo as redes à qual ela está associada.

Quando um pacote é recebido, esta examina o conteúdo do campo “endereço do destinatário” para verificar se ele está endereçado a um equipamento situado na mesma rede de origem ou não.

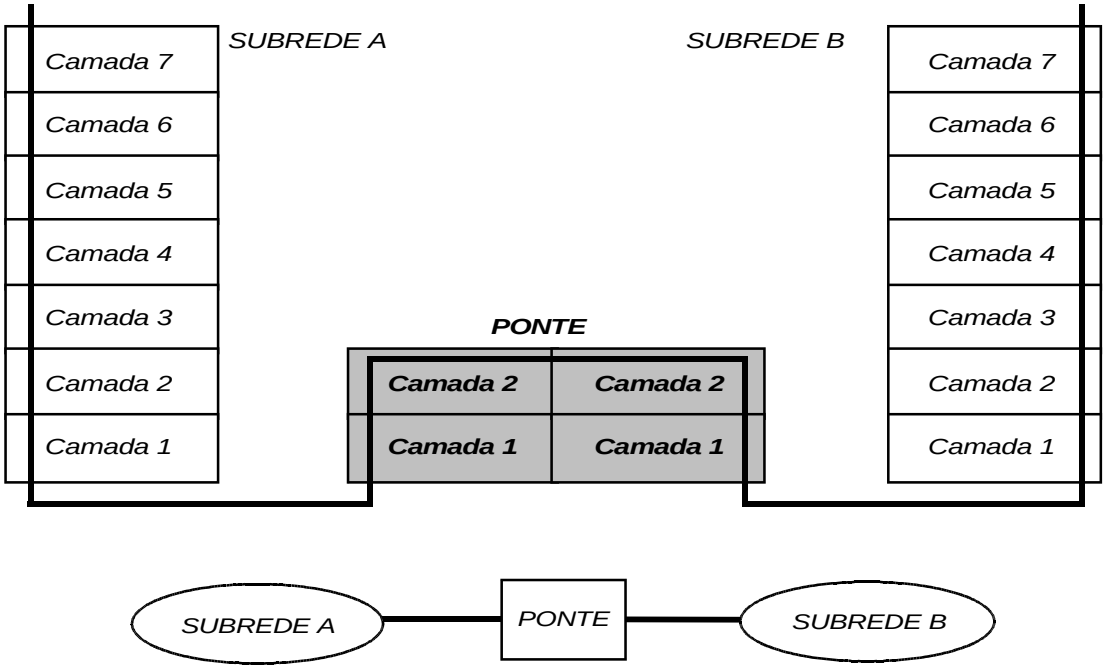


Figura 2.4.6 - Ponte interconectando duas sub-redes (nível OSI 2).

Em caso positivo, o pacote é encaminhado ao equipamento considerado. Caso contrário, este será despachado pela ponte para a outra sub-rede. Normalmente, as pontes implementam um algoritmo de "aprendizagem", utilizado para inicializar a tabela de endereçamento existente em cada uma delas.

Do ponto de vista do desempenho, as pontes são elementos de interconexão que apresentam um tempo de resposta relativamente curto, uma vez que, em grande parte de sua operação, os pacotes não sofrem nenhum processo de reformatação para serem despachados.

Um caso bastante comum encontrado nas empresas é a necessidade de interconexão de redes Ethernet com as redes do tipo Token-Ring (Anel com Ficha). Um primeiro problema a resolver, neste caso, é a grande diferença estrutural dos pacotes utilizados por cada um destes tipos de rede. Uma rede Ethernet utiliza pacotes cujo tamanho não deve ultrapassar os 1.500 bytes; uma rede Token Ring a 4 Mbit/s pode transmitir pacotes de até 4.000 bytes.

Uma ponte orientada à interconexão destes dois tipos de rede deve, então, oferecer a possibilidade de segmentação dos pacotes de grandes dimensões das redes Token-Ring em pacotes menores da Ethernet.

Dado que, no modelo OSI, a independência de redes é uma das características enfatizadas, é bem possível que as duas subredes utilizem suportes de transmissão (camada física) distintos. Um exemplo disto é a possibilidade do estabelecimento de interconexões por pontes nas arquiteturas MAP/TOP. Por utilizar a mesma camada de enlace (IEEE 802.2), as duas arquiteturas podem ser interconectadas, apesar de uma (MAP) utilizar o suporte de transmissão em banda larga e a outra (TOP) utilizar banda básica.

Na filosofia do modelo OSI, uma ponte pode interconectar duas redes quaisquer, desde que estas sejam totalmente compatíveis em suas camadas de enlace de dados e daí para cima (principalmente, no endereçamento). Ainda deste ponto de vista, dois usuários finais não precisam tomar conhecimento da existência de uma ponte na rede; esta é vista como um equipamento "observador" nas redes às quais ela está conectada. Sua função é monitorar todo pacote que circula em cada uma das redes à qual ela está associada.

2.4.7. OS ROTEADORES (ROUTERS)

Como foi dito acima, as pontes são equipamentos que permitem interconectar as subredes, duas a duas. Apesar de sua grande utilidade neste caso bastante comum, as pontes apresentam limitações que impedem outras maneiras também importantes de interconexão.

Os roteadores são elementos operando ao nível de Rede, que se utilizam do endereçamento definido a este nível para transferir e rotear as mensagens de uma rede a outra. Ao contrário das pontes que interligam as subredes duas a duas, os roteadores podem interligar duas ou mais subredes, sendo que a escolha de que linha utilizar é feita com base na execução de um algoritmo de roteamento, como já discutido anteriormente (camada de Rede).

A política de endereçamento implementada a nível da camada de Rede é bastante similar à codificação de números telefônicos numa rede de telefonia. Se alguém, por exemplo, quer fazer uma chamada telefônica de Florianópolis para Paris, ele deve discar inicialmente (após o código de discagem direta internacional) o código do país (no caso, 33 para a França), em seguida, o código da cidade (1, para Paris) e, finalmente, o número do telefone da pessoa

com quem ele vai querer dialogar. No modelo OSI, o esquema de endereçamento é definido de forma a cobrir os múltiplos formatos de endereçamento de rede (ver parte relativa à camada de Rede).

Uma das desvantagens das pontes que é o fato de que, todo pacote transitando ao longo das subredes é recebido por cada uma das estações conectadas a estas subredes, implicando, em condições normais, num aumento considerável de tráfego. Ainda, a possibilidade de um congestionamento não está muito distante uma vez que uma interface de rede em pane poderá despejar uma grande quantidade de pacotes incompatíveis na rede.

Ao contrário das pontes, um roteador não tem necessidade de analisar todos os pacotes circulando na rede. Isto significa que, no caso dos roteadores, o problema descrito acima pode ser evitado pois eles seriam capazes de bloquear aqueles pacotes que não obedecessem a um determinado perfil.

É um equipamento bastante poderoso, dado que ele pode interconectar um número relativamente grande de redes, de uma forma transparente ao usuário do serviço. Em aplicações onde haja necessidade de interligação de mais de duas subredes, um roteador deverá certamente ser o elemento escolhido para realizar a conexão (em lugar de uma ponte).

Em aplicações industriais, por exemplo, um bom número de empresas se caracteriza por possuir suas usinas, fornecedores, depósitos, lojas de venda, etc, em locais geográficamente dispersos. A fim de trocar informações entre estes setores, a conexão das diversas redes locais a uma rede de longa distância pode ser viabilizada através de um roteador.

Um papel importante desempenhado pelos roteadores está na interconexão de redes heterogêneas. Quando um pacote pertencente a uma subrede implementando um protocolo X deve ser encaminhado a uma subrede implementando um protocolo Y, o roteador deverá realizar as conversões de formato necessárias para que o pacote seja encaminhado respeitando os requisitos impostos pelo protocolo Y. A interconexão entre subredes através de um roteador é ilustrada nas figuras 2.4.7.

2.4.8. AS PASSARELAS (GATEWAYS)

Os gateways são os elementos de interconexão de concepção mais complexa. A sua importância no que diz respeito às necessidades de interconexão das redes é o fato de que nem todas as redes de comunicação implantadas e em funcionamento atualmente foram construídas com base no modelo OSI, muitas soluções "proprietárias" e "padrões de fato" sendo adotados na forma de redes locais.

Isto significa que está longe do incomum a necessidade de interligação de redes baseadas no modelo OSI com redes não-OSI. Isto requer, então, a construção de um equipamento de interconexão que seja capaz de compatibilizar as diferenças estruturais e de protocolo existentes entre as duas redes. Este equipamento é o gateway.

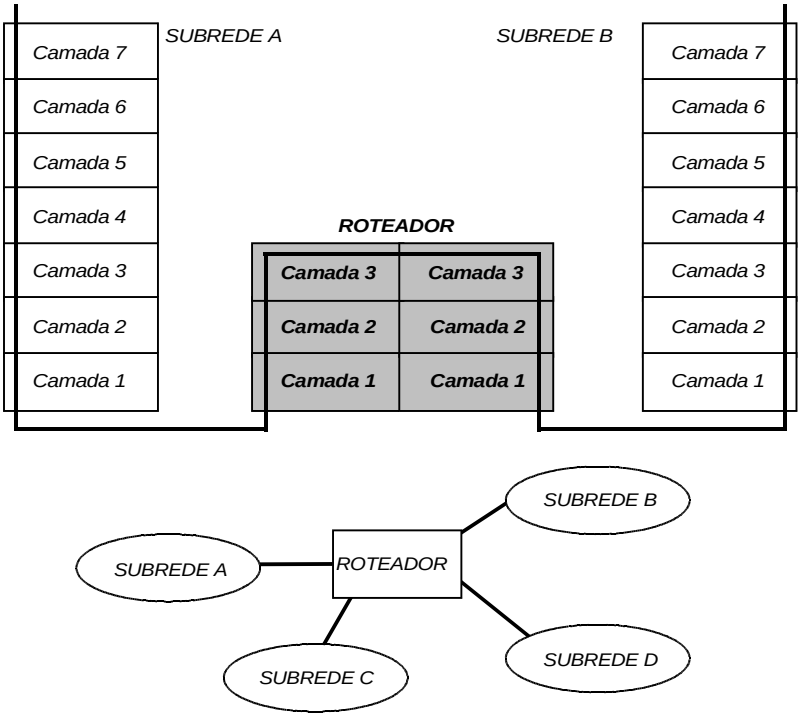


Figura 2.4.7 - Roteador interligando subredes (nível OSI 3).

Os gateways são elementos que devem possuir dois "stacks" de protocolos, um sendo baseado na arquitetura a 7 camadas do modelo OSI e o outro, baseado na arquitetura proprietária considerada (figura 2.4.8).

Normalmente, os gateways são construídos com uma orientação a uma dada aplicação, como, por exemplo, a interconexão entre uma rede proprietária e uma rede MAP. Um exemplo disto é a interconexão de uma rede com arquitetura SNA (proprietária da IBM) na qual um mainframe IBM está interconectado com uma rede MAP. O gateway a ser construído para realizar esta conexão deveria possuir as 7 camadas (OSI-like) definida pela arquitetura MAP, um programa de aplicação para realizar a transferência de arquivos, e um conjunto de protocolos necessários para a comunicação dentro da rede SNA.

2.4.9. CONCENTRADORES

Apesar de os concentradores não serem propriamente equipamentos de interconexão entre subredes como os repetidores, pontes, roteadores e passarelas, abordaremos aqui brevemente estes equipamentos, pois são também importantes elementos utilizados no gerenciamento e operação de redes. Os equipamentos chamados concentradores representam um retorno à topologia em estrela do ponto de vista físico, mas mantendo a topologia lógica requerida pelas placas de rede em uso (barramento, anel, etc.).

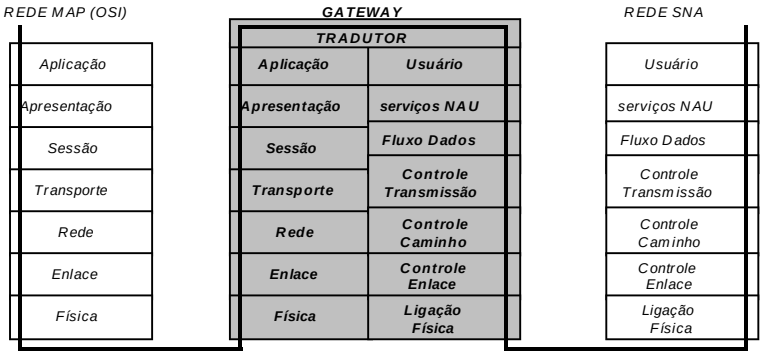


Figura 2.4.8 - Interconexão baseada em gateway.

A intenção é facilitar o gerenciamento e manutenção do sistema de comunicação, uma vez que os problemas que eventualmente ocorrerem na rede muito provavelmente estarão no concentrador e não em um ponto qualquer da mesma.

Existem dois tipos básicos de concentradores:

- concentradores passivos (ou ***HUBs***): usualmente não tem inteligência local, atuando como emuladores de barramentos (figura 2.4.9). Na maioria dos casos, operam com fios tipo par trançado ou fibra ótica. Cada conector do HUB para um nó de rede esta isolado galvanicamente, de modo que a abertura de uma das linhas não afeta as demais.
- concentradores ativos ou Comutadores (***Switchers***): tem inteligência local e podem chavear mensagens simultâneas para destinos diferentes em alta velocidade (figura 2.4.10). Permitem uma melhora significativa de desempenho da rede, uma vez que subdividem o sistema em várias sub-redes que podem operar de forma independente.

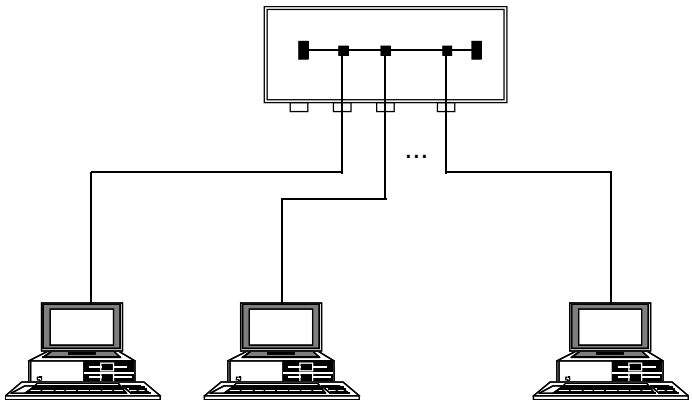


Figura 2.4.9 - HUB

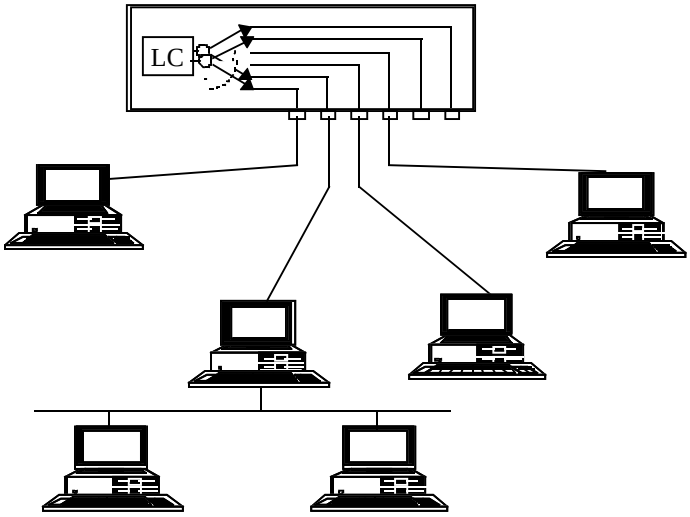


Figura 2.4.10 - Switcher

3. AS REDES LOCAIS INDUSTRIAIS

3.1. INTRODUÇÃO

A fabricação de produtos ou o fornecimento de um serviço da parte de uma empresa põe em jogo uma série de atividades e etapas, dedicadas à manutenção e ao aprimoramento do produto ou do serviço. A implementação destas etapas através de processos com maior ou menor grau de automatização fica a critério da empresa.

A tendência de informatização crescente das empresas e outras organizações, por um lado, permite acelerar cada processo fazendo parte das atividades de fabricação de um produto ou do oferecimento de um serviço e, por outro lado, cria uma nova necessidade no que diz respeito ao modo como as informações serão trocadas.

As redes locais industriais permitem levar em conta as particularidades de um processo de fabricação do ponto de vista das necessidades de comunicação, tais como o compartilhamento de recursos, evolutividade, gerenciamento da heterogeneidade e os diversos tipos de diálogo podendo ocorrer no ambiente industrial. Ainda, a nível de um processo de fabricação, certos requisitos tornam-se fundamentais, envolvendo principalmente os fatores econômicos que o cercam. A garantia de um tempo de resposta médio ou máximo, o débito de informação, a robustez (confiabilidade dos equipamentos e da informação), a flexibilidade (evolutividade e heterogeneidade dos equipamentos) são alguns exemplos destes requisitos.

Nos últimos anos, um esforço considerável tem sido realizado no sentido de definir arquiteturas de comunicação que respondam às características e aos requisitos mencionados acima. O exemplo mais evidente de resultados deste esforço é a arquitetura MAP (Manufacturing Automation Protocol), definida por iniciativa da General Motors e baseada no modelo OSI.

Por outro lado, considerando que as necessidades de comunicação em cada classe de atividades de uma empresa pode assumir diferente importância, outras propostas de arquiteturas podem ser consideradas, como por exemplo as arquiteturas da classe barramento de campo (fieldbus), mais dedicadas ao chão de fábrica, como o nome sugere.

O objetivo desta parte do documento, além de apresentar algumas propostas de arquiteturas de redes locais industriais, é discutir os principais aspectos relacionados com esta classe particular de redes.

Dados os requisitos e características apresentados anteriormente, um ponto que deve ser levantado como conseqüência disto é a necessidade de descentralização das funções de comunicação a serem implementadas. Este aspecto pode ter importância fundamental na escolha das soluções de comunicação a serem adotadas a nível de uma indústria. As redes do tipo ponto-a-ponto, por exemplo, são um exemplo típico de centralização das funções de comunicação, uma vez que os equipamentos compondo os nós da rede fazem papel de comutadores das mensagens transmitidas entre dois nós que não estejam ligados diretamente. Soluções do tipo rede de difusão são largamente adotadas levando em conta as possibilidades de descentralização do controle da comunicação.

É preciso assumir a realidade de que não existe uma rede única que poderia corresponder às necessidades de todas as classes ou níveis de atividade existentes em uma fábrica, a solução sendo, de fato, a adoção de várias redes interconectadas, cada rede servindo de suporte à comunicação no contexto de uma ou diversas atividades.

3.1.1. As redes e os níveis hierárquicos de integração fabril

Já há algum tempo vem se verificando uma tendência para a descentralização da inteligência e da capacidade decisória dos componentes de sistemas de automação industrial. Estes sistemas são decompostos em diferentes níveis hierárquicos de automação, cujos elementos inteligentes são interligados entre si através de redes industriais, conforme mostrado na figura 1.1. A tendência desta estruturação hierárquica é se aproximar cada vez mais do processo, de forma a obter-se cada vez mais subsistemas independentes e dotados de uma inteligência local, sem no entanto perder as vantagens de uma supervisão e condução central do sistema como um todo. Esta descentralização traz consigo uma série de vantagens técnicas, tais como a diminuição da sobrecarga de processamento da unidade central, entre outras.

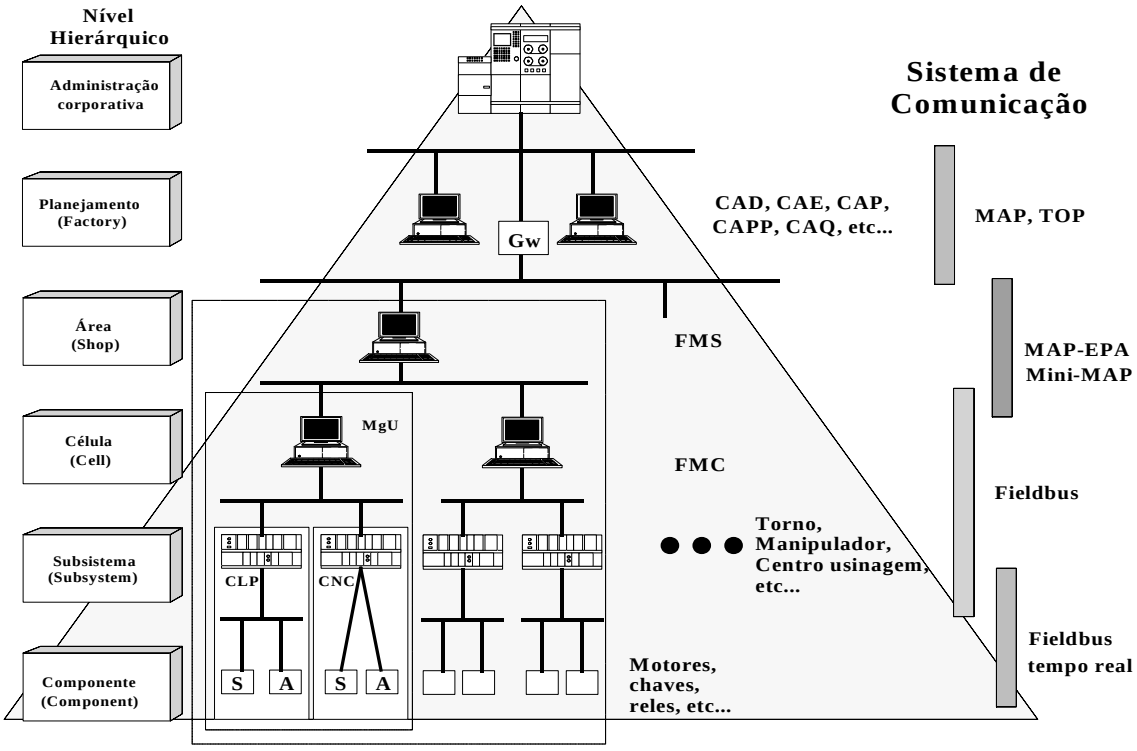


Figura 1.1 - As redes de comunicação e o modelo CIM

Cada nível da hierarquia fabril é representado por um conjunto de ações e processamentos que possuem requisitos de comunicação diferentes. A característica predominante nos níveis hierárquicos inferiores é a transferência de mensagens curtas com alta freqüência, entre um número elevado de estações. Nos níveis hierárquicos superiores há a predominância de transferência de mensagens longas entre um número menor de estações e a uma freqüência consideravelmente mais baixa (figura 1.2).

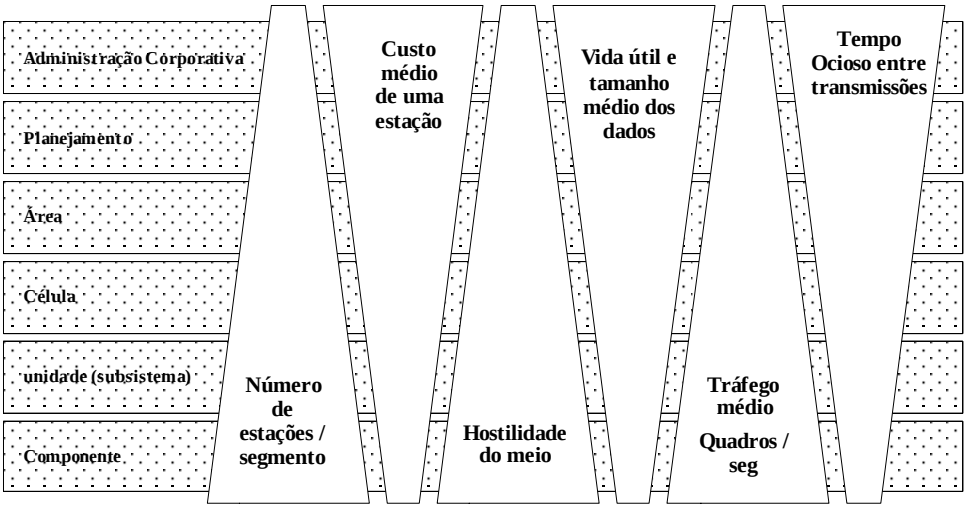


Figura 1.2 - Características da comunicação em CIM

Deste modo, não existe um sistema de comunicação único capaz de atender a todas as aplicações existentes na organização industrial, mas sim uma série de sub-redes locais adequada aos requisitos de comunicação de cada nível. As sub-redes serão conectadas à linha tronco ("Backbone") através de "Gateways", "Bridges" e "Routers" de modo que todas as estações possam ser acessadas, formando um sistema de comunicação coêso que atenda toda a fábrica, conforme os preceitos da filosofia CIM (Computer Integrated Manufacturing).

3.2. REDES LOCAIS INDUSTRIAIS

3.2.1. Motivação

A maioria das redes de comunicação existentes no mercado procuraram atender a demanda existente na automação de escritórios. A grande maioria destas redes são baseadas no protocolo CSMA/CD (Carrier Sense Multiple Access with Collision Detection), com o qual se iniciaram os desenvolvimentos de redes locais (LANs, Local Area Networks) e que será estudado mais adiante.

A comunicação de dados em ambiente industrial apresenta, no entanto, características e necessidades que tornam a maioria das redes para automação de escritório inadequadas. Algumas destas características são:

- ambiente hostil para operação dos equipamentos (perturbações eletromagnéticas, elevadas temperaturas, sujeira, etc.);
- a troca de informações se dá, na maioria das vezes, entre equipamentos e não entre um operador humano e o equipamento;
- os tempos de resposta e a segurança dos dados são críticos em diversas situações;
- uma grande quantidade de equipamentos pode estar conectada na rede, o que torna a questão de custos muito importante.

3.2.2. Características básicas das redes industriais

Os aspectos tecnológicos básicos discutidos nas seções que seguem diferenciam as redes locais para aplicações industriais das demais redes.

3.2.2.1. COMPORTAMENTO TEMPORAL

3.2.2.1.1 - Introdução à comunicação tempo real

Como foi dito na seção anterior, as redes de difusão apresentam aspectos interessantes que as tornam uma solução bastante adequada aos requisitos de comunicação industrial. Um problema importante na utilização das redes de difusão é o método de acesso ao meio (que é compartilhado) pois, uma vez que vários equipamentos deverão trocar informações num dado instante, a decisão de quem vai ter o direito de uso do meio para o envio de uma mensagem não é uma tarefa evidente, como será visto nesta seção. Os protocolos de acesso ao meio tem papel fundamental no tempo de entrega de uma mensagem via rede. Como veremos a seguir, este tempo é importante para aplicações com características de tempo real.

Aplicações Industriais freqüentemente requerem sistemas de controle e supervisão com características de Tempo-Real. Um Sistema Tempo-Real é um sistema computacional para o qual é requerida uma reação a estímulos (físicos ou lógicos) oriundos do ambiente dentro de intervalos de tempo impostos pelo próprio ambiente (figura 2.1). A correção não depende somente dos resultados lógicos obtidos, mas também do instante no qual são produzidos.

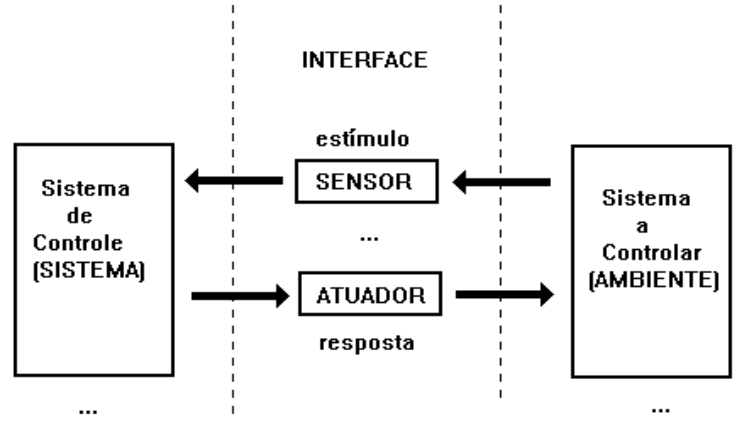


Figura 1.1 - Interação de um Sistema de Controle TR e seu ambiente

Figura 2.1 - Sistema Tempo Real e seu Ambiente

A arquitetura de sistemas computacionais utilizados para controle e supervisão de processos industriais em tempo real tem apresentado nos últimos anos uma clara tendência para a distribuição das funções de controle, como ilustrado na figura 2.2.

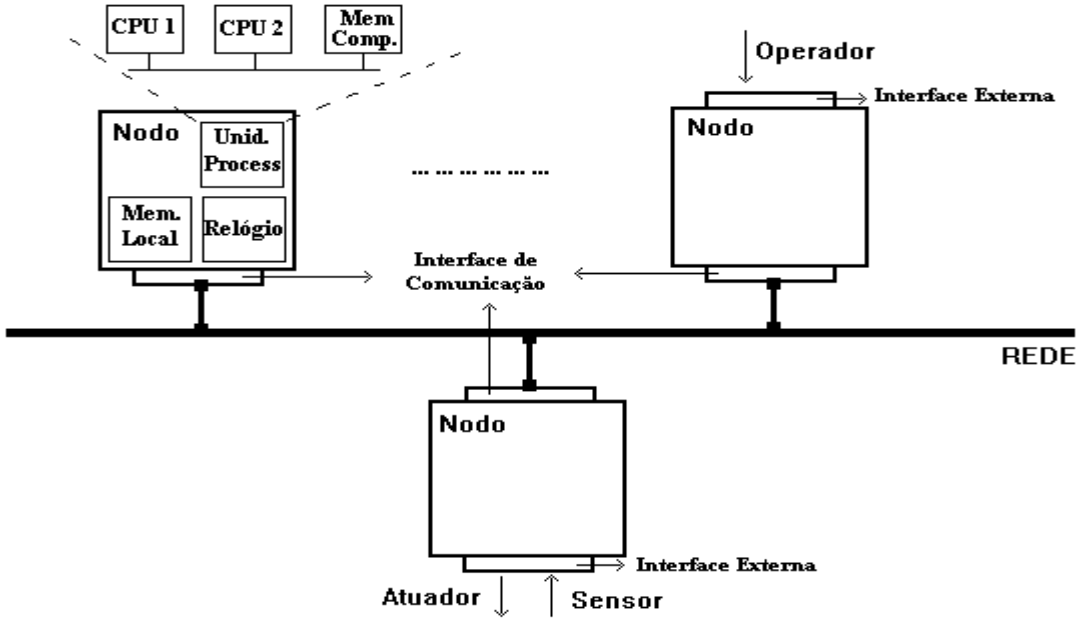


Figura 1.2 - Arquitetura de um STR

Figura 2.2 - Arquitetura distribuída de um sistema tempo real

Em aplicações tempo real, é importante poder determinar o comportamento temporal do sistema de comunicação envolvido. As ***mensagens*** em sistemas TR podem ter restrições temporais associadas e podem ser classificadas em:

- **Periódicas**: tem que ser enviadas em intervalos conhecidos e fixos de tempo. Ex.: mensagens ligadas a malhas de controle.
- **Esporádicas**: mensagens sem período fixo, mas que tem intervalo de tempo mínimo entre duas emissões consecutivas. Ex.: pedidos de status, pedidos de emissão de relatórios.
- **Aperiódicas**: tem que ser enviadas a qualquer momento, sem período nem previsão. Ex.: alarmes em caso de falhas.

Do ponto de vista da programação distribuída, o meio de transmissão (o barramento) constitui um recurso compartilhado entre as estações a ele conectadas. Os métodos de definição de direito de acesso utilizados nas redes locais são os denominados protocolos de acesso ao meio. O problema de comunicação em tempo real tem forte ligação com o tipo de protocolo de acesso ao meio adotado. A figura 2.3 ilustra a problemática aqui discutida. Suponha que desejamos transmitir 5 mensagens diferentes originadas de 5 estações na rede. Cada mensagem tem um tempo limite de entrega associado a ela, aqui denominado ***deadline***. Cada estação tem seu endereço na rede, também indicado na figura.

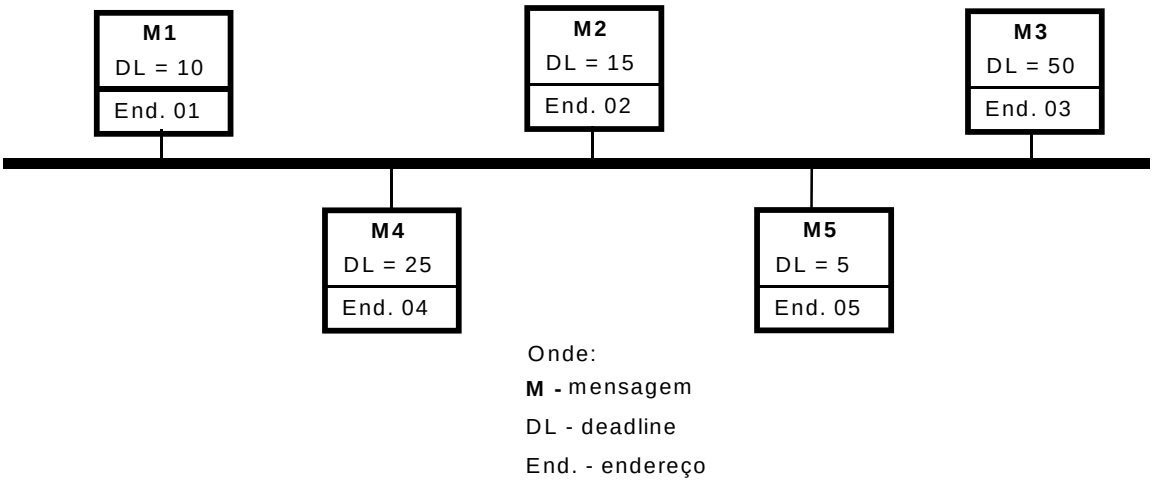


Figura 2.3 - A problemática da comunicação tempo real

As mensagens pendentes em cada estação devem ser entregues a seu destino antes de um prazo limite (deadline) associado. Assim, o problema de comunicação tempo real fica sendo o seguinte:

- como organizar as filas locais de mensagens pendentes, de forma que a mais prioritária seja colocada na cabeça da fila ?
- como definir concessão do direito de acesso ao meio de forma a garantir que a mensagem mais prioritária do conjunto de estações seja enviada primeiro e todas as mensagens sejam entregues antes de seu deadline ?

O Protocolo MAC utilizado precisa garantir rápido acesso ao barramento para mensagens esporádicas de alta prioridade. Ele deve também atender mensagens periódicas com a maior eficiência possível, respeitando seus deadlines.

Isto implica em que o sistema deve ter **comportamento determinista** (isto é, seu tempo de reação deve ser conhecido) e, idealmente, permitir **escalonamento** ótimo global de mensagens. Para tal, o LLC (Controle Lógico de Enlace) deve escalonar mensagens locais pendentes por deadline ou prioridade associada.

Para garantir um melhor desempenho temporal do sistema, é usual utilizar-se em sistemas tempo real uma arquitetura se software com apenas três camadas, com a camada de enlace subdividida em ***Controle de acesso ao meio*** (MAC) e ***Controle lógico de enlace*** (LLC), conforme mostrado na figura 2.4.

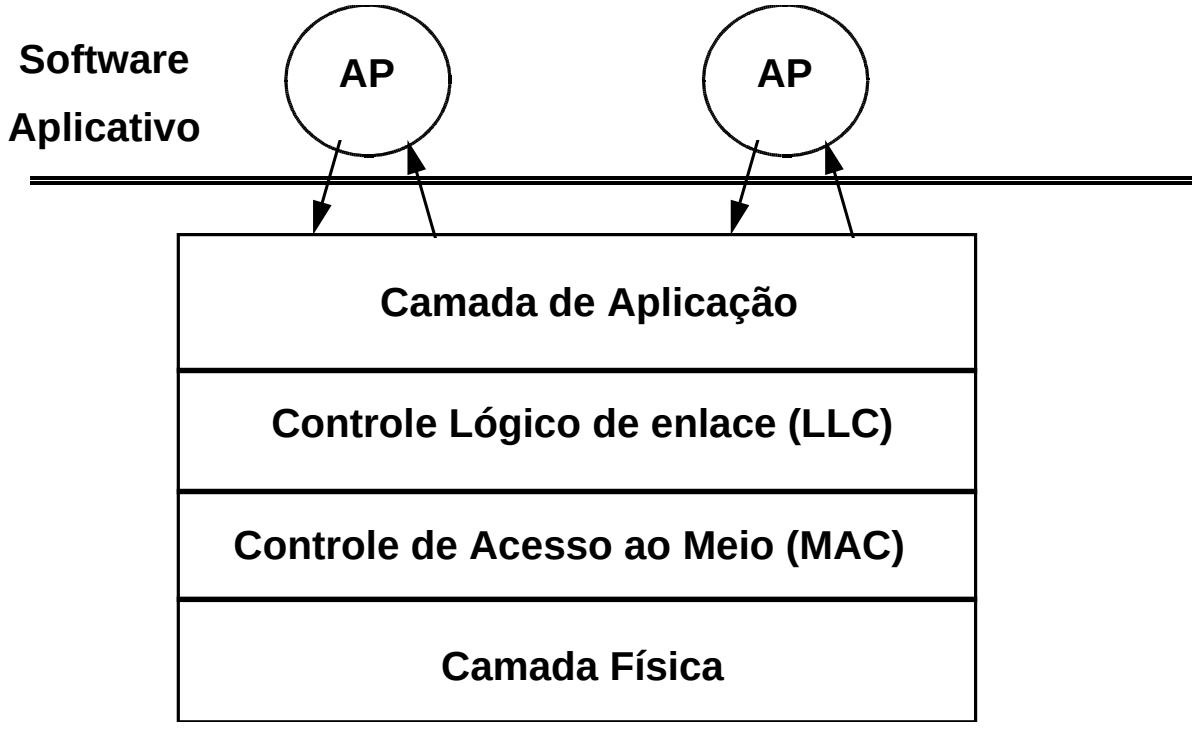


Figura 2.4 - Arquitetura para sistemas tempo real

3.2.2.1.2 - Classificação dos Protocolos de Controle de Acesso ao meio

Os protocolos de acesso ao meio podem ser classificados de maneira geral em 5 categorias:

- **Alocação fixa:** estes protocolos alocam o meio às estações por determinados intervalos de tempo (chamados de bandas), independentemente de haver ou não necessidade de acesso (ex.: TDMA = Time Division Multiple Access);
- **Alocação aleatória:** permitem acesso aleatório das estações ao meio (ex.: CSMA = Carrier Sense Multiple Access). Em caso de envio simultâneo por mais de uma estação, ocorre uma colisão e as estações envolvidas tem que transmitir suas mensagens após a resolução do conflito resultante (também chamados protocolos de contenção);
- **Alocação controlada:** cada estação tem direito de acesso apenas quando de posse de uma permissão, que é entregue às estações segundo alguma seqüência predefinida (ex.: Token-Passing, Master-Slaves);
- **Alocação por reserva:** para poder usar o meio, as estações tem que reservar banda com antecedência, enviando pedidos a uma estação controladora durante um intervalo de tempo pré-destinado e este fim (ex.: CRMA = Cyclic Reservation Multiple Access);
- **Híbridos:** consistem de 2 ou mais das categorias anteriores.

Com relação ao seu comportamento temporal, podemos organizar os protocolos de acesso ao meio em duas principais classes: os protocolos ***determinísticos*** e os ***não determinísticos***.

Os protocolos ditos determinísticos são caracterizados pela possibilidade de definir um tempo limite para a entrega de uma dada mensagem (mesmo que somente em pior caso), enquanto os protocolos não deterministicos não oferecem tal possibilidade.

Muitos dos atuais protocolos de acesso determinísticos são caracterizados pela concessão do direito ao acesso independentemente das necessidades de transmissão de cada nó da estação (alocação fixa). Dentre os protocolos conhecidos desta classe, podemos destacar o protocolo TDMA (Time Division Multiple Access), onde o acesso é dado a cada estação considerando faixas de tempo bem definidas. Este método apresenta um baixo desempenho, uma vez que muito tempo pode ser perdido no caso de estações que não tenham mensagens a transmitir.

Outros exemplos de protocolos de acesso determinísticos são aqueles baseados na passagem de ficha (*token passing*), onde uma ficha correspondendo ao direito de transmissão é passada de estação a estação da rede. Ao receber a ficha, uma estação que não tenha mensagens a transmitir repassa a ficha à estação seguinte na lista de estações compondo a rede (alocação controlada). Vamos analisar alguns destes protocolos mais adiante nesta seção.

Os protocolos de acesso não determinísticos, são freqüentemente caracterizados pela competição entre estações pelo direito de acessar o meio de transmissão. Um exemplo desta classe é o protocolo CSMA/CD.

Analisaremos a seguir alguns dos protocolos de acesso ao meio mais conhecidos, procurando destacar o aspecto de comportamento temporal.

3.2.2.1.3 - Os protocolos MAC não determinísticos

***- CSMA persistente e não persistente*:**

Estes protocolos, pertencentes à classe de protocolos ditos de detecção de portadora (carrier sense), baseiam-se no conceito de escuta do meio de transmissão para a seleção do direito de acesso a este.

Um primeiro exemplo deste protocolo é o *CSMA 1-persistente (Carrier Sense Multiple Access* ou *Acesso Múltiplo por Detecção de Portadora*). Neste protocolo, quando uma estação está pronta a enviar um quadro de dados, ela escuta o que está ocorrendo no suporte de transmissão. No caso em que o canal já está sendo ocupado por alguma transmissão, a estação aguarda na escuta até que o meio esteja livre para a sua emissão (daí o nome "*persistente"*); quando isto ocorre, ela pode então emitir um quadro. O método é chamado "1"-persistente porque, quando a linha esta livre, a estação enviará os dados com 100% de probabilidade. Após a transmissão dos dados, e estação emissora pode ser programada para esperar uma resposta (chamada *quadro de reconhecimento*) da estação receptora, indicando a correta recepção dos dados.

Se mais de uma estação estava a espera de uma oportunidade de enviar dados no mesmo momento, pode ocorrer que várias detectem o meio como estando livre ao mesmo tempo. Neste caso, todas irão enviar seus dados simultaneamente, de forma que o sinal no barramento será uma "mistura" ininteligível das várias mensagens. Esta condição recebe o nome de "*Colisão*". Na ocorrência de uma colisão, se estiver sendo usado um serviço confiável na subcamada de controle lógico de enlace (LLC com conexão ou com reconhecimento), a estação receptora não envia o quadro de reconhecimento esperado e a estação emissora tenta a emissão novamente após um determinado tempo. Se estiver sendo usado um serviço não confiável, o quadro é perdido.

O protocolo CSMA 1-persistente é altamente influenciado pelo tempo de propagação dos quadros no suporte de transmissão. Isto é ilustrado pelo exemplo de duas estações A e B querendo emitir um quadro. Vamos supor que A detecta o meio livre e emite um quadro; em seguida, B vai escutar o meio para ver o seu estado; se o atraso de propagação do quadro emitido por A é tal que o sinal ainda não pode ser detectado a nível da estação B, então esta vai considerar o meio livre e emitir o seu quadro, gerando naturalmente uma colisão. Isto

significa que, quanto maior o tempo de propagação no suporte de comunicação, pior o desempenho do protocolo devido à ocorrência de colisões. O tempo de propagação depende principalmente da taxa de transmissão (bits/s) e do comprimento do cabo.

Na verdade, embora as probabilidades não sejam muito grandes, as colisões podem ocorrer mesmo se o tempo de propagação é considerado nulo. Vamos supor agora que as estações A e B tem quadros a transmitir, mas que uma terceira estação C está utilizando o meio. Neste caso, as duas estações vão aguardar a liberação do meio e, quando este estiver liberado, ambas vão emitir seus quadros, caracterizando a colisão.

Para reduzir a probabilidade de ocorrência de colisões, foram criadas variantes deste protocolo, como por exemplo o *CSMA não persistente*. Segundo este protocolo, as estações comportam-se de maneira menos "afoita" para o envio de mensagens. Assim, uma estação que deseje emitir um quadro vai escutar o suporte de transmissão para verificar se este está disponível. Em caso positivo, o quadro será transmitido. Se o meio estiver ocupado, ao invés de ficar escutando persistentemente à espera da liberação do canal, a estação vai esperar um período de tempo aleatório e, após a expiração deste, vai escutar o canal novamente para verificar a sua liberação. Este protocolo permite reduzir as probabilidades de ocorrência de colisões, uma vez que os tempos de espera aleatórios só por muita coincidência serão iguais em todas as estações interessadas em transmitir. Como desvantagem temos o fato de que ele introduz um maior atraso de emissão a nível das estações do que o protocolo persistente, decorrente dos tempos aleatórios de espera pela liberação do meio.

O *CSMA p-persistente* é mais um exemplo de protocolo de acesso baseado em contenção, que procura ser um compromisso entre as duas propostas anteriores, funcionando da seguinte maneira: quando uma estação tem um quadro a enviar, ela escuta o canal para verificar a disponibilidade deste. Se o canal está disponível, a estação emite um quadro com probabilidade igual a *p*. A probabilidade de que esta não transmita o quadro e opte por aguardar por um intervalo de tempo fixo é igual a $q = 1 - p$. Se a escolha recair em não transmitir o quadro, após a passagem do intervalo de tempo especificado o canal é novamente testado e, se estiver disponível, as probabilidades de envio ou de espera continuam as mesmas. O processo continua, então, até que o quadro seja finalmente transmitido ou que outra estação tenha tomado posse do canal. Observe que o protocolo CSMA 1-persistente é um caso particular do CSMA p-persistente, onde $p = 1$.

O algoritmo não persistente é portanto eficiente para evitar colisões, mas implica em desperdício de tempo de transmissão. Já o algoritmo persistente não ocasiona este desperdício de tempo, mas apresenta elevada probabilidade de colisão. O algoritmo p-persistente é uma solução de compromisso entre os outros dois.

Todos os protocolos CSMA tem o inconveniente de que as colisões não são detectadas e a ocorrência de um problema de comunicação só é percebida, se optarmos por serviços de enlace confiáveis na subcamada acima (LLC), quando o tempo limite para

recepção do quadro de reconhecimento expira sem que este quadro seja recebido. Para resolver este problema, foi concebido o protocolo CSMA/CD, que veremos a seguir.

***- O protocolo CSMA/CD (CSMA with Collision Detection)*:**

Os protocolos descritos até aqui, embora apresentando aspectos interessantes, podem ser melhorados considerando-se que cada estação poderia detectar, antes da emissão, o estado de conflito com outras estações da rede, evitando assim a emissão do quadro considerado.

O protocolo CSMA/CD (*Carrier Sense Multiple Access with Collision Detection*) é um protocolo baseado neste princípio e muito utilizado nas redes locais (foi proposto originalmente pelos criadores da rede Ethernet). Neste protocolo, quando mais de uma estação esta pronta para emitir uma mensagem com o meio livre, estas emitem o quadro, o que vai gerar uma colisão. A primeira estação que detectar a colisão interrompe imediatamente a sua transmissão, reiniciando o processo todo após a expiração de um período de tempo aleatório, de forma a tornar improvável a ocorrência de uma nova colisão (figura 2.5). Para detectar a colisão, a estação emissora deve escutar aquilo que ela mesma colocou no meio (ao menos a primeira palavra de código enviada deve ser escutada pela própria estação emissora). O sinal enviado requer certo tempo para se propagar no meio, definido como um *time slot*. Os tempos de espera aleatórios usados pelas estações para tentar uma nova transmissão após uma colisão são sempre múltiplos de um time slot.

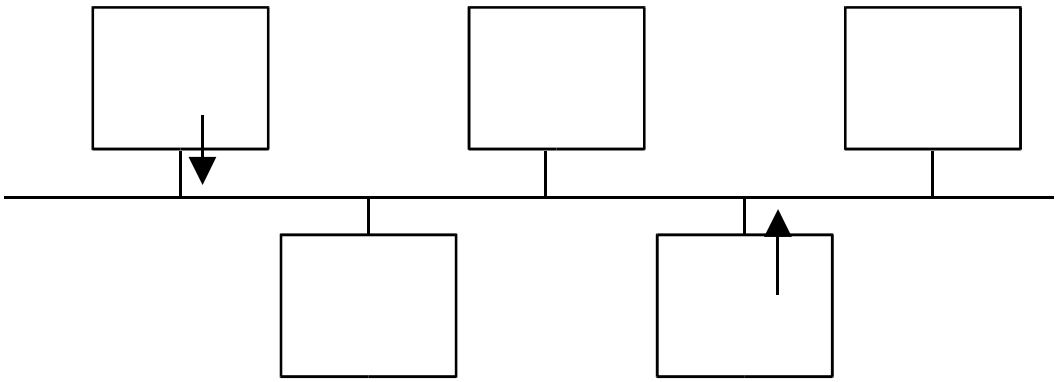


Figura 2.5 - Colisão no método de acesso CSMA/CD

O protocolo CSMA/CD segue um algoritmo de determinação do tempo limite de espera para retransmissão conhecido como *binary exponential backoff*, ou espera aleatória exponencial binária (alguns autores preferem o nome *truncated exponential backoff*). Neste algoritmo, a estação, ao detectar uma colisão, espera por um tempo gerado aleatoriamente que vai de zero até um limite superior. O limite superior inicia em 1 time slot e é pouco mais que dobrado a cada nova colisão. Se após algumas retransmissões as colisões ainda persistem, a

transmissão é finalmente abortada. O fluxograma da figura 2.6 mostra o principio deste algoritmo.

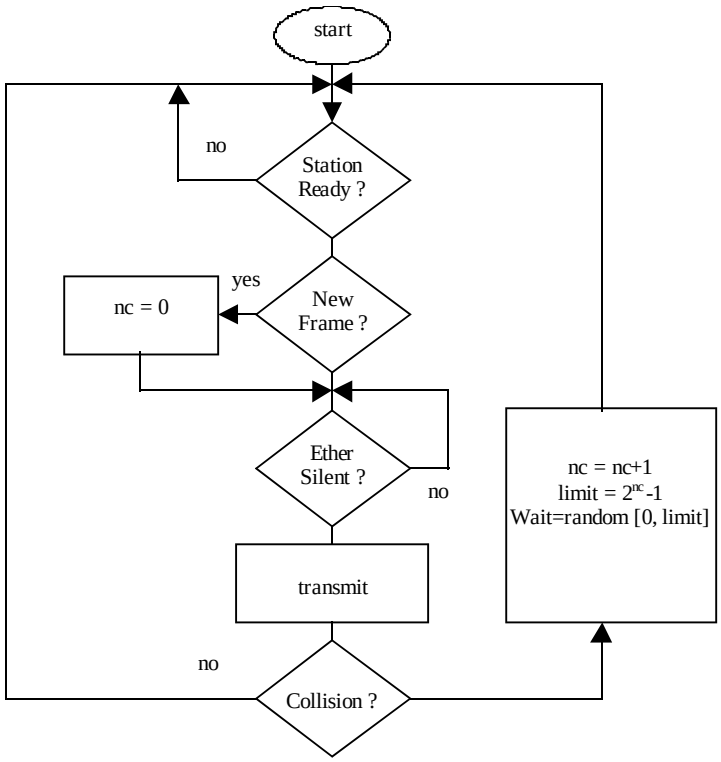


Figura 2.6 - Randomização de tempo no CSMA/CD (Binary Exponential Backoff)

Para melhor entender o mecanismo deste protocolo, vamos analisar o caso em que duas estações iniciem uma transmissão num instante de tempo t0. O tempo mínimo para a detecção de uma colisão é o tempo de propagação do sinal emitido por uma estação até a outra estação.

Isto significa, em uma primeira análise que, se uma estação que emitiu um quadro não detecta uma colisão num período de tempo igual ao tempo de propagação do sinal ao longo do canal de comunicação, pode considerar-se possuidora do meio e que as demais estações abstiveram-se de enviar.

Esta análise apresenta, porém, uma imprecisão. Vamos considerar t o tempo de propagação de um sinal entre duas estações, as mais distantes uma da outra. Consideremos que em t0 uma estação emite um quadro. Vamos supor que em t - e , pouco antes que o sinal tenha chegado, outra estação efetue uma transmissão. Ela vai detectar, imediatamente a colisão e interromper a sua emissão. Por outro lado, a perturbação ocasionada pela colisão não vai poder ser detectada pela primeira estação que emitiu o quadro antes de um período de tempo igual a 2t - e. Assim, no pior caso, uma estação só poderá estar segura de que ela adquiriu o acesso ao canal de transmissão após um período de tempo de 2t. Assim, no protocolo CSMA/CD, uma estação emite uma palavra de dados, espera por um tempo 2t e, se

não detectar nenhuma colisão, considera que é a única a usar o meio naquele momento e passa a enviar o restante dos dados.

O método CSMA/CD propicia uma grande otimização no uso do meio em relação aos protocolos anteriores, pois a ocorrência de uma colisão é detectada logo na primeira palavra e a emissão é interrompida, não tendo que ser completamente repetida depois.

No entanto, em todos os métodos de acesso CSMA temos que, quanto maior o número de estações e quanto maior o trafego médio de mensagens na rede, maior a probabilidade de ocorrência de colisões (esta probabilidade aumenta exponencialmente), de forma que o tempo de reação aumenta consideravelmente e não pode ser exatamente determinado. Para muitas aplicações fabris, especialmente aquelas com requisitos de tempo real, é importante utilizar redes com protocolos de acesso ao meio determinísticos. CSMA/CD não apresenta esta característica pois não se sabe se haverão colisões, não se sabe quantas colisões seguidas podem ocorrer e não se conhece de antemão o tempo (aleatório) de espera em caso de colisão. Já para aplicações de escritório os métodos não determinísticos, como CSMA/CD, são em geral satisfatórios.

3.2.2.1.4 - Os protocolos MAC determinísticos

Os métodos de acesso determinísticos são aqueles com tempo de resposta determinável, ao menos em pior caso. Estes métodos podem ser classificados em:

- métodos com comando centralizado (ex.: Mestre-Escravos);
- métodos com comando distribuído (ex.: Token-Passing ou variantes determinísticas do CSMA, como veremos a seguir).

- O protocolo Mestre-Escravos:

Nos sistemas com ***comando centralizado***, somente uma estação pode agir como detentora do direito de transmissão. Esta estação recebe o nome de estação ***Mestre***. O direito de acesso ao meio físico é distribuído por tempo limitado pela estação mestre as demais, que são denominadas estações ***Escravas***. Aqui todas as trocas de dados ocorrem apenas entre mestre e escravos (figura 2.7). A estação mestre realiza uma varredura cíclica de cada uma das estações escravas, solicitando dados ou verificando se uma delas dispõe de dados a enviar. Se uma estação escrava não dispõe de dados a enviar, ela sinaliza esta condição pelo envio de um quadro com formato especifico para este fim. Se, por outro lado, a estação escrava dispõe de dados a enviar, ela envia o quadro de dados correspondente ao mestre. Note que, em ambos os casos, o escravo tem que enviar um quadro em resposta à varredura (scan) do mestre.

Esta configuração deixa o sistema dependente da estação central, mas é a configuração usual dos sistemas de controle na maioria das aplicações, onde o mestre exerce a função de

um controlador (por exemplo, um CLP) e os escravos são sensores e atuadores inteligentes (microprocessados).

Este método de acesso ao meio garante um tempo definido entre transmissões consecutivas a qualquer estação da rede e segue a prática freqüente de fazer um controle distribuído com uma supervisão centralizada. Sendo conhecida a taxa de transmissão e o formato dos quadros usados pelo mestre para realizar a varredura cíclica e pelos escravos para responder, pode-se determinar a duração de um ciclo de varredura completo (questionamento por parte do mestre de todas as estações escravas e respectivas respostas). Este tempo corresponde ao pior caso de tempo de resposta de uma estação qualquer. Naturalmente, este tempo aumenta quando são adicionadas novas estações escravas à rede.

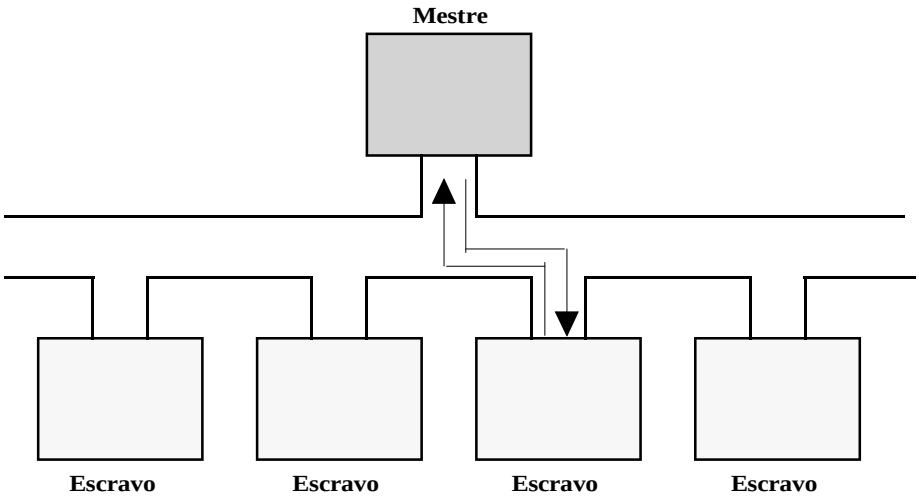


Figura 2.7 - Método de acesso mestre/escravos

- O protocolo Token-Passing:

Os sistemas com ***comando distribuído*** permitem a definição de mais de uma estação com direito de acesso ao meio físico.

Uma das técnicas mais conhecidas nesta categoria é o chamado ***Token-Passing*** (passagem de ficha ou de bastão, como nas corridas de revezamento). Nela, o direito de acesso ao meio (definido pela posse do "token") é transmitido ciclicamente entre as várias estações, que podem livremente trocar dados entre si sem a intromissão de um mestre ou outro intermediário qualquer. A figura 2.8 ilustra a configuração usual da rede conhecida como ***Token-Bus*** (barramento com passagem de ficha).

Este sistema é, no entanto, bem mais complexo do que o Mestre-Escravos visto anteriormente, já que providências especiais tem que ser tomadas no caso da perda do token ou da entrada ou saída de uma das estações da rede durante a operação. Veremos como isto é feito em uma sessão posterior, quando discutirmos a norma IEEE 802.4.

Este método é mais adequado para sistemas nos quais diversas unidades independentes desejam trocar livremente informações entre si.

Neste método, é possível determinar um tempo máximo entre duas oportunidades consecutivas de transmissão para cada estação. Cada estação pode reter o token por um tempo limitado, denominado "*token retention time*", após o que é obrigada a enviar o token (um quadro com formato especial) para a estação seguinte (a ordem de passagem do token é pré-definida, como veremos mais adiante, e forma um anel virtual ou lógico de seqüência de direito de acesso ao meio). Conhecendo-se o tempo de retenção do token e o número de estações acopladas à rede, pode-se calcular o tempo de rotação do token (*token rotation time*, que é o tempo que o token leva para passar por todas as estações uma vez e voltar a estação inicial). Este tempo é o pior caso de tempo de espera para enviar uma mensagem partindo de uma estação qualquer.

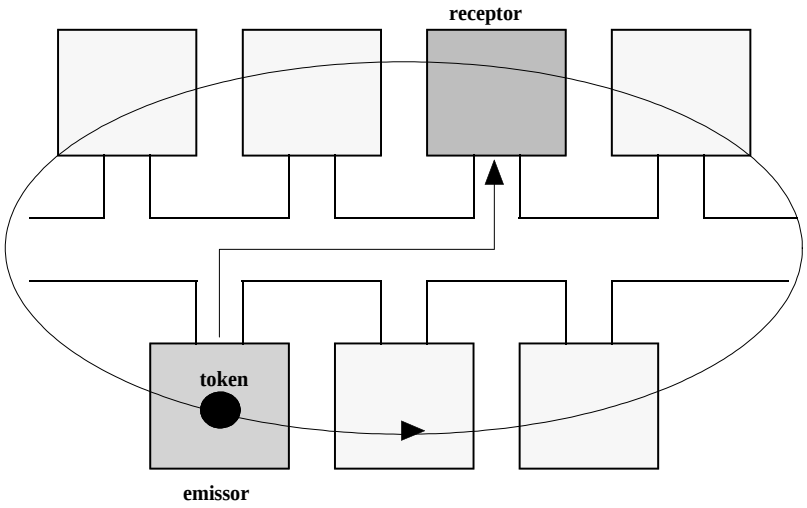


Figura 2.8 - Controle de acesso por passagem de ficha (Token-Bus)

Uma variação desta técnica é o chamado ***Token-Ring***, onde é utilizada uma rede com topologia em ***anel*** em lugar de barramento, como ilustrado na figura 2.9. Aqui o token circula no anel até ser removido por uma estação que deseje transmitir dados. Veremos esta técnica com mais detalhes ao estudarmos a norma IEEE 802.5.

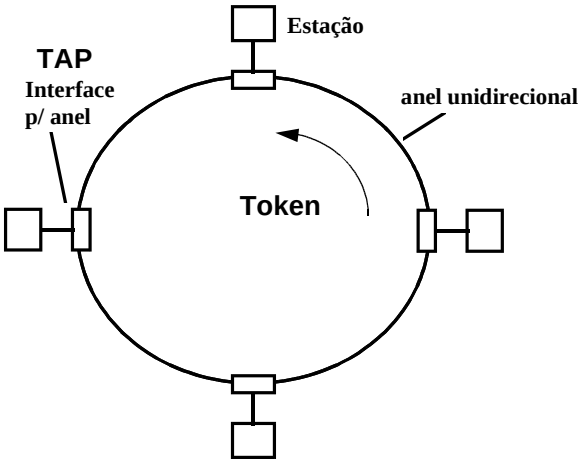


Figura 2.9 - Controle de acesso por passagem de ficha (Token-Ring)

- variantes determinísticas do CSMA:

Recentemente uma série de trabalhos de pesquisa e desenvolvimento tem se preocupado com a definição de protocolos determinísticos baseados em CSMA, devido as vantagens decorrentes da não necessidade de uma entidade controladora de acesso (como a estação mestre no método mestre/escravos ou o token no método Token-Passing). Cada estação pode tentar transmitir sempre que tiver mensagens pendentes, configurando assim um sistema de comunicação com comando distribuído e com elevada autonomia dos nós. O problema básico está em como resolver o problema decorrente das colisões.

Alguns exemplos de soluções são apresentados a seguir:

- **Cabeçalhos Forçantes (Forcing Headers):** Neste método de acesso ao meio, cada mensagem é iniciada por um cabeçalho (header) composto de uma série de bits, que definem sua prioridade. Não podem haver duas mensagens com prioridades idênticas em uma mesma aplicação. A transmissão de um quadro inicia sempre pelo cabeçalho, que é enviado bit a bit em velocidade relativamente baixa, de modo a permitir que o sinal se propague pelo barramento e retorne ao emissor. Após o envio de cada bit, o nível de sinal do barramento é lido. Os bits são codificados de forma que uma colisão tem efeito equivalente a executar uma operação lógica AND sobre cada bit enviado ao barramento (para isto, a detecção de colisão pode ser ativada ao enviar um 1 e desativada ao enviar um 0). A transmissão em uma dada estação é interrompida quando um 1 for enviado por ela e um 0 for lido. Desta forma, se todos os bits do identificador são 0, a prioridade é considerada máxima. Se o cabeçalho for transmitido até o fim sem colisão é porque aquela mensagem era a mais prioritária dentre as envolvidas na colisão e o resto da mensagem é enviado. Assim, a mensagem mais prioritária "atropela" as demais. O método é determinista e, sendo conhecida a prioridade de uma mensagem, pode-se calcular o tempo de entrega da mesma no pior caso. Um exemplo de operação deste protocolo pode ser visto

na figura 2.10. No caso ali representado, 5 estações estão envolvidas na colisão inicial (nós 0 até 4), sendo que as mensagens enviadas por cada uma delas tem headers com valores 000 (mais prioritária), 001, 010, 011 e 100 (menos prioritária) respectivamente.

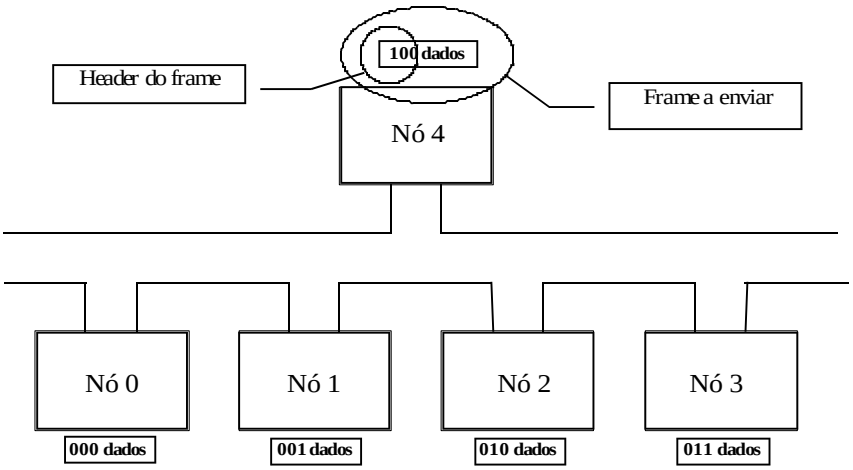


Figura 2.10 - Contenção não destrutiva

Após o envio do primeiro bit do header de cada mensagem (0, 0, 0, 0 e 1 respectivamente), a estação de numero 4 desiste (enviou um 1 e leu um 0, resultante de 0 AND 0 AND 0 AND 0 AND 1). Após o envio do segundo bit, os nós 2 e 3 desistem (ambos enviaram 1 e leram 0). Após o envio do terceiro bit, o nó 1 desiste (enviou 1 e leu 0) e o nó 0 transmite seus dados até o fim (mais prioritário). Em tentativas sucessivas por parte das demais estações, o nó 1 vai transmitir seus dados a seguir, depois o nó 2, depois o 3 e finalmente o 4 (pois tem a mensagem menos prioritária). Para evitar que um nó gerador de uma mensagem de alta prioridade monopolize o barramento tentando transmitir novas mensagens com a mesma prioridade imediatamente após a primeira (por exemplo, se o nó 0 gerar uma nova mensagem antes das demais serem enviadas), o espaço entre frames consecutivos é preenchido por um campo de bits em 1 inserido no final do quadro, com comprimento definido. O barramento só é considerado livre para um nó enviar nova mensagem após ter detectado que o espaço interframes não foi interrompido por um bit em 0. Isto implica em que a estação possuidora da mensagem de alta prioridade terá que esperar ao menos o envio de uma mensagem de prioridade menor para tomar o barramento para si novamente (isto será feito no espaço interframes de mensagem menos prioritária).

- **Comprimento de Preâmbulo (Preamble Length):** a cada mensagem é associado um preâmbulo com comprimento diferente, que é transmitido com a detecção de colisão desativada. Após transmitir o preâmbulo de sua mensagem, cada estação reativa sua detecção de colisão. Neste momento, se for detectada uma colisão, então existe outra

mensagem mais prioritária sendo enviada (com preâmbulo maior) e a estação interrompe sua transmissão. A mensagem com o preambulo mais longo terá portanto a prioridade maior. Isto é ilustrado na figura 2.11, onde 5 estações emitem mensagens simultaneamente com preâmbulos de comprimentos diferentes.

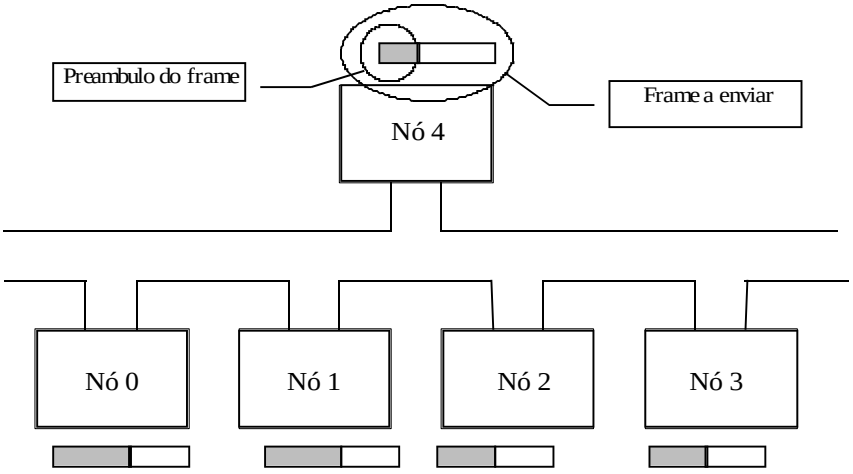


Figura 2.11 - Nós enviando mensagens com preâmbulos de comprimento diferente

O diagrama da figura 2.12 mostra as mensagens lado a lado.

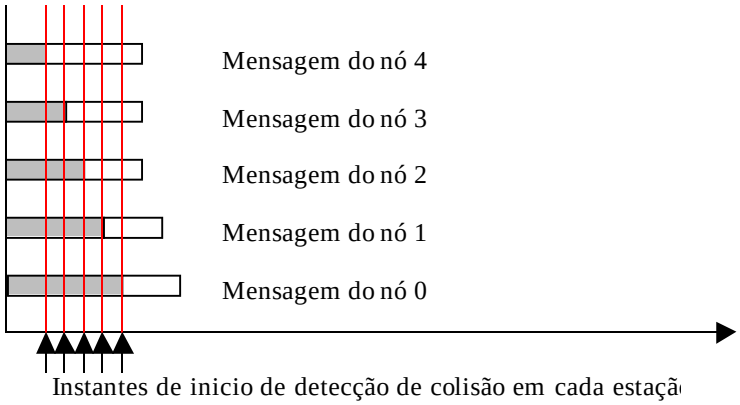


Figura 2.12 - Situação no instante de reinício de detecção de colisão

Como vemos na figura 2.12, a mensagem do nó 4 possui o preâmbulo mais curto e desiste após reiniciar a detecção de colisão. O mesmo acontece com as estações 3, 2 e 1 nos instantes seguintes. Quando a estação 0 reativa a detecção de colisão, já não há mais nenhuma outra estação transmitindo e ele termina de enviar sua mensagem. Em novas tentativas sucessivas, as estações 1, 2, 3 e por último 4 terão sucesso respectivamente.

- **CSMA/DCR (CSMA with Deterministic Collision Resolution):** neste protocolo é realizada uma busca em árvore binária balanceada quando ocorrem colisões. O protocolo é idêntico ao usado na Ethernet até o momento em que ocorre uma colisão. A partir deste momento, o conflito é resolvido de forma determinista, como veremos a seguir. Sua operação será descrita em mais detalhes que os protocolos anteriores, de forma a exemplificar como os tempos de entrega de mensagens podem ser calculados.

No CSMA/DCR , o determinismo é garantido através de busca em árvore binária balanceada. As prioridades são atribuídas a cada estação e não às mensagens, recebendo aqui o nome de "índices".

Para operar corretamente, cada estação deve conhecer:

- o status do barramento, que pode ser:
  - livre
  - ocupado com transmissão
  - ocupado com colisão
- seu próprio índice (prioridade)
- número total de índices consecutivos alocados à fontes de mensagens (**Q**)

O tamanho da árvore binária **q** é a menor potência de 2 maior ou igual a **Q** (ex.: **Q** = 12, **q** = 16). O protocolo opera como CSMA/CD até a ocorrência de uma colisão. Em caso de colisão, é iniciado um período de resolução por busca em árvore binária, denominado "época".

Todas as estações envolvidas na colisão se auto-classificam em dois grupos: os vencedores ou "Winners" (W) e os perdedores ou "Losers" (L):

- W = índices entre [0,q/2[
- L = índices entre [q/2, q]

As estações do grupo W tentam nova transmissão. Se ocorrer nova colisão, é realizada uma nova divisão em grupos:

- W = [0,q/4[
- L = [q/4, q/2[

Se não ocorrer nova colisão (só sobrou uma estação no grupo W), a estação restante transmite seu quadro de dados.

As estações do grupo L desistem e aguardam termino de transmissão bem sucedida de outro nó seguida de meio livre.

Se o grupo W estiver vazio, a busca é revertida, isto é, faz-se uma nova subdivisão de nós a partir do último grupo L:

- W = [q/2, 3q/4[
- L = [3q/4, q[

A época é encerrada quando todas as estações envolvidas na colisão original conseguiram transmitir seus dados sem colisão. O tempo de duração de uma época pode ser calculado, produzindo um resultado determinista.

A seqüência de concessão de direito de acesso ao meio é igual a seqüência de índices crescentes, de modo que os nós mais prioritários transmitem primeiro.

O exemplo a seguir ilustra a operação do protocolo. Uma rede possui 16 fontes de mensagens, das quais 6 tentam transmitir simultaneamente, gerando uma colisão, conforme mostra a figura 2.13.

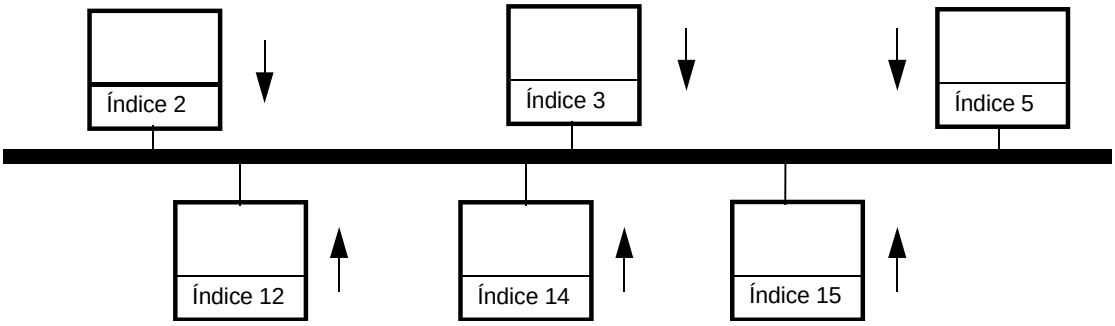


Figura 2.13 - Exemplo de colisão no CSMA/DCR

Neste exemplo temos as seguintes condições:

- 6 estações de uma rede com 16 fontes enviam quadros simultaneamente
- cada estação tem os índices conforme indicados na figura acima
- Q = 16
- q = 16 (2^4)

A altura da árvore binária é dada por $\log_2 16 = 4$. A figura 2.14 mostra a árvore binária balanceada completa para o exemplo. Os números entre colchetes indicam a faixa de índices sendo considerados em cada dicotomia. Os números no interior dos círculos indicam a seqüência de execução da busca na árvore.

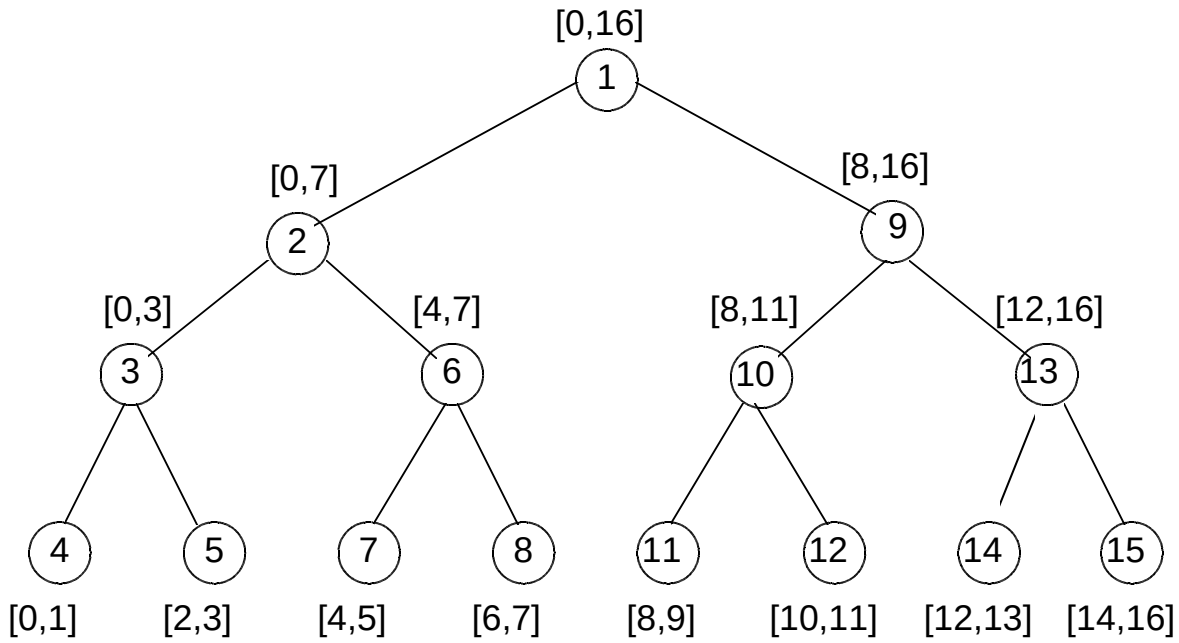


Figura 2.14 - Árvore binária balanceada para Q = 16

A evolução do protocolo (busca na árvore binária) é ilustrado na figura 2.15.

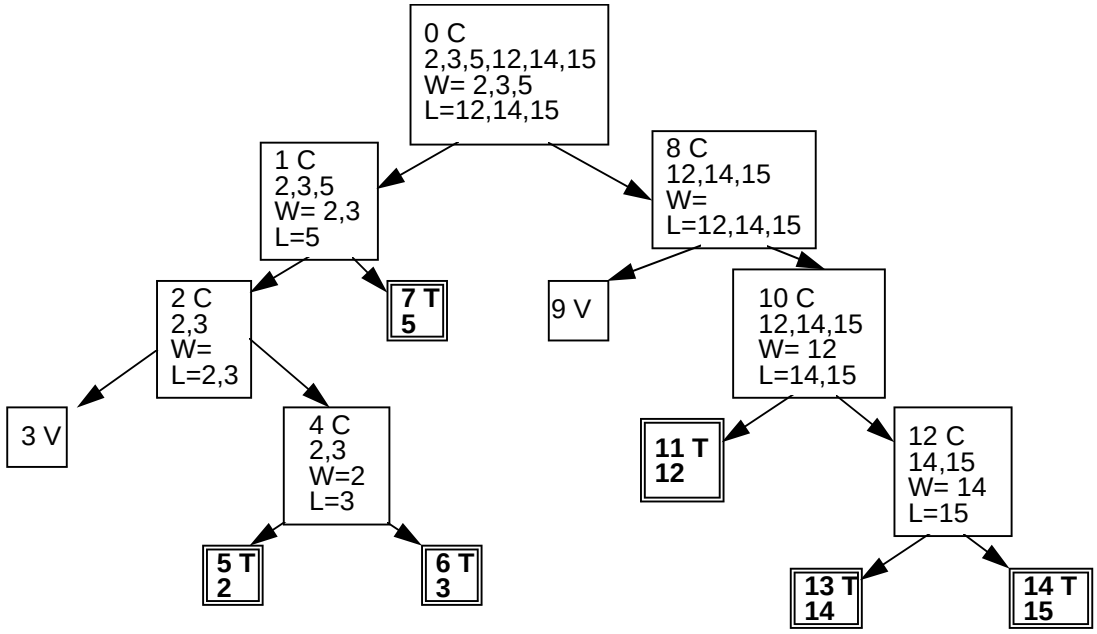


Figura 2.15 - Evolução do algoritmo de busca CSMA/DCR

O tempo até o inicio da transmissão da estação com índice 5 será:

- 5 colisões = 5. slot-time
- 2 transmissões = 2.(tamanho quadro em slot-times)

Assumindo que cada quadro tem um tamanho fixo de 6 slot-times e considerando 1 slot-time como 40 μs, o tempo de espera para a transmissão da mensagem da fonte com índice 5 seria:

- $T_{inicio\ (5)} = 5.40 + 2.6.40 = 680\ \mu s$

Este tempo, no entanto, não representa a situação de pior caso, pois poderiam haver mais estações envolvidas na colisão.

O tempo total de duração da época será:

- 7 colisões = 7.slot-time
- 2 vazios = 2. slot-time
- 6 transmissões = 6 .(tamanho do quadro em slot-times)

Assumindo 1 slot-time = 40 μs, teremos:

- $T_{época} = 7.40 + 2.40 + 6.6.40 = 1800\ \mu s$

Portanto, após 1.8 ms todas as estações envolvidas no conflito terão transmitido suas mensagens. O tempo de pior caso, que é o que nos interessa, pode ser calculado como segue.

Seja:

- $\varphi(v)$ = número de ramos da árvore binária percorridos por uma mensagem proveniente de um nó com índice v
- q = menor potência de 2 maior ou igual ao maior índice disponível
- $\sigma(v)$ = número de potências de 2 contidas em v
- s = 1 slot-time (2 vezes o tempo de propagação do sinal na rede)
- μ = tempo máximo de transmissão da uma mensagem no meio físico (depende do comprimento da mensagem em bits e da taxa de transmissão)

Para uma mensagem participando de uma dada época, temos que:

- $\varphi(v) = \log_2 q + v - \sigma(v)$
- $T_{espera\ (v)} = \varphi(v).s + v.\mu$

Para o exemplo anterior, tomando uma mensagem da estação com índice 5, temos:

- q =16
- v =5
- $\sigma(5) = 2 \quad (5 = 2^2+2^0)$
- $\varphi(5) = \log_2 16 + 5 - 2 = 7$
- $T_{espera\ (5)} = 7.s + 5.\mu$

Assumindo s = 40 μs e μ = 6.s = 240 μs, obteremos para o pior caso de tempo de espera da mensagem da fonte com índice 5 o valor de 1480 μs.

O tempo de duração da época, no pior caso, é dado por:

$$T_{época} = \varphi\,(q-1).s + Q.\mu = T_{busca} + T_{transmissão}$$

Para uma mensagem que chega a fila de emissão de uma fonte com índice **v** em um instante qualquer, o pior caso de tempo de espera é maior, pois a nova mensagem pode chegar na fila imediatamente após o inicio de uma época, da qual ela ainda não faz parte. Neste caso, o pior caso do tempo de espera será dado por:

$$T_{espera\,(v)} = T_{época} + \varphi\,(v).s + v.\mu$$

3.2.2.1.5 - Abordagens para comunicação tempo real

A definição de um protocolo determinístico de acesso ao meio ainda não resolve completamente nosso problema. A solução global do problema da comunicação em tempo real inclui a realização de um ***escalonamento das mensagens***. A exemplo do que se faz em sistemas multitarefas, onde **processos concorrentes** são escalonados segundo vários critérios de forma a definir qual deles terá acesso ao **processador** em um dado momento, aqui deve ser realizado um escalonamento de **mensagens** de forma a definir qual delas terá acesso ao **meio de comunicação** em um dado momento.

A tabela 2.1 apresenta de forma resumida algumas soluções apresentadas na literatura para a problemática da comunicação em tempo real.

Uma abordagem proposta é a atribuição de prioridades fixas às mensagens e a realização de um teste off-line de escalonabilidade sobre o conjunto. Neste caso, o protocolo MAC tem que ser capaz de distinguir prioridades e enviar a mensagem mais prioritária entre todas as estações primeiro. Exemplos de protocolos adequados neste caso seriam CSMA/CA ou Comprimento de Preâmbulo.

Outra proposta seria a realização de um escalonamento on-line das mensagens. O MAC utilizado deve apenas ter um tempo de transmissão limitado.

Outra classe de soluções são os chamados protocolos de reserva, baseados em conhecimento global do sistema, isto é, cada estação sabe que mensagens estão pendentes nas demais estações. Para isto, é necessário que o estado das filas locais seja transmitido às demais estações em intervalos de tempo (time slots) reservados para este fim.

Abordagem	Requistos	Ex.de Protocolos
Atribuição de Prioridades com teste de escalonabilidade Off-line (em tempo de projeto)	MAC com resolução de prioridades	Token-Ring c/Pr.
		Dif. atrasos
		Comp. Preâmbulo
		Forcing Headers (CSMA/CA)
Circuito Virtual TR com escalonamento On-line de mensagens	MAC com tempo de acesso ao meio limitado	TDMA
		Token-Passing
		Waiting Room
		CSMA/DCR
Reserva com escalonamento global	Requer cópias locais de todas as filas de mensagens, difundidas em "slots times" de reserva	PODA

Tabela 2.1 - Abordagens para comunicação tempo real

A camada de enlace de uma rede para tempo real deve prover ao usuário ou ao software da camada logo acima um conjunto mínimo de serviços, tais como:

- Serviços sem conexão:
    - SEND (identificação do receptor, mensagem, requisitos Tempo Real);
    - mensagem = RECEIVE (emissor);

Os requisitos Tempo Real podem ser expressos sob a forma de uma prioridade ou um tempo limite de entrega (deadline).

- Serviços com conexão:
    - rtcid = CONNECT(receptor, requistos TR);
    - SEND (rtcid, mensagem);
    - mensagem = RECEIVE (rtcid);
    - DISCONNECT(rtcid);

Aqui, "rtcid" significa "real time connection identifier", isto é, um identificador para a conexão.

3.2.2.2. CONFIABILIDADE

Em aplicações industriais onde são transmitidos muitos códigos de comando, leitura de medidores e comando de atuadores, um erro de um Bit qualquer pode ter conseqüências desastrosas. A transferência de programas para máquinas de Comando Numérico, por exemplo, exige um sistema altamente confiável, pois são transmitidos códigos de comando

cuja mínima alteração pode produzir danos de elevado custo. Desta forma, redes industriais de comunicação tem que oferecer uma elevada confiabilidade.

Para aumentar esta confiabilidade nas mensagens transmitidas, normalmente é usado um teste cíclico de redundância (CRC - Cyclical Redundance Check).

Em sistemas que necessitem de uma operação contínua, pode ser utilizado um meio de transmissão e estações de controle redundantes. Além disso, os cabos utilizados em geral são blindados.

3.2.2.3. REQUISITOS DO MEIO AMBIENTE

Devido às características do ambiente industrial, a presença de perturbações eletromagnéticas, provocadas principalmente pelos acionamentos de motores elétricos de grande porte ou outras fontes chaveadas (estações de solda, conversores estáticos, etc.), não pode ser desprezada na escolha do meio de transmissão de uma rede de comunicação.

Para a definição do meio físico de transmissão e do protocolo de comunicação, estas características devem ser consideradas. O meio de transmissão deve possuir uma boa resistência mecânica e deve estar eletricamente isolado.

O meio físico a ser adotado não depende apenas de aspectos técnicos mas também (e muito especialmente no chão de fábrica) do aspecto de custo. Cabos coaxiais são menos afetados por perturbações eletromagnéticas do que o par trançado. No entanto, o custo do cabo coaxial é superior ao do par trançado. Além disso, barramentos construídos com cabo coaxial e conectores T requerem resistências terminais (“terminadores”) para a correta operação da rede. Isto é necessário para evitar ressonâncias ou ecos, onde o sinal propagado na rede retorna sobre si mesmo. Se o cabo for aberto em qualquer ponto da fábrica, a rede cai.

Futuramente, a adoção de fibras óticas poderá vir a ser a melhor solução tanto do ponto de vista técnico quanto econômico. Atualmente ainda há dificuldades na realização de bifurcações com este meio físico, necessárias para as conexões em T usadas em redes com topologia em barramento, de modo que as fibras óticas são mais utilizados em sistemas com topologia em estrela ou anel. A realização de bifurcações tem alcançado melhores resultados adotando acopladores ativos (com eletrônica adicional para conversão do sinal ótico em elétrico e vice-versa nos pontos de derivação). Trabalhos vem sendo realizados também para a realização de bifurcações passivas, baseadas em prismas. Tanto as bifurcações ativas quanto as passivas encarecem a conexão, o que torna a solução antieconômica para o nível de chão de fábrica. Uma solução que vem ganhando terreno é o uso de Hubs, que emulam um barramento para as placas mas que efetivamente atuam como estações concentradoras, transformando a topologia física em estrela, o que permite o uso de fibras óticas sem a necessidade de bifurcações. Com fibras óticas, além disso, pode-se trabalhar com freqüências da ordem de vários GigaBaud (10^9 bits por segundo), o que permitiria uma melhoria de performance do sistema de comunicação (esta, no entanto, não é a vantagem chave da fibra ótica em redes

locais industriais, pois o gargalo em termos de tempo de transmissão não esta na camada física, como veremos a seguir). Diversos trabalhos de pesquisa vem sendo realizados no sentido de resolver os problemas técnicos existentes.

A figura 2.16 apresenta uma comparação sumária entre os três tipos de meio.

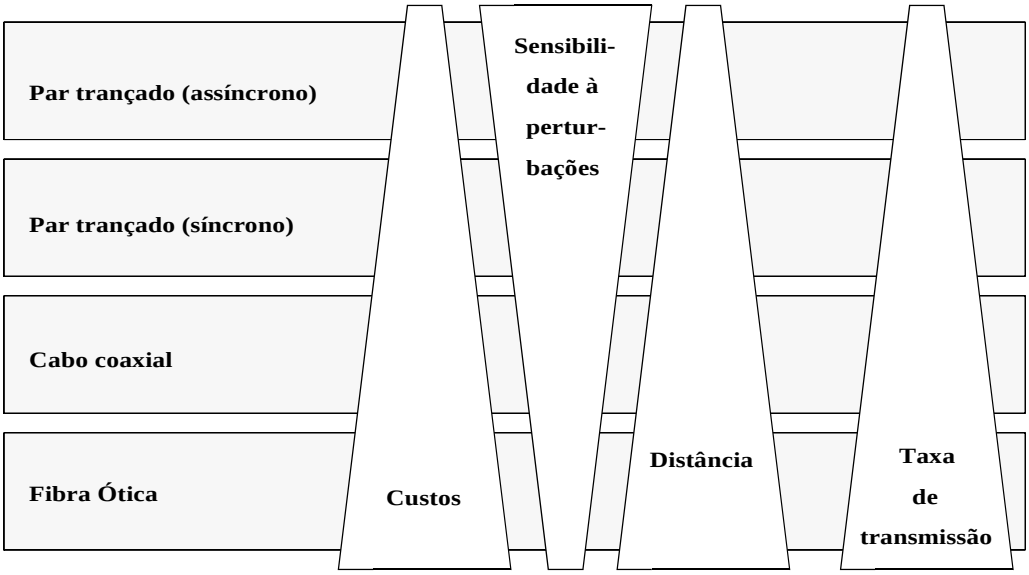


Figura 2.16 - Meios de transmissão

Atualmente, a melhor solução para o chão de fábrica ainda é o par trançado. A fibra ótica será certamente a melhor solução em futuro próximo, já existindo hoje diversas aplicações industriais bem sucedidas.

Áreas de segurança intrínseca:

•Sujeitas a incêndio, explosão

•Presença de líquidos ou gases inflamáveis/explosivos

•Não pode haver faiscamento

•Freqüência de sinais elétricos limitada

•Modelo FISCO (Fieldbus Intrinsically Safe Concept): desenvolvido na Alemanha pelo PTB (Physikalisch Technische Bundesanstalt) e reconhecido mundialmente como modelo básico para operação de redes em áreas de risco de explosão ou incêndio.

•Princípios de transmissão segundo modelo FISCO:

–Cada segmento possui uma única fonte de alimentação.

–Não se alimenta o barramento enquanto uma estação está enviando.

–Cada dispositivo de campo consome uma corrente constante em steady-state de pelo menos 10 mA, que alimenta o dispositivo.

–Os dispositivos de campo funcionam como uma carga passiva de corrente.

–Existe uma terminação passiva em ambos os extremos da rede.

–Topologias permitidas: linear, em árvore e em estrela.

•Norma IEC 1158-2 para camada física:

–Transmissão de dados: digital, bit - síncrona, Manchester

–Taxa de transmissão: 31,25 kbit/s, modo voltagem

–Cabo: STP com 2 fios

–Alimentação remota: opcional, via linhas de dados

–Classes de proteção contra explosão: Intrinsically safe (EEx ia/ib) e encapsulation (EEx d/m/p/q)

–Topologias: linha e árvore ou uma combinação

–Numero de estações: até 32 estações por segmento, máximo de 126 com 4 repeaters

3.2.2.4. TIPO DE MENSAGENS E VOLUME DE INFORMAÇÕES

Nos níveis hierárquicos superiores de automação (por ex. a nível de planejamento) são freqüentemente trocados pacotes de vários KByte, que requerem tempos de transmissão variando de alguns segundos até vários minutos. Nas aplicações mais próximas ao processo, normalmente são enviadas mensagens curtas, tais como:

- Comando para ligar ou desligar uma unidade (basta um bit);
- Comando para fazer a leitura de um sensor / medidor (bastam 8, 12 ou 16 bits, conforme a resolução do conversor A/D utilizado);
- Comando para alterar o estado de um atuador (idem acima);
- Comando para verificar o estado de uma chave ou relê (basta um bit).

Estas operações podem ser feitas normalmente com um único quadro de comando acrescido dos respectivos dados, quando existirem.

Para dispositivos programáveis encontrados no ambiente industrial (Controladores Lógicos Programáveis, Comandos Numéricos Computadorizados, Comandos de Robô, etc.), normalmente é necessário o envio de programas no inicio da produção de um lote, que caracterizam mensagens maiores do que as referidas anteriormente. No entanto, estes programas raramente ultrapassam 10 KBytes em tamanho e dificilmente são utilizados mais de 3 programas por unidade de fabricação durante um turno de trabalho.

Como conseqüência, uma taxa de transmissão de dados relativamente baixa a nível da camada física atende as necessidades de comunicação na maioria dos casos (1 Mbps é quase sempre suficiente). Por outro lado, tem-se uma elevada taxa de ocupação do barramento, com um grande número de mensagens sendo trocadas constantemente. Deve-se portanto evitar mensagens grandes, que podem monopolizar o meio de transmissão por um tempo muito longo.

3.2.3. Projetos de Padronização de redes industriais

As exigências de comunicação entre unidades para a integração flexível dos sistemas de automação, descritas nos itens anteriores, evidenciam a necessidade de uma especificação de redes locais para aplicações industriais diferente daquela adotada em automação de escritório. Existem diversas redes proprietárias para ambiente fabril, desenvolvidas por grandes empresas e que normalmente utilizam um protocolo específico desenvolvido pelo próprio fabricante. Estas redes não permitem a interligação de equipamentos de outros fabricantes. Desta forma o usuário fica na total dependência de um único fornecedor.

A arquitetura das redes de comunicação industrial deve integrar sistemas heterogêneos de diferentes fabricantes, suportando tanto a operação de chão de fábrica quanto as funções de apoio à produção. A definição de padrões de protocolos de comunicação e a sua adoção por diferentes fabricantes deverá permitir a interconexão (interoperabilidade) e até mesmo a intercambiabilidade das várias unidades de processamento (neste caso, equipamentos produzidos por fabricantes diferentes podem ser facilmente incorporados à instalação, simplesmente conectando-os ao sistema de comunicação). Entre as diversas iniciativas para padronização para redes industriais, podemos destacar: Projeto PROWAY, Projeto IEEE 802, Projeto MAP (incluindo MAP/EPA e MINI-MAP), projeto TOP e Projeto FIELDBUS.

3.2.3.1. PROJETO PROWAY

Em 1975 um grupo da IEC (International Electrotechnical Comission) iniciou seus trabalhos para a normalização de redes de comunicação para controle de processos. Destes trabalhos resultou a proposta PROWAY (Process Data Highway). O Proway passou por diversas fases (Proway A, B e C) em função dos trabalhos de padronização internacionais. Proway A e B utilizavam o protocolo HDLC da ISO na camada de enlace, adequando-se assim apenas a sistemas tipo mestre/escravos. Proway C adotou a técnica de token-passing. Sua arquitetura é composta de 4 camadas do modelo de referencia ISO/OSI, denominadas "Line" (camada física), "highway" (camada de enlace), "network" (camada de rede) e "application" (camada de aplicação).

3.2.3.2. PROJETO IEEE 802 E ISO/IEC 8802

O IEEE (Institute of Electrical and Electronics Engineers) iniciou em 1980 o projeto conhecido pelo número 802, que definiu originalmente uma série de normas para as camadas Física e Enlace do modelo de referência OSI. Na proposta do IEEE, a camada de Enlace é subdividida em duas subcamadas: LLC (Logical Link Control) e MAC (Medium Access

Control). Estas definições foram aceitas pelos demais organismos de padronização. A norma resultante é hoje reconhecida internacionalmente sob a designação ISO/IEC 8802.

O projeto, em sua forma atual, define uma norma com 12 partes, designadas por IEEE 802.1 até IEEE 802.12, como mostra a figura 2.17, e que tratam de vários aspectos ligados a redes de computadores, indo bastante além da proposta original.

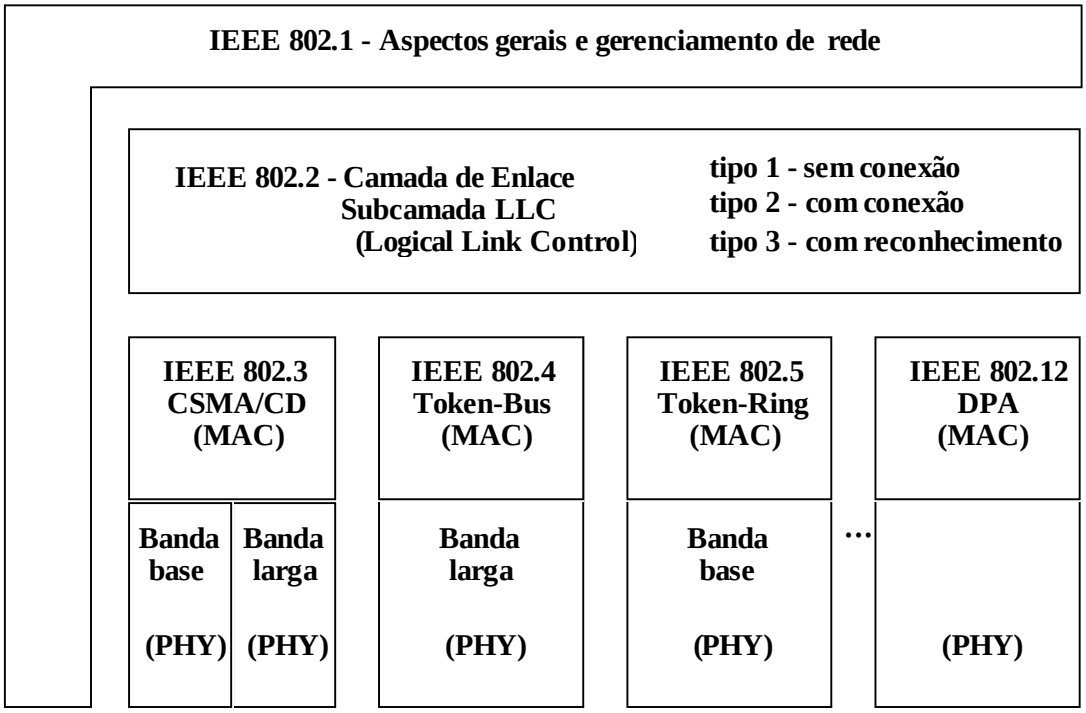


Figura 2.17 - Especificações do projeto IEEE 802

Resumidamente, as partes do projeto 802 abordam os seguintes aspectos:

- IEEE 802.1 : definição de serviços de gerenciamento de redes e generalidades;
- IEEE 802.2 : descrição da sub-camada LLC da camada de Enlace. Aqui a norma prevê três tipos de serviços:
  - LLC tipo 1: a troca de dados se dá sem o estabelecimento prévio de uma conexão. Não é feito controle de erros nem de fluxo e o receptor das mensagens não envia um quadro de reconhecimento ao emissor;
  - LLC tipo 2: antes de realizar qualquer troca de dados, as estações envolvidas na comunicação devem estabelecer uma conexão entre si. Neste caso, é feito controle de erros e de fluxo e a entidade receptora envia um quadro de reconhecimento para cada mensagem recebida;

    - LLC tipo 3: a comunicação é feita sem o estabelecimento prévio de uma conexão, mas é realizado controle de fluxo e de erros e o receptor envia um quadro de reconhecimento ao emissor para cada mensagem recebida.
- IEEE 802.3 : descrição da sub-camada MAC e camada Física para redes com topologia em barramento e método de acesso ao meio baseado em CSMA/CD;
- IEEE 802.4 : descrição da sub-camada MAC e camada Física para as redes com topologia em barramento e método de acesso ao meio baseado em "token-passing" (Token-Bus);
- IEEE 802.5 : descrição da sub-camada MAC e camada Física para as redes com topologia em anel e método de acesso ao meio baseado em "token-passing" (Token-Ring);
- IEEE 802.6 : descrição da sub-camada MAC e camada Física para as redes metropolitanas com DQDB (Distributed Queue Dual Bus ou barramento dual com filas distribuídas);
- IEEE 802.7 : contém recomendações do IEEE para LANs usando Broadband. Na versão da ISO/IEC, define uma subcamada MAC com slotted ring e a camada física correspondente;
- IEEE 802.9 : IS (Integrated Services) para integrar LANs com RDSI (Rede Digital de Serviços Integrados, ISDN em inglês) e FDDI (Fiber Distributed Data Intarface);
- IEEE 802.10 : aborda questões de segurança na interoperação de LANs e MANs (atualmente define o padrão SDE, Secure Data Exchange);
- IEEE 802.11 : padroniza LANs com MAC sem fio (Wireless) e a camada física correspondente (transceivers de rádio);
- IEEE 802.12 : método de acesso com demanda priorizada (DPA, Demand Priority Access) e camada física correspondente.

Analisaremos mais em detalhes as propostas mais relevantes para aplicações em automação industrial nas subseções seguintes.

A norma IEEE 802.3 (CSMA/CD)

Esta norma define um protocolo pertencente à família CSMA/CD 1-persistente, onde uma estação desejando emitir escuta o cabo para verificação de disponibilidade; se este não está disponível, a estação aguarda pela sua liberação; caso contrário, esta vai emitir o quadro considerado. Na ocorrência de emissões simultâneas de quadros (provocando colisões), todas as estações envolvidas na colisão interrompem imediatamente (a partir da detecção da ocorrência da colisão) a sua emissão e esperam um período aleatório de tempo para recomeçar as transmissões.

Um aspecto interessante da norma IEEE 802.3 é a sua origem, baseada na definição da rede Ethernet, definida pela Xerox em 1976. Esta rede definia um protocolo CSMA/CD sobre um cabo coaxial de 1000 metros de comprimento, com uma taxa de transmissão de 3 Mbps, sobre o qual 100 estações poderiam ser conectadas. O sucesso desta rede foi reconhecido também a nível de outras empresas, de tal maneira que a própria Xerox, DEC e Intel definiram posteriormente um padrão "de fato" (isto é, um padrão consagrado pelo uso) para uma rede Ethernet, com taxa de transmissão de 10 Mbps.

Ethernet serviu de base para a definição da norma IEEE 802.3, sendo que as diferenças fundamentais da norma em relação a esta rede estão na definição de toda uma família de protocolos CSMA/CD 1-persistentes, utilizando diferentes suportes de transmissão, cujas taxas de transmissão vão de 1 a 10 Mbit/s. Os parâmetros iniciais da norma são orientados a um canal de 10 Mbps em banda de base, o suporte utilizado sendo um cabo coaxial de 50 ohms. O comprimento máximo de um suporte físico segundo a norma IEEE 802.3 é de 500 m.

O formato de um quadro IEEE 802.3 é mostrado na figura 2.18. Cada quadro é iniciado por um preâmbulo de 7 bytes, cada um deles contendo a seqüência 10101010. O byte seguinte é o delimitador de início de quadro, contendo a seqüência 10101011. Estes dois campos do quadro permitem às estações sincronizar-se, respectivamente, a nível de bit e de caractere.

Cada quadro é composto de dois endereços físicos, o do destino e o da origem, sendo que os endereços podem ser representados segundo dois formatos possíveis, em 16 ou 48 bits. O bit mais significativo dos endereços permite definir se o endereço é individual (0) ou de grupo (1). O endereçamento de grupo permite que um conjunto de estações da rede receba o mesmo quadro, caracterizando uma comunicação em multicast. Caso o endereço do destinatário seja composto exclusivamente de bits a 1, o quadro será enviado a todas as estações da rede, caracterizando uma comunicação em broadcast.

O campo seguinte contém o tamanho do campo de dados transmitidos, expresso em bytes. O campo de dados pode variar de 0 a 1500 bytes.

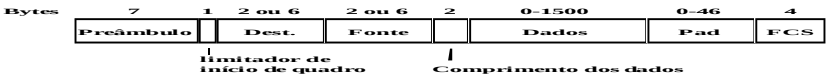


Figura 2.18 - Formato do quadro IEEE 802.3

A norma define a utilização de um quadro que não ocupe menos de 64 bytes, do endereço do destinatário ao campo de redundância cíclica (FCS, Frame Check Sequence). Desta forma, caso a zona reservada aos dados seja inferior a 46 bytes, o quadro deve ser

completado através do campo PAD, de preenchimento de quadro, cujo tamanho poderá ser menor ou igual a 46 bytes. Isto é necessário para garantir a detecção de colisão.

O último campo do quadro contém uma palavra de 32 bits para o controle de erros, utilizando a técnica de redundância cíclica (CRC, Cyclic Redundancy Check). O polinômio gerador utilizado é $G(x)=x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^{8}+x^{7}+x^{5}+x^{4}+x^{2}+1$.

O funcionamento deste protocolo é baseado, então, no que foi apresentado anteriormente, onde duas ou mais estações que detectem a disponibilidade do suporte de transmissão, vão emitir os seus quadros respectivos gerando uma colisão. Toda estação que detecte a ocorrência de colisão deve interromper imediatamente a sua emissão, gerando uma seqüência de interferência para advertir as outras estações da ocorrência da colisão. As estações deverão, então, esperar um período de tempo aleatório para recomeçar o ciclo de emissão. O tempo de espera é dividido em intervalos de tempo, que correspondem a duas vezes o tempo de propagação entre as duas estações mais distantes da rede, este intervalo de tempo sendo definido como *time slot*.

Após a primeira colisão, cada estação vai esperar 0 ou 1 time slot antes de tentar novamente a emissão. Caso duas estações escolham o mesmo número de time slots, elas serão envolvidas mais uma vez numa colisão. Após a segunda colisão, cada uma delas vai escolher, de maneira aleatória, um tempo de espera compreendido entre 0 e 3 time slots antes de tentar a emissão. Caso uma terceira colisão ocorra, a faixa de escolha de intervalos de tempo vai aumentar para 0 a $(2^3 - 1)$. Em geral, após i colisões, o intervalo de escolha será de 0 a $(2^i - 1)$. Assim, se 10 colisões consecutivas ocorrem (as chances para isto são mínimas mas reais), a faixa de tempo de espera vai atingir um máximo de 1023 time slots. Após a décima sexta colisão, a tentativa de emissão é desabilitada e a estação de trabalho será informada da situação, ficando uma camada mais acima como responsável pela decisão sobre a melhor forma de tratar o problema. A escolha deste algoritmo se justifica pela possibilidade de adaptação do tempo de espera em função do número de colisões ocorridas.

Por outro lado, o algoritmo CSMA/CD não fornece nenhuma informação sobre o destino de quadros que tenham sido transmitidos sem a ocorrência de nenhum incidente. Isto significa que a ausência de colisão não garante que o quadro foi recebido pela estação (ou estações) destinatária(s), sem nenhuma modificação devido, por exemplo, a sinais parasitas. Para que a transmissão seja considerada correta, é preciso que a estação receptora verifique o bloco de controle de redundância cíclica e, em caso positivo, transmita um quadro de reconhecimento. Isto será feito se for escolhido um serviço confiável a nível da subcamada acima, a LLC.

A nível das camada física e enlace, a norma IEEE 802.3 prevê uma arquitetura conforme a figura 2.19.

Enlace	LLC (Logical Link Control)
	MAC (Medium Access Control)
Física	PLS (Physical Layer Signaling)
	AUI (Attachment Unit Interface)
	MAU (Medium Attachment Unit)
	MDI (Medium Dependent Interface)

Figura 2.19 - Camadas Física e Enlace conforme IEEE 802.3

A subcamada PLS (Physical Layer Signaling) define a interface entre o nível físico e a subcamada MAC. A PLS fornece à subcamada MAC serviços de envio e recepção de bits e de detecção de colisão.

A AUI (Attachment Unit Interface) é usada em algumas configurações e consiste de um conjunto de cabos tipo par trançado blindado que permitem conectar à rede estações localizadas a uma certa distância do meio de transmissão (cerca de 50m no máximo). A AUI interliga a placa de rede propriamente dita (situada dentro do computador, em um slot) ao dispositivo transceptor chamado MAU (Medium Attachment Unit). O MAU é um dispositivo eletrônico que transmite, recebe e detecta a presença de sinais no meio e deve estar fisicamente muito próximo a este.

A conexão entre o MAU e o meio físico em si é feito por meio de um conector denominado MDI (Medium Dependent Interface).

A norma IEEE 802.3 define várias opções de meio físico e taxa de transmissão, especificadas da forma:

<taxa em Mbps><técnica de sinalização><tamanho máximo do segmento * 100>

Desta forma, a especificação 10BASE5, por exemplo, define uma camada física com taxa de transmissão de 10Mbps, técnica de sinalização em banda BASE (*baseband*) e comprimento máximo do cabo de 500 metros.

Veremos a seguir algumas das especificações mais comuns.

A figura 2.20 mostra a especificação 10BASE5 já mencionada. Nesta especificação é usada uma AUI para ligar a estação a uma MAU externa, que pode estar a no máximo 50 metros de distância e é alimentada de energia através do próprio cabo AUI. Utiliza-se aqui

como meio de transmissão o cabo coaxial grosso (thick coaxial cable) de 50 Ohm. A MAU é ligada ao cabo por meio de um conector de pressão.

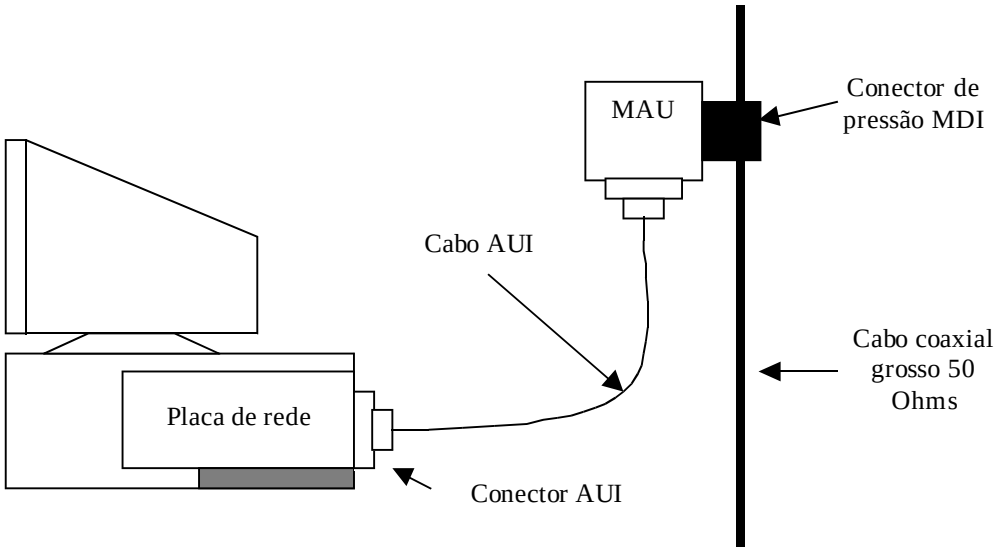


Figura 2.20 - Especificação IEEE 802.3 10BASE5

A figura 2.21 ilustra a especificação 10BASE2, que trabalha com uma taxa de transmissão de 10Mbps, banda base e um cabo de até 200 metros. A idéia aqui foi fazer uma especificação mais simples e mais barata que a 10BASE5. Nesta especificação as funções da MAU ficam dentro da placa de rede na estação, que é ligada diretamente a um cabo coaxial fino (thin coaxila cable) de 50 Ohms por meio de conectores BNC. Desta forma não é necessário o uso do cabo AUI. A MDI é um conector BNC fêmea. Em função do menor preço e da facilidade de interligação, esta especificação é amplamente difundida.

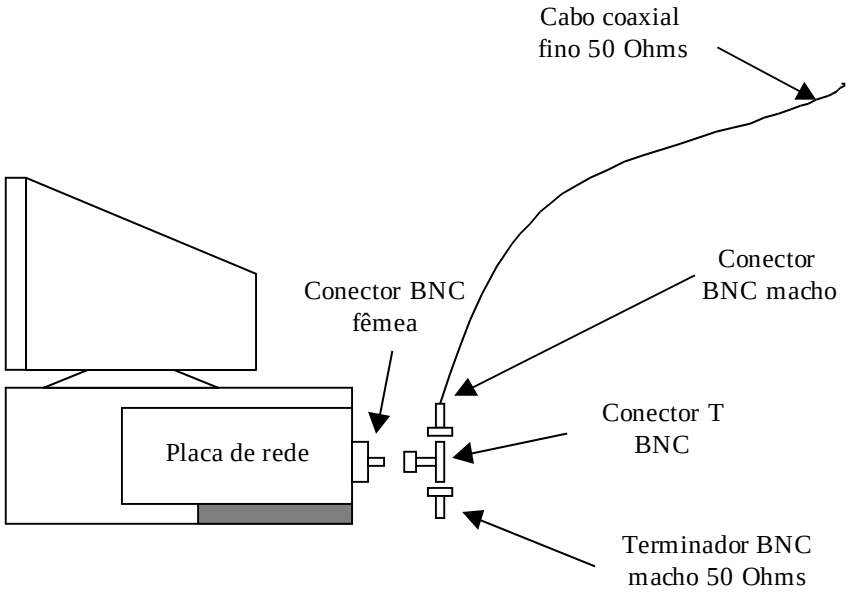


Figura 2.21 - Especificação IEEE 802.3 10BASE2

Além das especificações vistas acima, existe ainda a especificação 10BROAD36, que opera com taxa de transmissão de 10Mbps, técnica de sinalização em Banda Larga e um cabo de 3600 metros.

A norma IEEE 802.3 prevê ainda algumas especificações de MAU, como:

- 10BASE-T, que define uma MAU para par trançado em lugar de cabo coaxial e é usualmente empregada para conexão com repetidores multiporta (mais conhecidos como Hubs);
- 10BASE-FL, que define uma MAU para fibra ótica, usada para conectar uma estação a um Hub;
- 10BASE-FB, que define uma MAU para interligar repetidores entre si, usada em redes backbone;
- 10BASE-FP, que define uma MAU para operar como estrela passiva;

IEEE 802.3u (Fast Ethernet)
•3 versões com 100 Mbps, sempre com HUB:
–**100BASE-T4**: usa 4 pares de cabos UTP categoria 3 (fio telefônico), com sinalização em 25MHz cada, com até 100m até HUB, modo half-duplex.
–**100BASE-TX**: usa 2 pares de cabos UTP categoria 5 (usa isolante de teflon), um para o HUB e outro de retorno, até 100m até o HUB, modo full-duplex;
–**100BASE-FX**: usa 2 fibras óticas multimodo, uma em cada direção, distância de até 2 Km até HUB.
Switched Ethernet:
•Melhora de performance da ethernet pode ser obtida com fast ethernet, porém requer novas placas de rede

•Outra solução: manter placas 10BASE-T e ligar a um switcher

IEEE802.3z: 1000BASE-F

A norma IEEE 802.4 (barramento com ficha - "Token Bus")

Embora IEEE 802.3 tenha sido adotada em muitas aplicações, particularmente aquelas destinadas à automação de escritório, muitos especialistas da área de automação, como por exemplo da General Motors, colocam algumas restrições a ela. Um primeiro problema é o aspecto não determinístico do protocolo, onde, em alguns casos, uma estação pode ser condenada a esperar um longo intervalo de tempo para que um quadro possa ser transmitido. Outro aspecto é a impossibilidade, devido ao formato dos quadros IEEE 802.3, da definição de mecanismos de prioridade de quadros, ponto importante no caso de aplicações industriais.

A norma 802.4 procura cobrir estes aspectos, permitindo a comunicação num suporte de transmissão do tipo barramento, mas baseado na existência de uma ficha (token) que representa o direito de transmissão de uma estação.

Embora as estações sejam conectadas através de um barramento, caracterizando uma estrutura linear ou arborescente de comunicação, estas podem ser vistas como se constituíssem um anel lógico, onde cada estação conhece o endereço das estações vizinhas no anel. Uma vez que o anel lógico é inicializado, a estação que possui o endereço mais elevado pode transmitir o primeiro quadro. Uma vez que esta transmitiu o seu quadro, ela cede o direito de transmissão à estação vizinha, enviando a esta um quadro especial denominado ficha (token). Desta forma, a ficha se propaga ao longo do anel lógico, tendo como regra fundamental o fato de que apenas a estação possuidora da ficha tem o direito de transmitir um quadro. Sendo assim, a ocorrência de colisões é praticamente impossibilitada.

Um ponto importante a salientar é a total independência entre a localização física das estações na rede e a composição do anel lógico. Quando uma estação transmite a ficha, ela a endereça à estação imediatamente em aval no anel lógico, independente da sua localização física.

Ainda, por construção, o suporte de transmissão opera em modo difusão, ou seja, todas as estações recebem os quadros transmitidos, mas levam em consideração somente aqueles que lhes sejam diretamente endereçados. Além disso, quando uma estação conectada ao barramento entra em funcionamento, ela não é automaticamente inserida no anel lógico. O protocolo IEEE 802.4 é quem assume a função de inserção ou retirada de uma estação do anel lógico.

O protocolo opera da seguinte maneira: quando o anel lógico é inicializado, as estações de trabalho colocam-se seqüencialmente segundo a ordem descendente do valor do seu endereço físico. A passagem da ficha se dá segundo esta mesma ordem. A cada vez que uma estação torna-se proprietária da ficha, ela possui o direito exclusivo de transmissão sobre o barramento, este direito podendo ser exercido durante um certo período de tempo, após o qual ela deve ceder a ficha para a próxima estação do anel. No caso de quadros curtos, um grande número deles podem ser transmitidos durante este período, de maneira consecutiva. Se uma estação possuidora da ficha não tem quadros a enviar, ela passa a ficha adiante à estação seguinte do anel lógico.

O protocolo de barramento com ficha define um mecanismo de prioridades a quatro níveis, referenciados por 0, 2, 4 e 6, o nível 0 tendo a mais baixa prioridade e o nível 6 a mais alta. Isto pode ser visto como se cada estação fosse dividida em quatro subestações de quatro níveis de prioridade. Assim, quando uma estação recebe a ficha, o direito de transmissão é cedido à subestação de nível 6, após à estação 4, e assim por diante, até a estação 0, até que o período de emissão seja esgotado e a ficha tenha de ser retransmitida à próxima estação do anel lógico.

A figura 2.22 apresenta o formato de um quadro IEEE 802.4, onde os seguintes campos são definidos:

- Preâmbulo, que permite a sincronização do emissor e receptor a nível de bit;
- Delimitador de Início de Quadro, que permite a sincronização a nível de caractere;
- Controle de Quadro, que permite fazer a distinção entre os quadros de dados e os quadros de controle do anel lógico;
- os campos Endereço Destino e Endereço Origem, que, do mesmo modo que na norma IEEE 802.3, podem ser codificados em 16 ou 48 bits;
- campo de Dados, que pode ter até 8182 bytes de comprimento;
- campo de Controle, que permite efetuar o controle de erros através da técnica de redundância cíclica (CRC), como na norma IEEE 802.3;
- Delimitador de Fim de Quadro, que indica o fim da transmissão de um quadro.

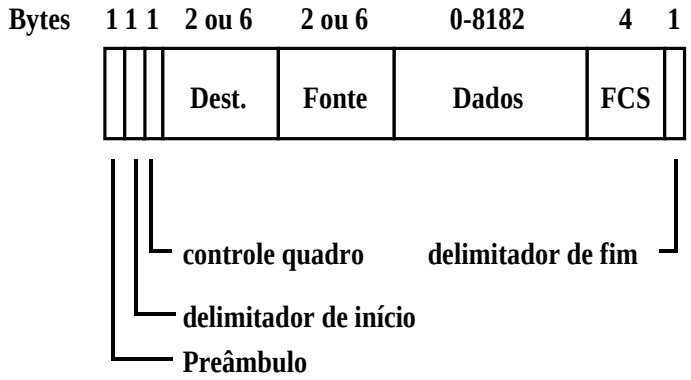


Figura 2.22 - Formato do quadro IEEE 802.4

O processo de evolução do anel lógico dá-se em função do estado de cada estação fazendo parte deste. A cada instante, tem-se uma determinada quantidade de estações ativas e outras que, por alguma razão devem encerrar as suas atividades. O protocolo de acesso ao meio oferece instruções permitindo a atualização do anel lógico de maneira ordenada.

Uma vez que o anel lógico está constituído com as estações ativas num dado instante, cada estação compondo o anel deve manter os endereços das suas estações vizinhas nos dois sentidos considerados (sucessora e predecessora). Periodicamente, a estação que detêm a ficha vai consultar as estações inativas para verificar se alguma delas querem fazer parte do anel lógico. Isto é feito pelo envio de um quadro do tipo "Procura Sucessor". Este quadro vai indicar o endereço da estação que emite o quadro e o da estação seguinte a esta no anel lógico. Apenas as estações cujos endereços estiverem entre os dois endereços indicados poderão candidatar-se à participação no anel lógico. Isto permite que seja respeitada a ordem de composição do anel. Se nenhuma estação apresenta interesse, a estação proprietária da ficha retoma a evolução normal do anel. Se apenas uma estação apresenta-se como candidata, ela passa a compor o anel lógico e torna-se a próxima destinatária da ficha. Caso duas ou mais

estações estejam se candidatando, ocorrerá colisão dos seus quadros (pelas mesmas razões da norma IEEE 802.3), o que requer que a estação proprietária da ficha resolva o conflito. Neste caso, ela emite um quadro "Resolução de Conflito" destinada a inicializar o processo de resolução de contenção. Ao receber um quadro de resolução de conflito, todas as estações que tinham se candidatado a entrar no anel lógico esperam por um tempo aleatório entre 0 e 3 time slots, após o que verificam se o meio está livre. Se o meio estiver ocupado após o término do tempo de espera escolhido, as estações desistem e aguardam uma nova oportunidade para entrar no anel lógico. Se houver nova colisão, o processo é repetido até que somente uma estação responda a busca de sucessor.

Já o abandono do anel é mais simples. Se uma estação X, situada entre duas estações A e B quer abandonar o anel, ela envia à estação A um quadro indicando que sua vizinha (seguinte) a partir de agora será a estação B.

A norma IEEE 802.4 especifica ainda quatro tipos diferentes de camadas físicas:

- Rede com canal único e modulação FSK (Frequency Shift Keying) fase contínua, com topologia em barra bidirecional, taxa de transmissão de 1Mbps;
- Rede com canal único e modulação FSK fase coerente, topologia em barra bidirecional, taxas de transmissão de 5Mbps ou 10Mbps;
- Rede em banda larga (vários canais modulados em freqüência sobre o mesmo meio), topologia em barra bidirecional com headend (central repetidora com conversor de freqüências do canal de recepção para o canal de envio), taxas de transmissão de 1Mbps, 5Mbps ou 10Mbps;
- Rede utilizando fibra ótica, topologia lógica em barra (mas fisicamente em estrela com um Hub como elemento central), requer um par de fibras para cada estação (uma para receber e outra para transmitir), taxas de transmissão de 5Mbps, 10Mbps ou 20Mbps.

A norma IEEE 802.5 (anel com ficha - "Token Ring")

Esta norma foi definida para privilegiar as redes em anel existentes. Um ponto importante no que diz respeito à concepção das redes em anel é a ocupação física de um bit no suporte de transmissão. Caso o débito normal de um anel seja de R Mbps, isto significa que um bit é transmitido a cada 1/R µs. Se a velocidade de propagação do sinal é de 200 m/µs, cada bit vai ocupar 200/R metros no suporte de transmissão. Isto significa que, para um anel operando a 1Mbps e utilizando 1000 metros de suporte físico, este só poderá suportar 5 bits a cada instante.

Uma rede em anel consiste de um conjunto de ligações ponto-a-ponto, em modo unidirecional, como mostra a figura 2.23. Cada nó do anel é equipado de um acoplador que permite conectar duas extremidades do cabo. Cada bit chegando na entrada de um acoplador é

copiado numa memória de espera antes de ser retransmitido ao nó seguinte. Este bit pode ser inspecionado e, mesmo, ter o seu valor modificado antes da retransmissão.

Num anel com ficha, uma seqüência binária particular, denominada ficha, fica circulando em permanência quando não existe transmissão de quadro. Quando uma estação quer emitir um quadro, ela deve adquirir a posse da ficha e substituí-la pelo quadro a enviar. Uma vez que apenas uma ficha está circulando no anel, a emissão de um quadro é uma ação exclusiva a uma única estação do anel, o que significa que os conflitos são, assim, eliminados.

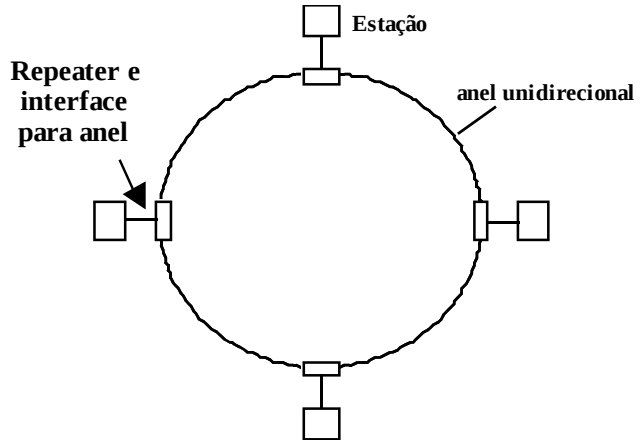


Figura 2.23 - Anel com ficha (Token-Ring)

A operação do protocolo IEEE 802.5 é relativamente simples. Quando não existem quadros sendo transferidos sobre a rede, uma ficha de 3 bytes fica circulando. Quando uma estação deseja transmitir um quadro, ela captura a ficha, posicionando um dos bits do segundo byte a 1, o que permite transformar os dois primeiros bytes da ficha numa nova seqüência, denominada, início de quadro. Em seguida, a estação completa a seqüência. Normalmente, o primeiro bit do quadro retorna à estação antes que todo o quadro tenha sido emitido. Por isto, a estação deve extrair do anel o que ela já emitiu para poder continuar a emissão do quadro. Cada estação só pode utilizar o direito de emissão por um período de tempo de 10 ms, a menos que algum outro valor tenha sido especificado no momento da ativação dos acopladores do anel. Se, após a transmissão de um quadro, restar ainda algum tempo para a emissão de outro quadro, a estação deverá fazê-lo. Caso contrário, ela encerra o seu processo de emissão e gera uma nova ficha que ficará circulando no anel.

Na figura 2.24 vemos o formato do quadro previsto na norma IEEE 802.5. Como mostra a figura, além dos campos similares aos quadros das normas anteriores, existe um campo adicional, denominado Status (Estado) do quadro. Este campo, de 1 byte, é composto de dois pares de bits denominados bits A e C. Quando um quadro é reconhecido por um acoplador de rede como sendo um quadro a ele destinado, ele copia o quadro inteiro e coloca os bits A deste campo a 1. Os possíveis valores dos bits A e C são apresentados abaixo, juntamente com os seus significados:

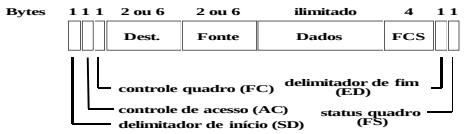


Figura 2.24 - Formato do quadro IEEE 802.5

- A = 0 e C = 0: o destinatário está inativo;
- A = 1 e C = 0: o destinatário está ativo mas o quadro não foi copiado;
- A = 1 e C = 1: o destinatário está ativo e o quadro foi copiado.

Este método permite implementar um reconhecimento automático do quadro. Os bits são duplicados para prevenção da ocorrência de erros de transmissão (estes bits não são controlados pela técnica de redundância cíclica).

A norma IEEE 802.5 prevê a nível da camada física o uso de segmentos com par trançado blindado operando a 4 ou 16Mbps com até 250 repetidores no anel ou ainda o par trançado comum com 4Mbps e também 250 repetidores. Os bits são codificados com a técnica Manchester diferencial.

A norma IEEE 802.11 - Wireless Networks (redes sem fio)

Nas redes sem fio, os pacotes são transmitidos através de canais de freqüência de rádio ou infravermelho, como ilustrado na figura 2.25. As redes sem fio são uma boa alternativa para aplicações onde é difícil instalar cabos metálicos ou de fibra ótica. Seu emprego é especialmente importante para comunicações entre computadores portáteis em um ambiente de rede local móvel ou onde a confiabilidade do meio de transmissão é requisito indispensável (por exemplo, onde o rompimento de um cabo pode paralisar todo o sistema).

Este tipo de rede vem ganhando certa aceitação em aplicações de chão de fábrica, devido a flexibilidade de instalação e operação, pois não há necessidade de projeto de layout, nem canaletes para cabos de rede, etc. Equipamentos móveis inteligentes do chão de fábrica podem ser configurados como estações de rede sem fio. Exemplos deste tipo de equipamento são os AGVs (Automatic Guided Vehicles), Robôs Autônomos e Sensores Inteligentes.

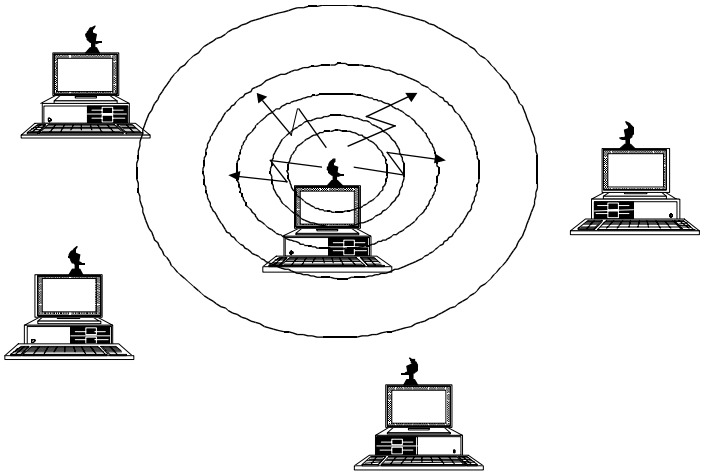
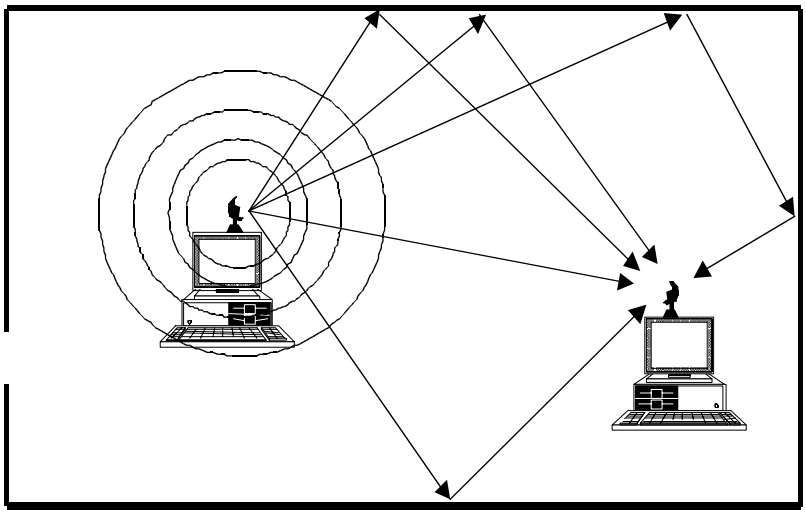


Figura 2.25 - Redes sem fio

Tais redes operam em bandas de freqüência denominadas ISM (Industrial, Scientific and Medical), que podem ser utilizadas sem que seja necessária uma licença. A norma IEEE 802.11 especifica como opções as bandas 902 até 928 MHz, 2.4 até 2.48 GHz e 5.75 até 5.85 GHz. O sinal emitido por uma estação operando nestas freqüências cobre uma área de 500 m^2 com uma potência de apenas 100mW. Áreas maiores podem ser cobertas decompondo a rede como um todo em várias subredes, responsáveis pela comunicação em uma área ou célula chamada BSA (Basic Service Area). Como a potência do sinal de rádio decai com o quadrado da distância do emissor, pode-se reutilizar a mesma freqüência de transmissão para estações em BSAs diferentes, desde que estas estejam suficientemente distantes entre si. Para permitir a construção de redes cobrindo áreas maiores que uma célula, as BSAs são interligadas por um sistema de distribuição, que consiste de uma rede usando meio físico convencional (cabo coaxial, par trançado ou fibra ótica).

Um problema típico das redes baseadas em sinal de rádio operando nas bandas de freqüência mencionadas é o chamado *desvanecimento de Rayleigh*, que pode ser explicado como segue. Parte das ondas de rádio são refletidas quando encontram objetos sólidos. Em



decorrência desta reflexão, várias cópias de uma mensagem de rádio podem estar em propagação no meio e chegar a estação receptora em instantes de tempo diferentes. Quando as várias cópias do sinal chegam ao receptor após percorrerem distancias diferentes, elas se somam aleatoriamente, podendo resultar em um sinal muito enfraquecido ou mesmo nulo, como ilustrado na figura 2.26 (se a diferença no comprimento dos caminhos for um múltiplo do comprimento de onda da portadora do sinal, os vários componentes podem cancelar-se mutuamente). O resultado disto é que, no mesmo ambiente, podemos ter ótima recepção em alguns locais é péssima recepção em outros locais a apenas poucos metros de distância. Desta forma, a qualidade da recepção varia muito a medida que uma estação se move no ambiente.

Figura 2.26 - Reflexão das ondas de rádio em um ambiente fechado

Como várias estações compartilham o mesmo meio, caracterizando uma rede de difusão, é necessário utilizar um método de acesso que discipline este compartilhamento. Uma primeira abordagem seria utilizar CSMA, isto é, cada estação escuta o meio e, se estiver livre, envia seu quadro. Isto, no entanto, não é tão simples em redes sem fio em decorrência do alcance do sinal de rádio, como veremos a seguir.

Suponha a situação ilustrada na figura 2.27, onde quatro estações sem fio estão representadas. Suponha que um sinal oriundo de A pode alcançar B, mas não alcança C nem D. Um sinal oriundo de C alcança B e D, mas não A.

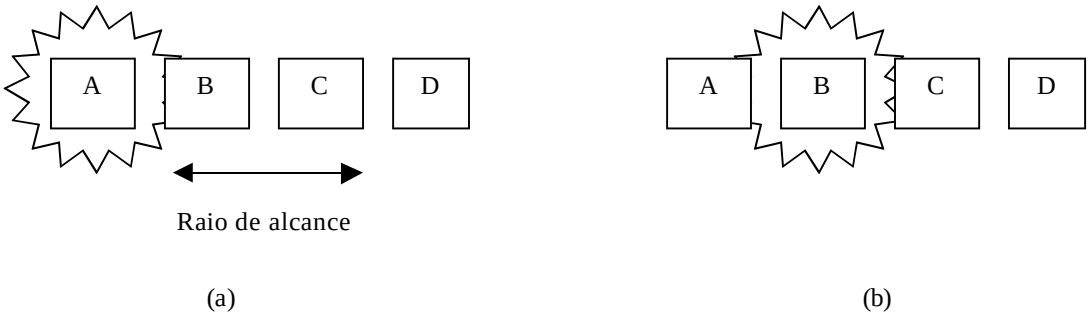


Figura 2.27 - (a) estação A transmitindo; (b) estação B transmitindo

Suponha agora que A está enviando dados para B. Se C escutar o meio, não irá detectar que A esta enviando, pois A esta fora do seu alcance. Logo, C pode tentar enviar um quadro para B ou D, mas como B está no alcance de C, o quadro enviado por A irá colidir com o quadro enviado por C a nível de B. O fato de uma estação não poder detectar que o meio não está livre porque o concorrente está fora de alcance é chamado de "problema da estação escondida" (hidden station problem). Por outro lado, se B estiver transmitindo um quadro para A, C irá detectar a transmissão (pois B está no seu alcance) e falsamente concluir que não pode transmitir um quadro para D neste momento. No entanto, como os receptores de A e D

não estão na área de interferência uma da outra, nada impede que C envie dados para D enquanto B envia para A. Esta situação é conhecida como o "problema da estação exposta" (exposed station problem). Em resumo, o que realmente interessa a uma estação pretendendo enviar um quadro em redes sem fio é saber se há ou não atividade na área do receptor.

Para resolver os problemas acima, a norma IEEE 802.11 utiliza em sua subcamada MAC (chamada DFWMAC, para Distributed Foudation Wireless MAC) um protocolo de acesso ao meio conhecido como MACA (*Multiple Access with Collision Avoidance*). A idéia básica por trás deste protocolo é fazer com que o emissor de um quadro estimule o receptor a emitir um quadro pequeno que possa ser detectado pelos seus vizinhos antes de mandar os dados em si.

Suponhamos agora que B quer enviar um quadro para C. Neste método de acesso, a estação B envia para C primeiro um quadro especial denominado RTS (Request To Send), contendo o tamanho do quadro de dados que deseja enviar a seguir. C deve responder com outro quadro especial chamado CTS (Clear To Send), contendo a mesma informação de tamanho. B inicia a transmissão quando recebe o quadro CTS de C, conforme ilustrado na figura 2.28. Qualquer estação que captar o quadro RTS estará forçosamente próxima a B e deve se manter em silencio por tempo suficiente para que B receba o CTS. Qualquer estação que captar o CTS estará forçosamente próxima a C e deve também se manter em silencio por tempo suficiente para que C receba o quadro de dados que B vai enviar a seguir, cujo tamanho pode ser avaliado examinando o quadro CTS. Vejamos como se comportam as demais estações: A escuta o RTS de B mas não o CTS de C, de modo que, desde que não queira mandar dados para B, A pode enviar seus quadros a qualquer outra estação em seu raio de alcance; Por outro lado, D escuta o CTS de C mas não o RTS de B, o que indica que está próxima a uma estação que vai receber um quadro de dados logo a seguir e portanto deve se manter em silencio até que este seja recebido.

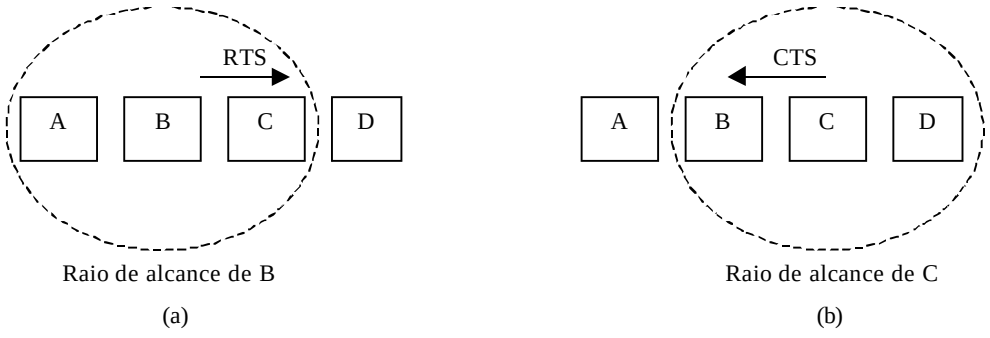


Figura 2.28 - O protocolo MACA

Apesar destas precauções, colisões ainda podem ocorrer. Por exemplo, A e C podem enviar quadros RTS para B ao mesmo tempo. Estes irão colidir e ser perdidos. No caso de colisão, o emissor do RTS espera um certo tempo pelo CTS e, se não receber nada, tenta novamente mais tarde.

Redes Submarinas:

•Comunicação subaquática tradicionalmente limitada a aplicações militares (submarinos, torpedos teleguiados, sonares).

•Primeiro sistema de comunicação UWA (Under-Water Acoustic): telefone criado em 1945 para comunicação com submarinos (águas rasas, modulação FSK de 8 a 11 khz).

•Recentemente surgiram várias aplicações civis:

–Exploração submarina para fins científicos;

–Soldagem e reparação de cascos de navios e dutos por robôs submarinos;

–Monitoração de poluição;

–Veículos submarinos não tripulados (AUV = Autonomous Underwater Vehicles);

–Sensores e atuadores submarinos (sismógrafos, válvulas, etc.);

–Comunicação entre mergulhadores;

–Montagem/manutenção/operação de plataformas de exploração/produção de petróleo.

•Geração de sinais:

–Sinais de rádio: para boa propagação na água, requerem ondas de baixíssima freqüência (30 a 300 Hz) => antenas grandes e transmissores de alta potencia.

–Sinais óticos: principal problema não é atenuação, mas dispersão.

–Sinais acústicos: melhor solução, podem se propagar na água por milhares de Km.

•Requisitos para tipos de dados mais usuais:

–Sinais de controle (comando de válvulas, solicitação de status, comandos de navegação para AUV, etc): requerem cerca de 1Kbps;

–Dados telemetria (hidrofones, sismógrafos, sonares, etc): requerem cerca de 10Kbps;

–Vídeo: requer de 10Kbps a 500Kbps para boa taxa atualização.

•Problemas tecnológicos a superar:

–Perda de transmissão: espalhamento de energia e absorção de som (proporcional ao quadrado da distância);

–Ruído acústico: pior em águas rasas, portos, etc.

–Reverberação: propagação de sinal por múltiplos caminhos causada por reflexão em obstáculos (desvanecimento de Rayleigh);

–Variações espaciais e temporais do meio (temperatura/densidade água, obstáculos móveis, etc.): problema pior se estações móveis.

•Considerações de projeto de sistemas UWA:

–Importante eliminar reverberação (muito pior que rádio).

–Uso de dispositivos direcionados: problemático se estações móveis;

–Técnicas FSK com tempo de espera entre pulsos de mesma freqüência (espera ecos desaparecerem);

–Técnicas Spread-Spectrum;

–Uso de equalizadores.

•Tipos de sistemas UWA em uso:

–<u>Longo alcance</u>: 20Km até 2.000Km, modulação FSK de 200Hz até 10KHz, taxas de transmissão baixas (típico: 1 bps);

–<u>Médio alcance</u>: 1Km até 20Km, uso em águas rasas, modulação FSK de 10KHz até 100KHz, 5Kbps;

–<u>Curto alcance</u>: até cerca de 60m, uso para robôs de manutenção e mergulhadores em águas rasas, modulação FSK de 1MHz, taxa de 500Kbps.

•Pesquisas atuais:

–Uso de PSK e QAM (Quadrature Amplitude Modulation) em lugar de FSK;

–Testes com sinais capazes de se propagar por todo o planeta (testado sinal gerado Austrália e lido na Califórnia/USA);

–Desenvolvimento de ALAN (Acoustic LAN): tendência de usar protocolos MACA e MACAW (IEEE 802.11), multiplexação de canais por TDM ou CDMA+Spread Spectrum.

3.2.3.3. PROJETO MAP

O projeto MAP ("Manufacturing Automation Protocol") foi uma iniciativa da General Motors, iniciada em 1980, com a finalidade de definir dentro do modelo OSI um ambiente de comunicação voltado para a automação da manufatura. MAP define mecanismos de comunicação entre equipamentos de chão de fábrica, tais como Robôs, CNCs, CLPs, terminais de coleta de dados, Computadores, etc.

Esta proposta encontra boa aceitação a nível mundial por parte de usuários e fornecedores de bens de automação, mas ainda esta sendo questionada pelos altos custos de cada estação.

Para aplicações voltadas ao controle da manufatura com tempos críticos foi definida a versão MAP/EPA (Enhanced Performance Architecture) para conseguir tempo de resposta menores. A Versão MAP/EPA apresenta para algumas estações da rede a arquitetura MAP completa (7 camadas) e uma arquitetura simplificada com as camadas 1,2 e 7 do modelo OSI como caminho alternativo para satisfazer os requisitos de tempo. A versão mais simplificada é conhecida como MINI-MAP e implementa somente as camadas 1, 2 e 7 do modelo OSI. O projeto MAP, devido a sua importância no contexto de redes industriais, será discutido mais detalhadamente em um capítulo a parte mais adiante.

3.2.3.4. PROJETO TOP

Com objetivos semelhantes ao MAP, foi desenvolvido pela BOEING a partir de 1983 o projeto TOP ("Technical and Office Protocol"), voltado à redes para automação de áreas técnicas e administrativas. Também é baseado no modelo OSI de 7 camadas e tem como finalidade fornecer aos usuários os seguintes serviços: correio eletrônico, processamento de textos, acesso a base de dados, transferência de arquivos, CAD/CAM distribuído, troca de documentos, transações bancárias, entre outros.

A partir de 1986 os projetos MAP e TOP passaram a ser coordenados conjuntamente, resultando no projeto MAP/TOP.

3.2.3.5. PROJETO FIELDBUS

O Fieldbus é uma solução de comunicação para os níveis hierárquicos mais baixos dentro da hierarquia fabril, interconectando os dispositivos primários de automação instalados na área de campo (Sensores, atuadores, chaves, etc) e os dispositivos de controle de nível imediatamente superior (CLPs, CNCs, etc).

Ainda estão sendo definidos os padrões para o Fieldbus. Os principais grupos envolvidos nos trabalhos de padronização são:

- Avaliadores: IEC, ISA, EUREKA, NEMA
- Proponentes: PROFIBUS, FIP, ISA-SP50

O Fieldbus e suas principais propostas de padronização será discutido em detalhes em um capítulo específico mais adiante.

3.3. O PROJETO MAP (Manufacturing Automation Protocol)

3.3.1. Introdução

O projeto MAP tem como mérito a apresentação de uma proposta concreta para a comunicação no ambiente de fábrica, estabelecendo as condições necessárias para a integração dos componentes de automação em um ambiente integrado segundo a filosofia CIM (Computer Integrated Manufacturing). Ao contrário do que se propunha inicialmente, hoje está claro que não existe uma solução única de rede que atenda aos requisitos de todos os níveis de comunicação em uma fábrica.

O projeto MAP nasceu no início dos anos 80 por iniciativa da GM (General Motors). Na época, apenas 15% dos equipamentos programáveis de suas fábricas eram capazes de se comunicar entre si.

Além disso, os custos de comunicação eram muito elevados, avaliados em 50% do custo total da automação, isto devido às conexões especiais necessárias entre cada equipamento. Ainda, cada nova instalação ou expansão no sistema existente estava associada a uma despesa não desprezível.

Considerando que, na época, estava previsto que a quantidade de equipamentos programáveis deveria sofrer uma expansão de 400 a 500% num prazo de 5 anos, o problema de comunicação tornou-se, efetivamente, uma prioridade a nível da empresa.

Diante do grave problema, a decisão deveria ser tomada no sentido de definir a solução, cujas opções eram as seguintes:

- continuar a produção utilizando máquinas programáveis isoladas (stand-alone) de uma variedade de fabricantes e solucionar o problema da maneira como vinha sendo feito;
- basear os processos de produção na aquisição de equipamentos de um único fabricante;
- desenvolver uma proposta padronizada que permitisse interconectar todos os equipamentos da planta.

Dadas as perspectivas de evolução e o grande desenvolvimento dos equipamentos de automação, a primeira proposta era, naturalmente, inviável. Com relação à segunda proposta, era (e continua sendo) impossível encontrar um único fabricante capaz de fornecer todos os equipamentos necessários (robôs, máquinas de comando numérico, CLPs, sensores, etc.) ao processo de fabricação.

A solução viria, então, pela terceira opção, que foi o ponto de partida para o projeto MAP, através da criação de uma força tarefa reunindo profissionais das diversas divisões da GM, cujo objetivo inicial era investigar a possibilidade de utilização do modelo de referência OSI como base para a proposta padronizada da empresa.

Um ano mais tarde, em 1981, a GM uniu-se a outras empresas -Digital Equipment Corporation (DEC), Hewlett-Packard (HP) e IBM -definindo a solução do problema baseada na utilização de uma arquitetura de comunicação para rede local baseada no modelo a sete camadas do OSI. Uma primeira preocupação deste grupo de trabalho foi a seleção de alguns dos padrões de protocolo definidos para o modelo OSI que pudessem ser adotados na arquitetura MAP.

A partir dai o projeto foi ganhando corpo e adesões por parte de outras empresas, tornando-se uma realidade nos anos 90 e dando origem a outras propostas de arquiteturas de comunicação orientadas a outros níveis das atividades da empresa.

3.3.2. A arquitetura MAP

Uma vez adotado o modelo OSI como referência para a arquitetura de comunicação, o problema era selecionar as propostas a serem implementadas a nível de cada camada. Para as camadas 1 e 2, foram selecionados, respectivamente, as normas IEEE 802.4 (barramento com ficha) e IEEE 802.2 (LLC).

Do ponto de vista da camada Física, foi escolhido o suporte de comunicação em banda larga (broadband). A escolha foi baseada nas razões seguintes:

- possibilidade de definição de vários canais de comunicação sobre um mesmo suporte de comunicação, o que permitiria a coexistência de várias redes, minimizando as modificações de cablagem durante a transição para MAP;
- permitiria a troca de outros sinais, como voz e imagem para determinadas aplicações, tais como a supervisão, o circuito fechado de TV, a teleconferência, etc;
- broadband é parte da norma IEEE 802.4 e estava sob estudos suportar o padrão IEEE 802.3 (CSMA/CD);
- a GM já possuía muitas instalações operando em broadband.

As razões que conduziram à escolha do barramento com ficha foram as seguintes:

- inicialmente, era o único protocolo suportado em broadband;
- muitos equipamentos programáveis já eram providos com o protocolo de enlace suportado por broadband e IEEE 802;
- a possibilidade de implementar um esquema de prioridades de mensagens;
- em caso de falhas físicas, mensagens de alta prioridade poderiam ser enviadas num tempo limitado.

Apesar das razões expostas acima para a escolha do barramento com ficha, esta foi uma escolha relativamente debatida, principalmente porque a arquitetura MAP é a única a

adotá-la e os circuitos integrados implementando IEEE 802.4 são utilizados exclusivamente para esta arquitetura. Além disso, outras propostas tinham sido adotadas pelos grandes fabricantes: Ethernet (IEEE 802.3) no caso da DEC e IEEE 802.5 no caso da IBM.

A nível da camada de Enlace, embora as funções associadas sejam principalmente a detecção e recuperação de erros, optou-se por um protocolo que não implementasse estes serviços, o LLC tipo 1, deixando estas funções a cargo dos níveis superiores (mais particularmente, o nível Transporte).

O serviço de Rede é sem conexão, cada mensagem sendo roteada individualmente através da rede. A norma ISO 8348, adotada a este nível, permite definir um conjunto de regras de endereçamento através da rede. O protocolo de roteamento utilizado aqui foi definido pelo projeto MAP e é atualmente normalizado na ISO sob o número 9542.

A nível do Transporte, foi adotada a classe 4 do protocolo de Transporte da ISO (TP4, ISO 8072/73), orientado à conexão, com controle de erros. O serviço de Transporte oferece, então, um canal de comunicação confiável, sem perdas, erros, nem duplicação de mensagens. TP4 assegura ainda as funções de fragmentação e montagem de mensagens, o que permite que as mensagens trocadas a este nível sejam de qualquer dimensão.

A norma ISO 8326/27 foi adotada para a camada de Sessão, assegurando as funções de comunicação full-duplex e de ressincronização.

Na camada de Apresentação, os problemas de representação de dados são resolvidos com a adoção da sintaxe abstrata ASN.1, que serve de linguagem comum às diferentes formas de representação dos dados, características de cada equipamento.

Dentre as funções oferecidas aos processos de aplicação, foram definidas, na camada de Aplicação, as seguintes normas:

- MMS, para a troca de mensagens entre equipamentos de produção (que será visto em detalhes mais a frente);
- FTAM, para o acesso e a transferência de arquivos;
- ROS, para a gestão de nomes (diretório);
- funções de gerenciamento de rede, para a gestão dos recursos, medição de desempenho e modificação dos parâmetros da rede.

A figura 3.1 apresenta as escolhas efetuadas a nível do projeto MAP, incluindo as versões EPA e Mini-MAP. Como a partir da versão 3.0 ocorreu uma unificação dos projetos MAP e TOP, a figura apresenta também as normas adotadas para a arquitetura TOP.

<table>
<tr><th>Especificação / Camada</th><th>TOP</th><th>MAP</th><th>MAP/EPA</th><th>Mini-MAP</th></tr>
<tr><td>Aplicação</td><td>ACSE
FTAM
VTS</td><td colspan="2">MMS
FTAM
ROS</td><td></td></tr>
<tr><td>Apresentação</td><td colspan="3">ISO 8822 - ASN.1</td><td rowspan="4">VAZIO</td></tr>
<tr><td>Sessão</td><td colspan="3">ISO 8326 e 8327</td></tr>
<tr><td>Transporte</td><td colspan="3">ISO 8072 e 8073 - Classe 4</td></tr>
<tr><td>Rede</td><td colspan="3">ISO 8348 - Sem Conexão</td></tr>
<tr><td>Enlace</td><td>LLC 802.2 tipo 1
MAC 802.3 CSMA/CD</td><td>LLC 802.2 tipo 1
MAC 802.4 Token bus</td><td colspan="2">LLC 802.2 tipos 1 e 3
MAC 802.4</td></tr>
<tr><td>Física</td><td>Banda Base
10 Mbps</td><td>Banda larga
10 Mbps</td><td colspan="2">Banda base
5 Mbps</td></tr>
</table>

Figura 3.1 - Especificação MAP/TOP 3.0

3.3.3. A arquitetura MAP-EPA

Dadas as necessidades específicas de cada nível hierárquico de uma empresa, verificou-se que a proposta MAP original não permitia cobrir todos os níveis considerados, sendo mais adequada aos níveis superiores. Uma razão principal disto é que, apesar da excelente qualidade dos serviços oferecidos, a arquitetura a sete camadas oferece um overhead que passa a ser indesejável nos níveis mais baixos das atividades de uma empresa.

Uma primeira solução a este problema foi a definição de uma versão simplificada da arquitetura MAP, denominada MAP-EPA (Enhanced Performance Architecture). A figura 3.2 apresenta a proposta MAP-EPA.

Esta proposta foi baseada na definição de duas pilhas de protocolos, a pilha normal Full-MAP e a pilha MAP-EPA, desprovida das camadas de Rede, Transporte, Sessão e Apresentação. Do ponto de vista das camadas baixas, o protocolo IEEE 802.4 continuava sendo adotado, porém sobre um suporte de transmissão em banda de base (baseband) a 5 Mbit/s.

Nesta arquitetura, um processo de aplicação tem a opção de enviar seus dados através da pilha normal ou, em casos onde o requisito seja um tempo de resposta rápida, pela pilha MAP-EPA. Evidentemente, o fato das camadas 3 a 6 estarem ausentes acarreta a perda dos serviços oferecidos por estas.

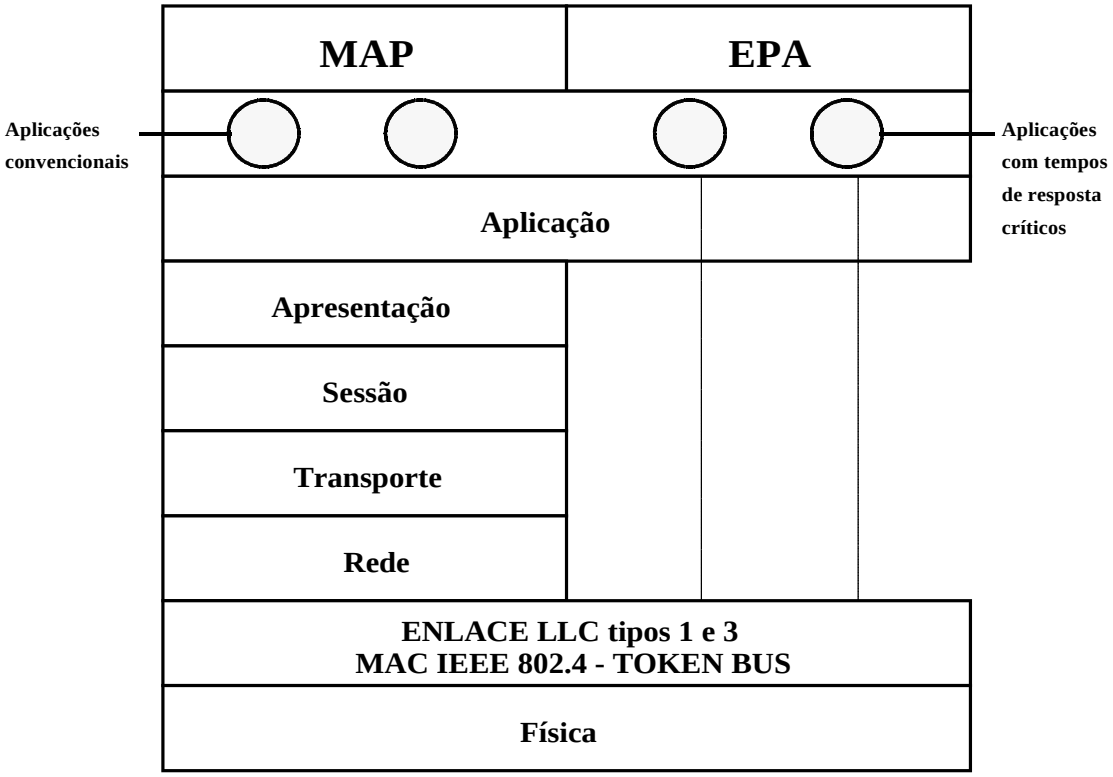


Figura 3.2 - Arquitetura MAP-EPA

3.3.4. A arquitetura Mini-MAP

Uma terceira opção relacionada com a norma MAP foi a arquitetura Mini-MAP, baseada igualmente na supressão das camadas 3 a 6 para eliminar o overhead dos protocolos daquelas camadas. A arquitetura Mini-MAP é composta unicamente do segmento simplificado de MAP-EPA, e foi assim definida para evitar o alto custo das pilhas de protocolos paralelas de MAP-EPA (figura 3.3). Esta nova proposta era dedicada aos níveis mais baixos, permitindo a comunicação em aplicações mais simples como, por exemplo, entre sensores inteligentes.

O fato de não possuir a camada de Transporte fez introduzir um protocolo de Enlace mais sofisticado que o da proposta MAP, o LLC tipo 3, datagrama com reconhecimento.

3.3.5. Os serviços de mensagem industrial (MMS)

MMS (Manufacturing Message Services) foi normalizado na ISO como sendo o conjunto de serviços de comunicação oferecido às aplicações industriais, particularmente para viabilizar, dentro do ambiente OSI, as interações entre equipamentos de produção programáveis.

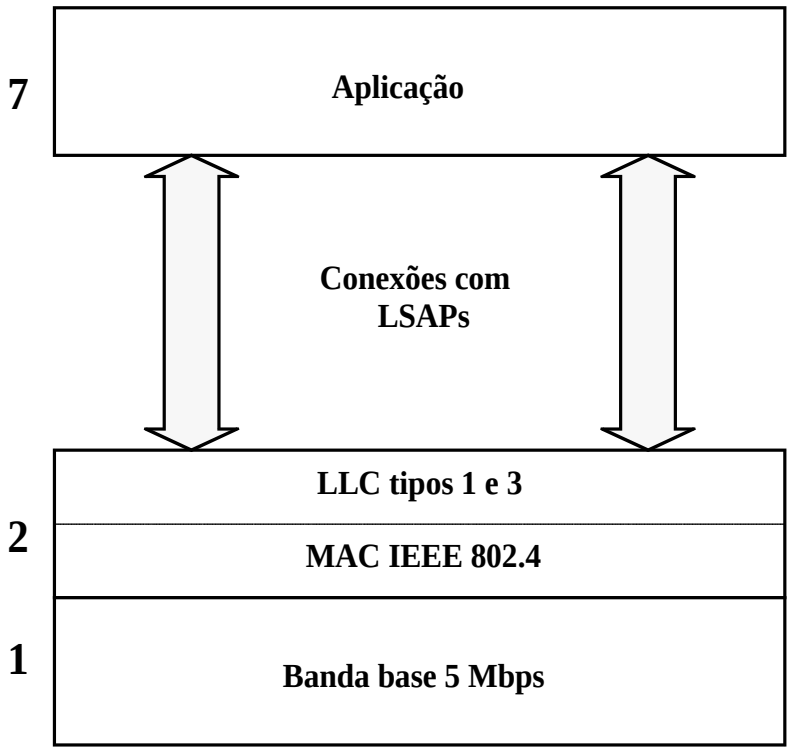


Figura 3.3 - Arquitetura Mini-MAP

MMS é o resultado dos trabalhos realizados no contexto do projeto MAP, para a definição de um conjunto de serviços de comunicação orientados às aplicações industriais.

A primeira proposta de MMS/RS-511 foi apresentada em junho de 1985, na forma de um documento organizado em duas partes:

- Manufacturing Message Services: Definição dos Serviços;
- Manufacturing Message Specification: Especificação do Protocolo.

Atualmente, MMS tornou-se norma internacional, fazendo parte da camada de Aplicação da versão 3.0 de MAP, publicada em agosto de 1988. Os dois documentos mencionados acima apresentam, de forma geral, como os serviços e o protocolo podem ser aplicados no contexto da utilização de um equipamento de produção genérico, sem levar em conta as particularidades de uma classe de equipamentos específica. Para complementar a norma existente, outros documentos foram e estão sendo produzidos, denominados normas de acompanhamento ("Companion Standards"), cujo objetivo é levar em conta as especificidades de classes de equipamentos tais como os robôs, as máquinas de comando numérico, os sistemas de visão, os controladores lógicos programáveis e os sistemas de controle de processos.

O objetivo de MMS é oferecer serviços de comunicação que permitam a um sistema aberto (no sentido OSI) acessar os recursos existentes em outros sistemas abertos conectados à rede de comunicação. Eles permitem cobrir grande parte das necessidades de comunicação entre sistemas de produção, como, por exemplo, o carregamento remoto de programas, o controle remoto de um equipamento, a elaboração de relatórios de produção, etc.

Os programas escritos pelos programadores de aplicação vão acessar (direta ou indiretamente) as primitivas de serviço MMS, que vão manipular objetos virtuais representando os recursos reais disponíveis num equipamento de produção distante.

3.3.5.1. OS OBJETOS MMS

Os usuários dos serviços MMS são os processos de Aplicação (APs - Application Processes) executando num equipamento de produção ou num computador de supervisão. A comunicação entre dois APs através dos serviços MMS é realizada segundo um modelo Cliente-Servidor, onde o usuário Cliente é aquele que requisita operação sobre os recursos disponíveis num equipamento de produção distante, este sendo modelizado por um usuário Servidor.

O objeto de base definido em MMS é o *Dispositivo Virtual de Produção* ou VMD (Virtual Manufacturing Device), que representa, no contexto dos serviços MMS, um equipamento real de produção. Todo processo de aplicação modelizado por um Servidor MMS possui, no mínimo, um objeto VMD (figura 3.4). O principal componente do VMD é a Função Executiva, responsável pela gestão de acesso aos diferentes recursos do equipamento considerado, tais como memória, processadores, portas de E/S, etc.

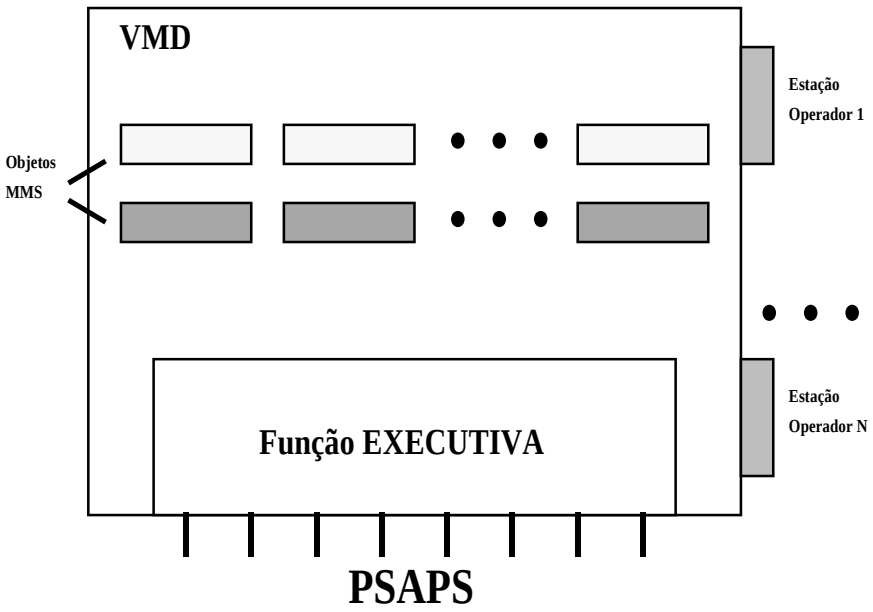


Figura 3.4 - Estrutura geral de um VMD

O VMD define uma classe de objetos denominados *domínios* (Domains), que permitem reagrupar os programas e os dados necessários à execução do equipamento considerado. Estes programas e dados podem ser definidos de maneira estática ou dinâmica por meio dos serviços MMS.

A execução de programas é gerenciada através de objetos denominados *Invocação de Programa* (Program Invocation), que podem, também, ser criados estática ou dinamicamente. Um operador humano pode se comunicar com um equipamento de produção, fazendo a entrada e saída de dados graças à definição de um objeto *Estação Operador*, sendo que um VMD pode gerenciar uma ou mais estações de operador.

A norma prevê, ainda, objetos permitindo gerenciar a sincronização de processos e o acesso concorrente a recursos, que são os objetos *Semáforos*; para a detecção e o tratamento de eventos, os objetos *Condição de Evento*, *Ação de Evento* e *Inscrição de Evento*; e para a produção de relatórios de produção, os objetos *Jornais*.

Foram definidos também objetos denominados *variáveis* (variables), que podem ser alocados dentro de um VMD. As variáveis podem se referir, por exemplo, a entradas e saídas de um CLP e podem ser lidas ou escritas remotamente.

A cada classe de objetos MMS é associada uma classe de serviços responsáveis da sua manipulação, sob demanda de um usuário Cliente remoto.

3.3.5.2. OS SERVIÇOS MMS

A norma MMS define 84 serviços, dos quais 3 não confirmados, organizados em 9 classes distintas. Os serviços de *Gestão de Contexto* são utilizados para o estabelecimento e a manutenção do diálogo entre usuários MMS, nenhum serviço podendo ser ativado sem que o contexto tenha sido estabelecido, suportado por uma associação. O serviço *Initiate* é o serviço definido para o estabelecimento da associação, este sendo mapeado diretamente sobre o serviço A_ASSOCIATE do ACSE. Esta classe de serviços reagrupa, ainda, serviços para a terminação negociada do contexto (*Conclude*), a liberação abrupta (*Abort*). Uma requisição de serviço pode ser anulada por um usuário MMS cliente, através da ativação do serviço *Cancel*. Ainda, o fornecedor de serviço MMS pode sinalizar a ocorrência de um erro de protocolo através do envio de uma primitiva de serviço (indicação) *Reject*, informando o tipo de erro detectado.

Os serviços de *Suporte de VMD* permitem a um usuário Cliente MMS obter informações de estado sobre um objeto VMD, assim como sobre os objetos gerenciados por este. As informações possíveis de serem obtidas são:

- Estado lógico do VMD (*Status*);
- Lista dos objetos do VMD (*GetNameList*);
- Características do equipamento de produção (*Identify*).

Finalmente, é ainda possível modificar os identificadores dos objetos de um VMD, isto através do serviço *Rename*.

Os serviços de *Gestão de Domínio* são utilizados para o carregamento e armazenamento à distância de programas e dados de um equipamento de produção. Estas operações são baseadas na realização de duas seqüências de serviços MMS, a primeira para o carregamento remoto, *DownLoadSequence*, a outra para a recuperação do conteúdo de domínios, *UpLoadSequence*.

Um usuário MMS ativa a seqüência DownLoad para pedir a criação e o carregamento de informações para um domínio. Esta seqüência é caracterizada por três etapas:

- inicialização da seqüência e criação de um domínio (*InitiateDownLoadSequence*);
- transferência das informações para o domínio (*DownLoadSegment*);
- terminação da seqüência (*TerminateDownLoadSequence*).

Os serviços de *Gestão de Programa* permitem a um usuário MMS manipular, de maneira remota, os programas num VMD, para a criação, eliminação e controle de execução. As operações podendo ser efetuadas sobre um objeto Invocação de Programa são as seguintes:

- criação e eliminação (*CreateProgramInvocation* e *DeleteProgramInvocation*);
- controle de execução de um programa (*Start*, *Stop*, *Resume*, *Reset* e *Kill*);
- obtenção de atributos (*GetProgramInvocationAttributes*).

Os serviços de *Acesso às Variáveis* permitem a um cliente MMS definir e acessar variáveis definidas num VMD de um servidor MMS. O acesso às variáveis reais de um equipamento é possível graças àdefinição de um conjunto de objetos variáveis MMS, manipulados pelos 14 serviços definidos nesta classe. Os serviços *Read* e *Write*, por exemplo, permitem acessar, respectivamente em leitura e escrita, uma variável. Existem ainda serviços permitindo definir os diferentes objetos associados às variáveis, tais como *DefineNamed-Variable, DefineScatteredAccess, DeleteVariableAccess, DefineNamedVariableList, Delete-NamedVariableList, DefineNamedTyp*e e *DeleteNamedType*.

Os serviços de *Gestão de Semáforos* permitem gerenciar o acesso concorrente aos recursos compartilháveis de um VMD. Os objetos associados a esta classe são os Semáforos, organizados em duas classes específicas:

- *Token Semaphores*, que gerenciam um número N de fichas de mesmo tipo;
- *Pool Semaphores*, que gerenciam uma lista de fichas etiquetadas, cada uma associada a um recurso particular do VMD.

Um usuário MMS pode requisitar o acesso a um recurso do VMD através do serviço *TakeControl*, indicando o semáforo responsável por aquele recurso. Um aspecto interessante desta classe de serviços é a existência de um modificador *AttachToSemaphore*, que permite condicionar a execução de um serviço MMS à posse de um semáforo.

Esta classe oferece, ainda, serviços para a liberação, a criação e a obtenção de informações de estado sobre os objetos semáforo.

Os serviços de *Gestão da Estação de Operador* implementam a comunicação, através de uma estação de operador, com um VMD. Eles permitem, particularmente, a entrada e a obtenção de dados relativos à execução do VMD considerado.

Os serviços de *Gestão de Eventos* oferecem facilidades para um Cliente MMS definir e tratar a ocorrência de eventos num VMD. O tratamento de eventos é associado à definição de três objetos:

- as *Condições de Evento*, que representam as pré-condições associadas a um dado eventos num VMD (modificação de uma variável, fim de execução de um programa, etc.);
- as *Ações de Evento*, que representam o que deve ser feito quando um evento é detectado (execução de um serviço MMS, por exemplo);
- as *Inscrições de Evento*, que permitem associar uma Condição de Evento a uma ou mais Ações de Evento.

Um usuário MMS é notificado da ocorrência de um evento através do serviço *EventNotify*, esta notificação podendo ser reconhecida pelo usuário através do serviço *AcknowledgeEventNotification*. A ocorrência de um evento pode servir a ativar a execução de um serviço MMS, isto sendo feito através do modificador *AttachToEvent*. Nesta classe, tem-se, ainda, serviços para a criação, destruição de objetos, obtenção de informações (estado, atributos, etc.) dos objetos e modificação de atributos dos objetos de evento.

Os serviços de *Gestão de Jornais* fornecem facilidades para o armazenamento e a recuperação, de maneira ordenada, a partir dos objetos Jornais, das informações e variáveis associadas a eventos, assim como de texto que pode servir como comentários ou explicações. Estes serviços são utilizados principalmente para a produção de relatórios sobre o funcionamento de um equipamento de produção.

Ainda, a norma MMS prevê uma classe de serviços para o tratamento de arquivos, particularmente para pequenos serviços de criação, eliminação, etc. No caso de aplicações que requisitem serviços mais sofisticados para o acesso e a transferência de arquivos, a entidade de Aplicação deverá, então, ser composta do elemento de serviço de Aplicação FTAM, que oferece estes serviços.

As tabelas a seguir apresentam uma síntese das diferentes classes de serviços do MMS.

Classe	Primitivas de Serviço	Comentários
Gestão de Semáforos	TakeControl RelinquishControl DefineSemaphore DeleteSemaphore ReportSemaphoreStatus ReportPoolSemaphoreStatus ReportSemaphoreEntryStatus	são encarregados da sincronização e do controle do acesso aos recursos de um VMD pelos processos de aplicação
Estação Operador	Input Output	controlam a entrada e saída de informações via estações de operador
Gestão de Eventos	DefineEventCondition DeleteEventCondition GetEventConditionAttribute ReportEventConditionStatus AlterEventConditionMonitoring TriggerEvent DefineEventAction DeleteEventAction GetEventActionAttributes ReportEventActionStatus DefineEventEnrollment DeleteEventEnrollment GetEventEnrollment ReportEventEnrollment AlterEventEnrollment EventNotification* AcknowledgeEventNotification GetAlarmSummary GetAlarmEnrollmentSummary AttachToEventModifier	permitem a definição e o tratamento de eventos via serviços MMS. A possibilidade de associar a execução de um serviço MMS àocorrência de um evento é um aspecto interessante, implementado pelo Modificador AttachToEvent
Gestão de Jornal	ReadJournal WriteJournal InitializeJournal ReportJournalStatus	permitem o salvamento de informações sobre a execução de um VMD, particularmente no que diz respeito à ocorrência de eventos e à afetação de variáveis.

* serviços não confirmados

Classe	Primitivas de Serviço	Comentários
Gestão de Contexto	Initiate Conclude Abort* Cancel Reject*	iniciação, liberação, abandono e rejeição de conexão com outro usuário MMS
Gestão de VMD	Status UnsolicitedStatus* GetNameList Identify Rename	oferece serviços de VMD, particularmente informações sobre os objetos
Gestão de Domínio	InitiateDownLoadSequence DownLoadSegment TerminateDownLoadSequence InitiateUpLoadSequence UpLoadSegment TerminateUpLoadSequence RequestDomainDownLoad RequestDomainUpLoad LoadDomainContent StoreDomainContent DeleteDomain GetDomainAttribute DomainFile	permitem transferir informações, tais como códigos e dados de programa, para serem carregados num domínio de forma dinâmica: as seqüências DownLoad e UpLoad são atividades que permitem gerenciar as transferências entre Cliente e Servidor
Gestão de Programas	CreateProgramInvocation DeleteProgramInvocation Start Stop Resume Reset Kill GetProgramInvocationAttributes	permitem que um usuário Cliente MMS gerencie a execução remota de programas num usuário Servidor
Acesso a Variáveis	Read Write InformationReport GetVariableAccessAttributes DeleteNamedVariable DefineScatteredAccessAttributes DeleteVariableAccess DefineNamedVariableList GetNamedVariableListAttributes DeleteNamedVariableList DefineNamedType GetNamedTypeAttributes DeleteNamedType	permitem a definição e o acesso às variáveis de um VMD e estabelecer a relação entre as variáveis de um VMD (objetos) e as variáveis real de um equipamento de produção

* serviços não confirmados

3.4. REDES FIELDBUS

3.4.1. Motivações e Requisitos do Fieldbus

A evolução dos microprocessadores conduziu a estruturação hierárquica fabril até o nível de simples componentes de automação, envolvendo elementos diretamente ligados ao processo a ser controlado ou supervisionado, tais como sensores, atuadores e controladores.

Através da aplicação de microprocessadores em sistemas sensórios é possível realizar um pré-processamento do sinal diretamente no local de medição, diminuindo assim a probabilidade de deturpação do sinal por perturbações eletromagnéticas, às quais sinais analógicos são em geral muito suscetíveis. Além disto, a unidade que requer os dados do sensor é aliviada destas funções. O mesmo principio é aplicado aos atuadores, que passam a poder interpretar e executar comandos complexos advindos da unidade central.

Uma vez que os sinais utilizados por tais componentes inteligentes já se encontram em forma digital, é conveniente definir sistemas de comunicação de dados também digitais para substituir as clássicas interfaces analógicas de corrente e tensão. As interfaces de corrente (4..20 mA) são bem mais difundidas em ambientes industriais, devido a menor suscetibilidade às interferências eletromagnéticas.

Inicialmente foram definidas e grandemente difundidas interfaces digitais para interligação *ponto-a-ponto*, tipo RS 232 C que, no entanto, devido à utilização de sinal referenciado ao terra (v24) se mostram mais adequadas à utilização em aplicações de escritório. Para reduzir a influência das perturbações foram desenvolvidas interfaces com sinal diferencial (v11), tipo RS 422 e RS 423. Estas interfaces oferecem no entanto a desvantagem de que não permitem o acoplamento simultâneo de vários elementos inteligentes entre si. Para suprir esta necessidade foram desenvolvidas interfaces *multiponto*, adequadas a interconexão dos elementos em redes tipo barramento (bus), como por exemplo a RS 485. Esta tendência é mostrada na figura 4.1.

Para uma rede aplicada à interligação de elementos simples a nível de chão de fábrica é utilizada a denominação genérica de "barramentos de campo", ou *Fieldbus*. O Fieldbus pode ser definido como uma linha de comunicação serial, digital, bidirecional (de acesso compartilhado) para a interligação dos dispositivos primários de automação (instrumentos de medição, atuação e controle final e outros pequenos dispositivos "inteligentes" com capacidade de processamento local) a um sistema integrado de automação e controle de processos.

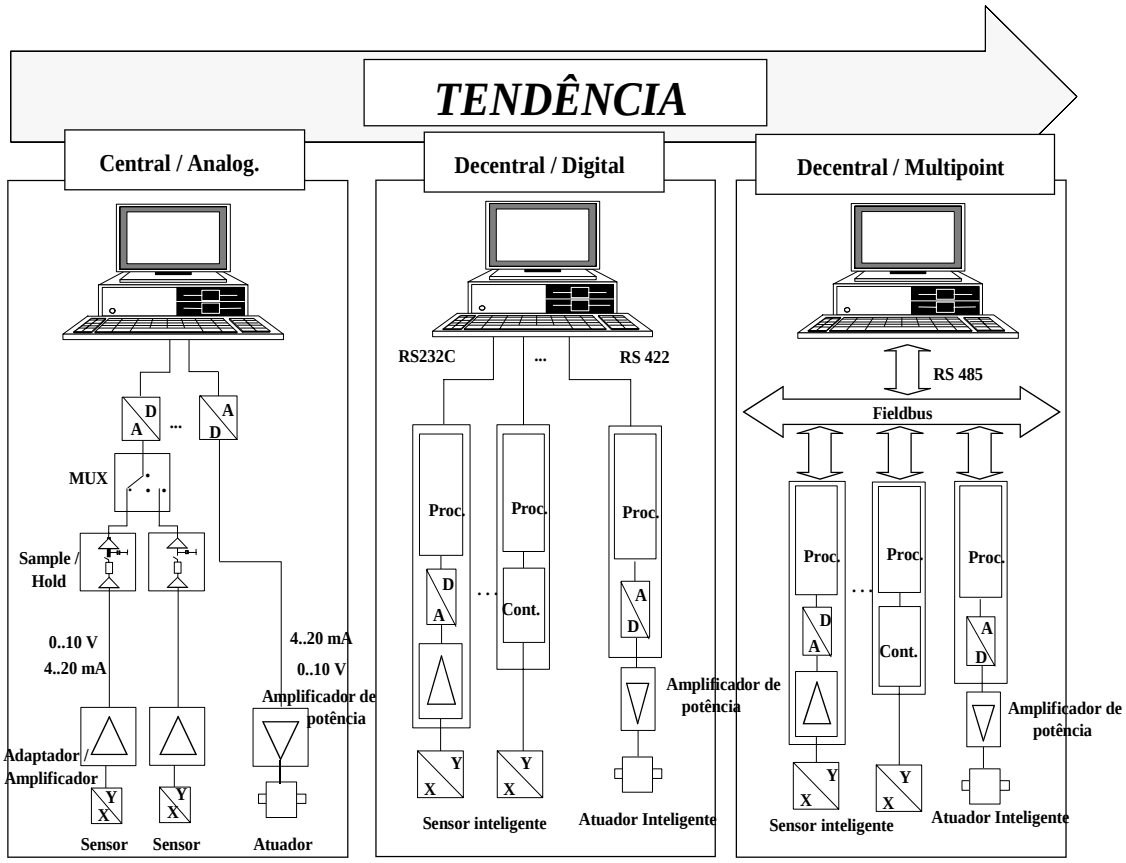


Figura 4.1 - Tendências na interligação de sensores / atuadores

As principais vantagens e benefícios da utilização do Fieldbus em relação às conexões convencionais podem ser resumidas como segue:

a) vantagens econômicas:

- drástica redução da cablagem pela utilização de um meio físico compartilhado;
- redução do número de canais de comunicação com o processo;
- redução do tempo e complexidade do projeto de layout;

b) vantagens técnico-operacionais:

- facilidade de instalação e manutenção, pela manipulação de um menor número de cabos e conexões;
- facilidade de detecção, localização e identificação de falhas, através de funções de monitoração automática;
- maior modularidade no projeto e instalação, aumentando a flexibilidade de expansão de funções e módulos;

c) vantagens sistêmicas:

- aumento da consistência e da confiabilidade da informação advinda dos instrumentos de campo, através da digitalização e pré-processamento;
- possibilidade de sincronização dos instantes de amostragem de Entrada/Saída;
- melhoria do desempenho global da aplicação pela descentralização do processamento;
- maior facilidade de interconexão entre níveis hierárquicos diferentes de automação;

d) vantagens mercadológicas:

- redução dos custos de sistemas através da aquisição seletiva de dispositivos compatíveis de diferentes fornecedores, eliminando a dependência de somente um fornecedor;
- desacoplamento do software de supervisão da dependência de um fornecedor específico do Hardware;

Inicialmente se supunha que uma única rede iria cobrir todas as necessidades de interconexão no chão de fábrica. Notadamente a proposta MAP, acrescida das versões de maior performance e menor custo MAP/EPA e MINI-MAP se propunham também a integrar elementos de automação ao nível hierárquico de componente. No entanto, logo pode ser verificado que estas propostas eram demasiado caras e muito lentas para as aplicações típicas a nível hierárquico de componente.

Enquanto os segmentos de banda base do MAP (MAP-EPA e Mini-MAP) permitem a realização de tempos de resposta de cerca de 100 ms, sistemas tipo Fieldbus se propõe a reduzir este tempo para a faixa de 1 a 10 ms, como requerido para o controle e supervisão de importantes grandezas envolvidas na automação, tais como velocidade, posição de eixos, torque, aceleração e força.

Um aspecto essencial na definição de protocolos de comunicação para o Fieldbus é a redução a um mínimo estritamente necessário das informações adicionais incluídas nos telegramas, de forma a permitir uma adequada performance em tempo real. Estas simplificações em relação a sistemas tipo MAP e TOP são indispensáveis quando se pretende interligar componentes simples tais como sensores e atuadores. Para sistemas tipo Fieldbus são definidas, por questões de eficiência, somente as camadas 1, 2 e 7 do modelo de referência OSI. As funções das camadas 3 até 6 que são indispensáveis para a comunicação são aqui absorvidas pelas camadas 2 ou 7.

O aspecto de custo também assume grande importância, uma vez que os dispositivos a serem interligados tem em geral custo inferior ao da própria interface MAP, como pode ser visto na tabela 4.1. São requeridos aqui nós a um custo da ordem de U$ 50 ou inferior.

Componente MAP	**Preço médio**	**Elemento Campo**	**Preço médio**
Cabo Coaxial	U$ 2,5 / m	CLP	U$ 3.000
Controlador	U$ 5.000	Controle Robô	$20.000
Demodulador	U$ 1.500	PC	U$ 2.000
Componente Ethernet / IBM	**Preço médio**	Sensor/Atuador	U$ 50 a 1000
Nó CSMA/CD	U$ 500 - 1500	I/O Binária	U$ 50 a 1000
Nó Token-Ring	U$ 750 - 1500		

Tabela 4.1 - Preços médios de componentes de rede

O Fieldbus deve atender igualmente aos requisitos impostos pelos sistemas discretos (Manufatura) e os sistemas contínuos de produção (Controle de Processos).

Podem-se considerar em principio três classes distintas de aplicação:

- sistemas "Stand-Alone", nos quais as transações ocorrem somente entre dispositivos ligados em um mesmo segmento de rede. Aqui teríamos, por exemplo, sensores e atuadores ligados a um CNC dentro de uma máquina.
- sistemas em cascata, nos quais dispositivos conectados a segmentos distintos podem trocar informações por meio de um "bridge". Situação típica é a encontrada em um SDCD (Sistema Distribuído de Controle Digital).
- sistemas hierárquicos, nos quais o segmento Fieldbus esta interligado via "gateway" a uma rede interligando dispositivos de um nível hierárquico superior da automação fabril. É o tipo de situação encontrado em uma estrutura CIM.

O fluxo de dados nestas aplicações pode se dar em dois sentidos:

- no sentido *vertical*, entre níveis hierárquicos diferentes, de modo a permitir a supervisão do sistema. Para esta forma de comunicação deverão ser oferecidos serviços de instalação, start-up, parametrização da aplicação, visualização de dados para supervisão, etc. Estes serviços não são críticos no tempo.
- no sentido *horizontal*, entre elementos do mesmo nível, compondo um sistema distribuído. Os serviços oferecidos para este tipo de comunicação se destinam basicamente à atualização e consumo de dados e a checagem de "status". A execução destes serviços é crítica no tempo.

Em função do tipo de aplicações que se propõe a atender, um conjunto de requisitos básicos são impostos ao Fieldbus:

- necessidade de elevado desempenho para atender as aplicações com requisitos de tempo críticos;
- método e meio de transmissão simples e de preço acessível. Os sistemas tipo Fieldbus utilizam transmissão tipo Baseband. Como meio de transmissão adota-se o par trançado;
- necessidade de consistência de dados;
- necessidade de serviços compatíveis com redes dos níveis hierárquicos superiores. Em geral, se procura uma compatibilidade com os serviços oferecidos pelo MMS;

Diversos fabricantes de sistemas de comunicação industriais tem desenvolvidos suas próprias soluções para o Fieldbus que, no entanto, não apresentam todos os requisitos necessários às diversas aplicações na área de chão de fábrica. Para suprir esta deficiência, vários esforços nacionais e internacionais tem sido feitos no sentido de definir uma norma universalmente aceita para o Fieldbus, conforme mostrado na figura 4.2.

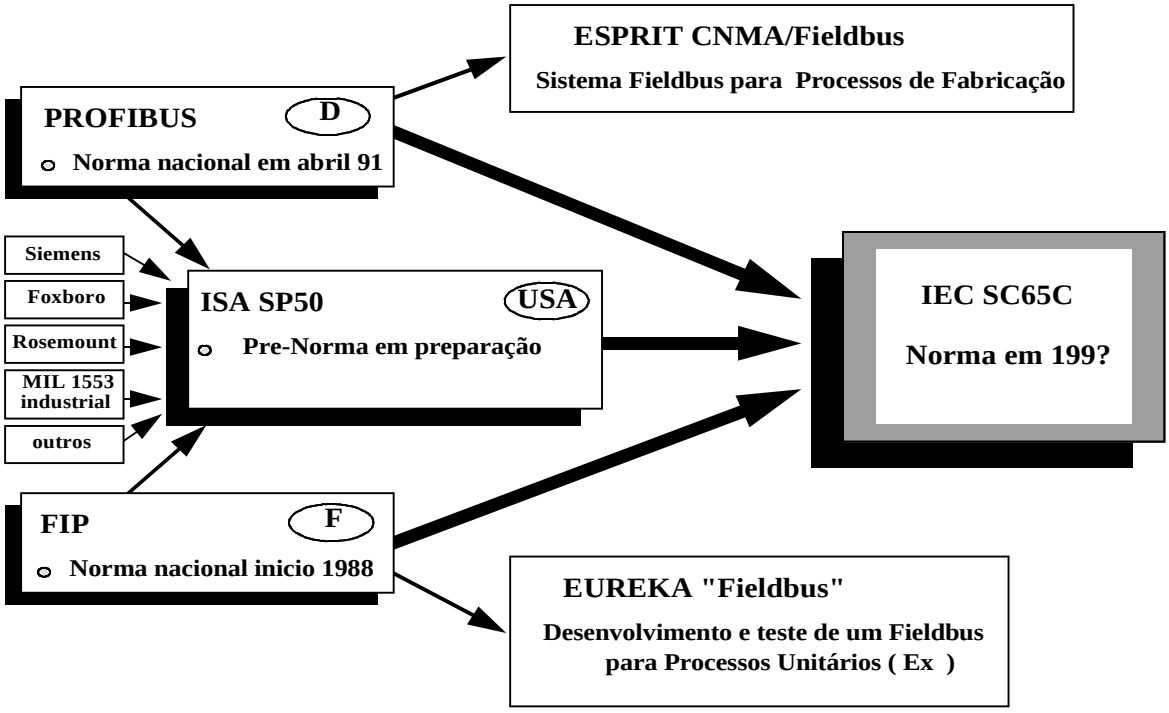


Figura 4.2 - Atividades de Normalização do Fieldbus

Dentre os sistemas Fieldbus atualmente em discussão, os mais fortes candidatos à normalização são o PROFIBUS (Process Fieldbus, proposta alemã) e o FIP (Factory Instrumentation Protocol, proposta francesa). Nos Estados Unidos foi definido o grupo de

estudos SP-50 da ISA (Intrumentation Society of America), que deverá apresentar brevemente uma proposta de normalização americana e que leva em consideração partes das propostas FIP e PROFIBUS bem como as propostas das firmas Faxboro e Rosemount / Philips.

Todas estas propostas serão examinadas pela comissão SC65C/WG6 da IEC (International Electrotechnical Comission) para a elaboração de uma norma internacional. As expectativas são de que a passagem da norma americana elaborada pela ISA SP-50 para a norma internacional da IEC deverá representar um passo meramente formal.

Ao lado das questões de normalização permanecem ainda abertas à discussão questões referentes ao espectro de aplicações a ser atendido pelo Fieldbus. Por um lado é desejada uma boa compatibilidade e interconectabilidade com os níveis hierárquicos superiores de automação, por outro devem ser considerados os requisitos técnicos e econômicos para a conexão de componentes inteligentes simples e de baixo custo.

Enquanto soluções que permitam uma compatibilidade com a definição da camada de aplicação adotada no sistema MAP através da definição de Subsets do MMS são nitidamente preferidas, permanece aberta a questão da adequação de tais sistemas para o acoplamento direto de sensores e atuadores em processos com dinâmica elevada, como é o caso de sistemas de controle em malha fechada na fabricação. A forma de estruturação física de uma tal malha de controle com Fieldbus é mostrada na Figura 4.3.

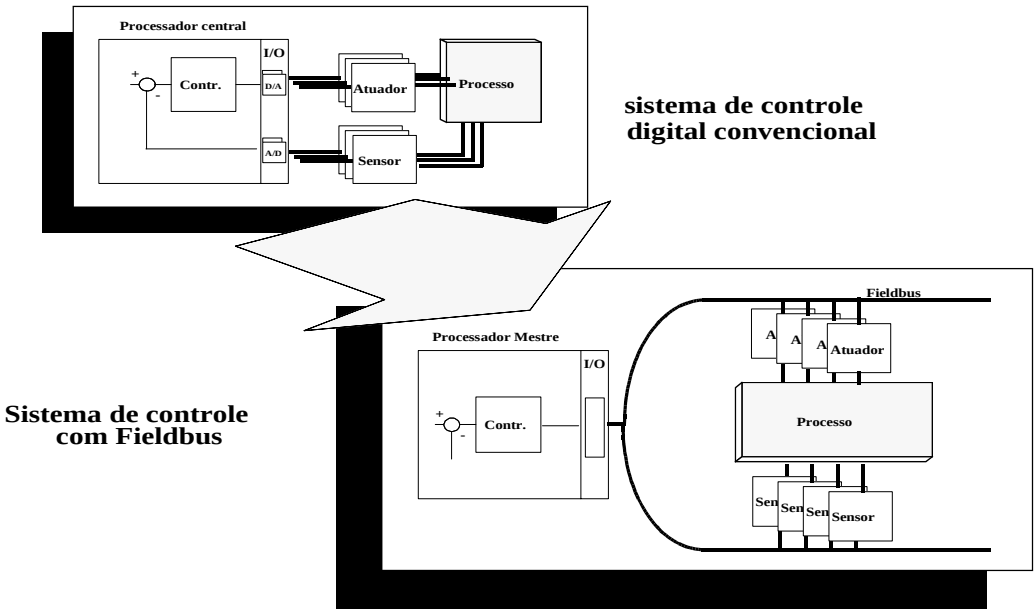


Figura 4.3 - Malha Fechada de Controle com Fieldbus

Nos itens seguintes apresentaremos as principais características dos sistemas Fieldbus atualmente propostos para padronização internacional.

3.4.2 A proposta FIP (Factory Instrumentation Protocol)

3.4.2.1. INTRODUÇÃO

O projeto FIP foi elaborado por um conjunto de empresas européias (Telemecanique, CEGELEC, CGEE Alsthom), órgãos do governo francês e centros de pesquisa conglomerados em torno do chamado "Club FIP". A proposta procurou levar em consideração as restrições de tempo real impostas por um grande número de aplicações a nível de chão de fábrica. Definiu-se um modelo de transmissão "produtor-consumidor", que difere das soluções mais usualmente encontradas em redes locais. Para este fieldbus existem "chips" integrados que implementam as funções das três camadas (FIPART, FULLFIP).

3.4.2.2. A CAMADA FÍSICA

FIP oferece opcionalmente como meios de transmissão a fibra ótica e o par trançado (blindado ou não). Para o par trançado são previstas três velocidades de transmissão de dados:

- S1: 31.25 Kbps, distância até 2000 m;
- S2: 1 Mbps, distância até 500 m, par trançado blindado;
- S3: 2.5 Mbps, distância até 200 m, par trançado blindado.

A velocidade padrão é a S2. A velocidade S1 é utilizada em áreas sujeitas a explosão (segurança intrínseca). S3 é utilizadas para aplicações que requerem elevado desempenho temporal. Para uso com fibra ótica é prevista uma velocidade adicional de 5 Mbps.

Os bits a serem enviados são codificados em Manchester, que permite o envio simultâneo do sinal de sincronização e dos dados propriamente ditos.

A camada física do FIP suporta segmentos com comprimento de até 2000 m e até 256 estações. O barramento principal pode ser decomposto em vários segmentos, que são ligados a este por meio de "taps". Os segmentos podem ser interligados por meio de repetidores, que fortalecem e reconstituem o sinal.

A camada física oferece os seguintes serviços de comunicação:

- PHY_data_request: pedido de transmissão de dados;
- PHY_data_indication: indicação de serviço concluído;

Além disso, a camada física oferece serviços de gerenciamento, que são:

- PHY_Reset: reinicialização do nível físico;
- PHY_SetValue: ajuste de parâmetros da camada física;
- PHY_ReadValue: leitura de parâmetros ajustados;
- PHY_Event: comunicação de eventos do nível físico.

3.4.2.3. A CAMADA DE ENLACE

A camada de enlace do FIP não faz uma distinção formal entre subcamadas LLC e MAC, como proposto na norma IEEE 802.

O método de acesso ao meio da rede FIP é baseado na difusão ("Broadcasting"). A difusão é organizada por uma entidade centralizada denominada "árbitro de barramento". O projeto FIP baseou-se no fato de que, nas redes industriais, uma informação gerada em um determinado ponto pode interessar a várias outras estações (por exemplo, o dado gerado por um sensor de temperatura pode interessar ao controlador, ao atuador e ao terminal de vídeo do operador, simultaneamente).

A maioria dos dados transmitidos pelo barramento érepresentada por objetos (variáveis). Cada objeto é representado por um "nome" único no sistema. Um objeto é, por definição, elaborado por um único transmissor (produtor) e levado em conta por qualquer número de receptores (consumidores). Devido ao uso da difusão, os endereços de transmissores e receptores não precisam ser conhecidos pelas aplicações.

A comunicação transcorre da seguinte forma (figura 4.4):

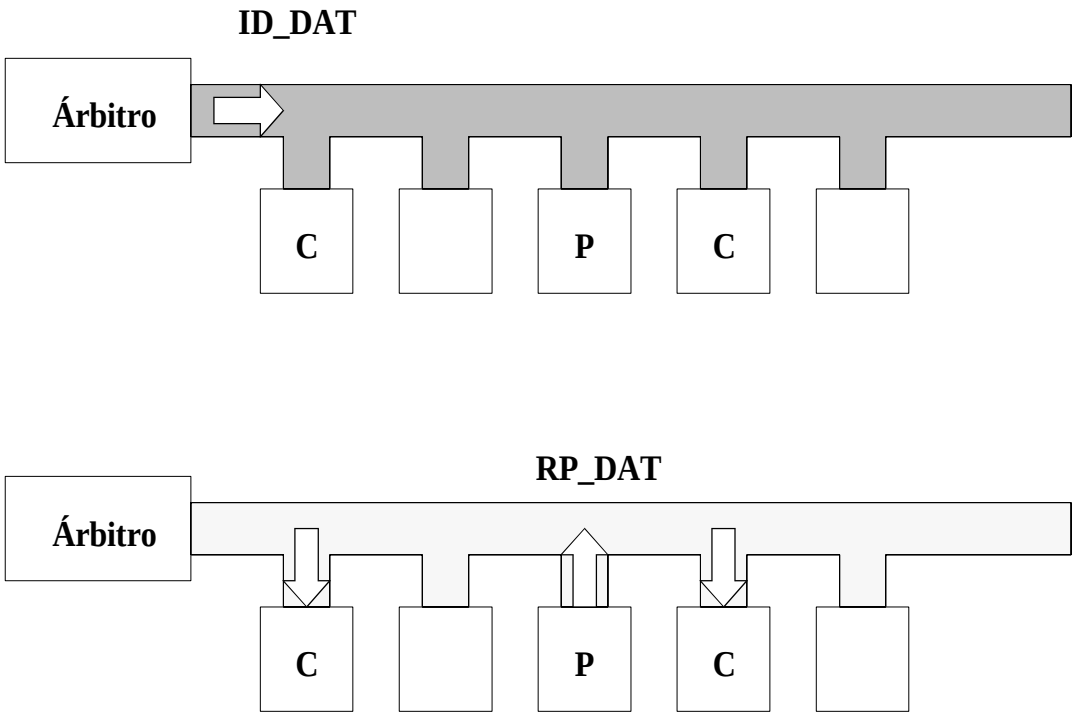


Figura 4.4 - Método de acesso ao meio do FIP. a) Árbitro difunde identificador. b) Produtor difunde dados pedidos.

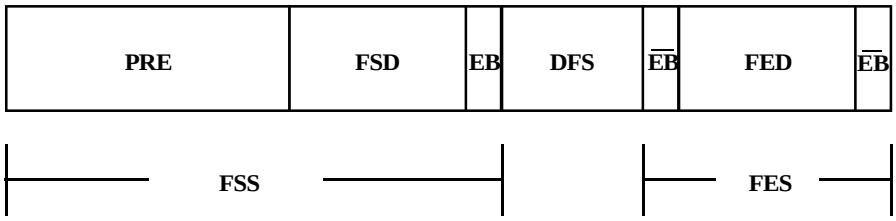
- Em uma primeira fase, o árbitro difunde na rede o nome da variável (objeto) a ser transmitida (quadro de identificação);
- O produtor da variável difunde, em seguida, a informação ligada ao identificador (quadro de dados);

- todos os consumidores interessados passam a copia-la na fase final.

Cada estação é completamente autônoma. O único requisito imposto às estações é o de difundir, por solicitação do árbitro de barramento, a variável ou variáveis por elas produzidas. Naturalmente, as estações devem também aceitar as variáveis que lhe são enviadas. A varredura das variáveis periódicas é feita a partir de uma lista implementada no árbitro na fase de inicialização e que, em geral, não é alterada posteriormente.

A transmissão de mensagens é feita conforme a norma IEEE 802.2, LLC tipos 1 (sem conexão) e 3 (com reconhecimento). Até 24.000 objetos (variáveis) são identificáveis e podem ser trocados de forma cíclica utilizando uma tabela configurável.

O formato do quadro do FIP é mostrado na figura 4.5, onde:



PRE : Preâmbulo
FSD : Frame Start Delimiter
EB : Equalization Bit
DFS : Data Frame Sequence - Controle, dados, FCS
FSS : Frame Start Sequence
FED : Frame End Delimiter
FES : Frame End Sequence

Figura 4.5 - Formato do quadro FIP

- PRE: preâmbulo, utilizado para sincronização entre emissor e receptores. Compõe-se de um caractere de 16 bits com alterações dos bits adjacentes de 1 para 0.
- FSD/FED: delimitadores de inicio e fim de quadro, respectivamente. São codificados de forma a não serem confundidos com dados.
- EB: Bits de equalização, que operam como bits de interface entre os delimitadores e os dados codificados em Manchester II.
- Controle: define tipo de quadro, podendo ser quadro de identificação de informação ou de envio de informação (6 bits).
- Dados: pode conter um identificador representando o endereço lógico de uma variável, o valor de uma variável, uma mensagem, o reconhecimento de uma mensagem ou uma lista de identificadores.
- FCS: o controle de erros é feito utilizando a técnica polinomial, permitindo uma distância de Hamming de 4 e adotando o polinômio gerador proposto pela CCITT.

Os serviços oferecidos pela camada de enlace são apresentados na tabela 4.2. Foram especificadas quatro classes de serviços:

- atualização cíclica de dados;
- atualização não periódica de dados;
- transferência de mensagem com ACK;
- transferência de mensagem sem ACK.

Maior importância foi atribuída aos serviços cíclicos, de forma que não houve preocupação com uma otimização das demais classes.

Classe	**Primitiva de serviço**	**Comentários**
Atualização cíclica de dados	L_PUT.req/cnf L_SENT.ind L_GET.req/cnf L_RECEIVED.ind	atualiza dados sinaliza envio busca de dados sinaliza recepção
Atualização não periódica de dados	L_PARAM.req/cnf	requisita um dado
Transmissão de mensagem com ACK	L_MESSAGE_ACK.req/ ind/cnf	com reconhecimento
Transmissão de mensagem sem ACK	L_MESSAGE.req/ind	sem reconhecimento

Tabela 4.2 - Serviços de enlace do FIP

3.4.2.4. A CAMADA DE APLICAÇÃO

O projeto FIP propõe, a nível da camada de aplicação, um sub-conjunto do MMS apresentado no projeto MAP, definindo serviços de mensagem industrial que não interferem com o tráfego de tempo real. Além disso, foi definida uma segunda família de serviços, denominada MPS ("Message Periodic/Aperiodic Services"), apresentada na tabela 4.3.

Classe	**Primitiva de serviço**	**Comentários**

Leitura de variáveis	A_READ.req/cnf A_READFAR.ind	lê nomes de variáveis, estruturas, status, valores
Escrita de variáveis	A_WRITE.req/cnf A_WRITEFAR.ind	escreve especificação, valor, status
Leitura do tipo de variável	A_GETOBJECT_DESCRIPTION.req/ cnf	lê especificação
Acesso à listas de variáveis	A_READLIST.req/cnf A_WRITELIST.req/cnf	lê e escreve atributos, valores
Serviços de sincronização	A_SEND.ind A_RECEIVE.ind	sincronização local e remota

Tabela 4.3 - Serviços MPS do FIP

3.4.2.5. FUNÇÕES DE GERENCIAMENTO DA REDE

O projeto FIP definiu uma série de funções de gerenciamento de rede, que permitem coordenar as várias fases do ciclo de vida da mesma. Para inicializar a rede é preciso:

- Configurar a rede (definição de parâmetros, identificadores e listas de objetos, gerar tabelas de varredura);
- Implementar e testar a configuração.

Durante a operação da rede, é necessário realizar sua manutenção, que inclui:

- Atualização das listas de objetos;
- Atualização das tabelas de varredura;
- Gerenciamento das operações de partida e parada;
- Detecção e correção de falhas;

3.4.3. A proposta PROFIBUS (PROcess FIeld BUS)

3.4.3.1. INTRODUÇÃO

O PROFIBUS (Process Field Bus) foi desenvolvido na Alemanha, inicialmente pela Siemens em conjunto com a Bosch e Klockner-Moeller, em 1987. Em 1988 tornou-se um "Trial Use Standard" no contexto da norma DIN (DIN 19245, parte 1), que define as camadas Física e Enlace. Posteriormente, um grupo de 13 empresas e 5 centros de pesquisa propuseram alterações nas camadas Física e Enlace e definiram a camada de Aplicação (norma DIN 19245, parte 2). O PROFIBUS representa a proposta alemã para a padronização internacional

do fieldbus. Esta proposta é atualmente apoiada por cerca de 110 empresas européias e internacionais (Siemens, ABB, AEG, Bosch, entre outras). Em 1996 tornou-se um padrão da comunidade européia, sob a designação EN50170.

Como nas demais redes fieldbus para aplicações industriais, para atender aos requisitos de tempo de resposta, o PROFIBUS implementa o modelo de referência ISO/OSI reduzido a três camadas (1, 2 e 7), descritas a seguir.

3.4.3.2. A CAMADA FÍSICA

A camada física do PROFIBUS baseia-se no padrão EIA RS-485 (EIA - Eletronic Insdustries Association), topologia barramento, utilizando como meio um par trançado blindado (130 Ohms). Permite a interligação de até 32 elementos (estações ativas, passivas ou repetidoras) por segmento. São permitidos até 4 segmentos, totalizando assim um máximo de 122 estações.

A codificação utilizada é a NRZ, podendo ser implementada com uma USART simples (assíncrona). As taxas de transmissão propostas na norma original são 9.6, 19.2 e 93.75 kbps para distâncias até 1200 m, 187.5 kbps para distâncias até 600 m e 500 kbps até 200 m.

A camada física oferece duas primitivas de serviço acessíveis à camada de enlace: PHY_DATA.request (requisição de envio de dados) e PHY_DATA.indication (indicação da presença de dados).

3.4.3.3. A CAMADA DE ENLACE

A exemplo da norma IEEE 802, o PROFIBUS define duas subcamadas para a camada de Enlace de Dados: a subcamada de controle de acesso ao meio (MAC) e a subcamada de controle de ligação lógica (LLC).

Na subcamada MAC, o PROFIBUS combina dois métodos determinísticos de acesso ao meio: as estações ditas "ativas" encontram-se em um anel lógico, no qual o direito de acesso ao meio é repassado ciclicamente por passagem de token (baseado na especificação IEEE 802.4), enquanto as estações ditas "passivas" comportam-se como escravas, isto é, só tem acesso ao meio por requisição da estação ativa detentora do token. O PROFIBUS representa assim uma combinação dos métodos "Master/Slave" e "Token-Passing" (figura 4.6). O token só é repassado entre as estações ativas.

Estações ativas podem entrar e saír do anel lógico de forma dinâmica, como previsto na norma IEEE 802.4. O token é repassado na ordem ascendente de endereços, seguindo uma lista de estações ativas (LAS, List of Active Stations). Esta lista é gerada na inicialização e atualizada sempre que uma estação entra ou sai do anel lógico.

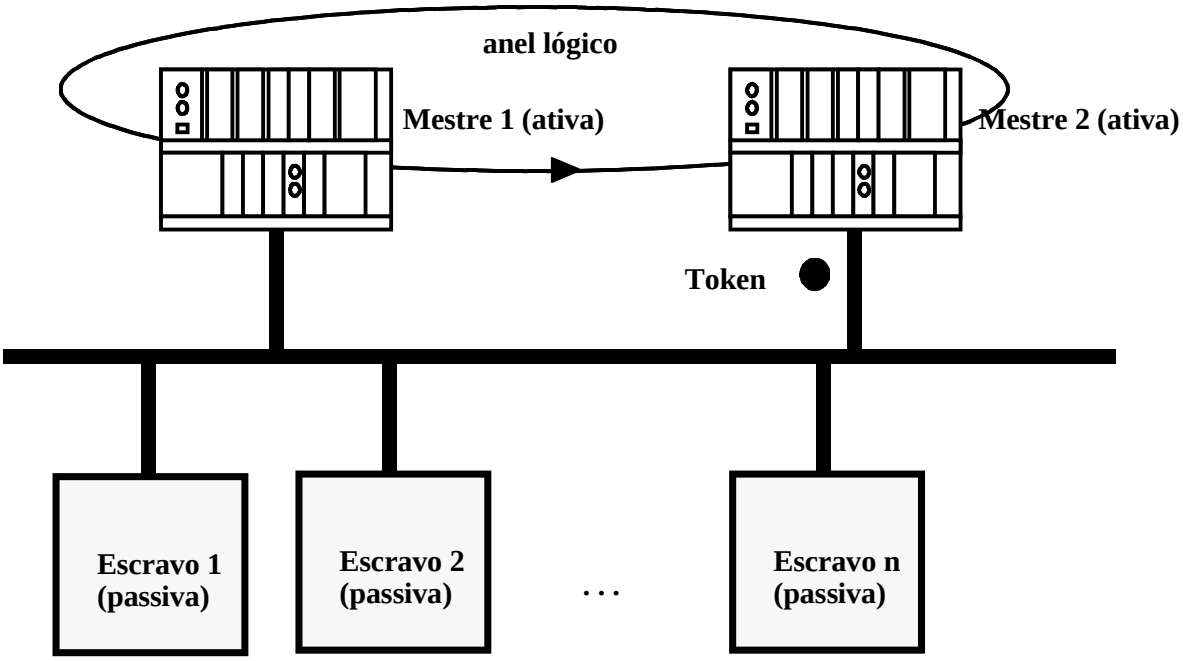


Figura 4.6 - Método de acesso ao meio do PROFIBUS

O PROFIBUS prevê uma série de quadros diferentes, agrupados em duas classes, uma para transmissão entre estações mais complexas (quadros longos) e outra para dispositivos de campo simples (quadros curtos). Os quadros previstos incluem:

- quadro longo sem campo de dados;
- quadro longo com campo de dados fixo;
- quadro longo com campo de dados variável;
- quadro curto sem campo de dados;
- quadro curto com campo de dados;
- quadro curto de passagem de token.

Cada quadro é formado por vários caracteres UART. Um caracter UART é composto de 11 bits: 1 bit de inicio ("start bit"), 8 bits de dados, 1 bit de paridade e 1 bit de finalização ("stop bit"). Em particular têm-se os seguintes campos em cada quadro:

- SD: Start Delimiter (delimitador de inicio de quadro);
- DA: Destination Address (Endereço de destino do quadro);
- SA: Source Address (Endereço de origem do quadro);
- FC: Frame Control (Controle de quadro);
- DATA_UNIT: Dados propriamente ditos;

- FCS: Frame Check Sequence (seqüência de checagem de erros);
- ED: End Delimiter (Delimitador de fim de quadro).

O protocolo implementado na camada de enlace é denominado FDL ("Fieldbus Data Link"), que oferece serviços de administração do token, de atualização das estações e de transferência de dados. Estes últimos são apresentados na tabela 4.4. A troca de dados é feita em ciclos compostos por um "send-request" de parte da estação ativa e seguida por um "ack-response" de parte de uma estação passiva ou de outra estação ativa.

Classe	**Primitiva de serviço**	**Comentários**
SDN (Send Data with No Acknowledge)	FDL_DATA	envio de dados sem reconhecimento
SDA (Send Data with Acknowledge)	FDL_DATA_ACK	envio de dados com reconhecimento
RDR (Request Data with Reply)	FDL_REPLY FDL_REPLY_UPDATE	requisição de dados com reconhecimento
CRDR (Cyclic Request Data with Reply)	FDL_CYC_REPLY FDL_CYC_DEACT FDL_REPLY FDL_REPLY_UPDATE	estação local requisita ciclicamente dados ao usuário remoto.
CSRD (Cyclic Send and Request Data)	FDL_SEND_UPDATE FDL_CYC_DATA_REPLY FDL_CYC_DEACT FDL_DATA_REPLY FDL_DATA_UPDATE	estação local envia ciclicamente e requisita simultaneamente dados de resposta.
SRD (Send and Request Data)	FDL_DATA_REPLY FDL_REPLY_UPDATE	estação local envia e requisita dados.

Tabela 4.4 - Serviços da camada de enlace (FDL) do PROFIBUS

3.4.3.4. A CAMADA DE APLICAÇÃO

Na camada de aplicação foi definido um subset do MMS (ISO 9506), utilizando primitivas de serviços apropriadas para atender os aspectos específicos do barramento de campo. A Camada de Aplicação está dividida em três subcamadas:

- Fieldbus Message Specification (FMS), que é o protocolo propriamente dito, derivado do MMS;
- Lower Layer Interface (LLI), responsável pela interface com a camada de Enlace, mapeando os serviços FMS e FMA (Fieldbus Management) em serviços correspondentes do FDL;
- Application Layer Interface (ALI), responsável pela interface com as aplicações do usuário. A ALI opera como solicitador de serviços("Cliente") a um Dispositivo Virtual de Campo (VFD, Virtual Field Device), que opera como fornecedor de serviços ("Servidor").

O VFD é equivalente ao VMD (Virtual Manufacturing Device) do MMS usado em MAP, constituindo uma representação abstrata de várias classes de dispositivos reais.

As classes de onjetos definidas são: Variáveis, Domínios, Alarmes, Listas de Variáveis e Invocações de Programa. Todos os objetos da rede são cadastrados em um diretório de objetos, denominado OV (Objekt Verzeichniss). Cada estação contem uma cópia total ou parcial do OV.

São definidos serviços com e sem conexão, cíclicos e acíclicos, entre estações ativas e entre estas e estações passivas. Os serviços sem conexão são usados para "Broadcast" e "Multicast" (difusão de quadros entre todas as estações ou entre certos agrupamentos pré-definidos de estações). Os serviços com conexão são usados para garantir uma troca de dados confiável.

A tabela 4.5 apresenta uma visão geral dos serviços oferecidos, que podem ser decompostos em três classes: serviços de Aplicação, serviços de Adminstração e serviços de Gerenciamento de Rede.

Em relação ao MMS, tem-se como novidade os serviços Phy_Read e Phy_Write, que permitem acesso direto a strings de octetos para os quais não se tenha uma descrição no OV.

Classe	Primitivas de serviço	Comentários
Serviços de Acesso a variáveis	READ WRITE INFORMATION_REPORT PHY_WRITE PHY_READ DEFINE_VARIABLE_LIST DELETE_VARIABLE_LIST	leitura e escrita de variáveis contidas em dispositivos servidores
Serviços de Acesso a Domínios	INITIATE_DOWNLOAD_SEQUENCE DOWNLOAD_SEGMENT TERMINATE_DOWNLOAD_SEQUENCE INITIATE_UPLOAD_SEQUENCE UPLOAD_SEGMENT TERMINATE_UPLOAD_SEQUENCE REQUEST_DOMAIN_DOWNLOAD REQUEST_DOMAIN_UPLOAD	transferência de dados ou programas de dispositivo cliente para dispositivo servidor e vice-versa
Serviços de Invocação de Programas	CREATE_PROGRAM INVOCATION_DELETE_PROGRAM INVOCATION_START INVOCATION_STOP INVOCATION_RESUME INVOCATION_RESET	partida, parada, retorno da execução, retorno ao estado inicial e deleção de programas
Serviços de Notificação de Eventos	ALTER_EVENT_COND._MONITORING EVENT_NOTIFICATION ACK_EVENT_NOTIFICATION	servidor notifica cliente a ocorrência de um evento (alarme)
Serviços de Leitura de Status	STATUS UNSOLICITED_STATUS STATUS_IDENTIFY	informações acerca do estado dos dispositivos servidores
Serviços de Gerenciamento de Dicionário de Objetos	GET_OV PUT_OV INITIATE_PUT_OV TERMINATE_PUT_OV	descrição de todos os objetos na rede (nomes, endereços, tipos de dados, etc)
Serviços de Gerenciamento de Contexto	INITIATE REJECT ABORT	estabelecimento e encerramento de associação entre dois dispositivos e a rejeição de mensagens recebidas

Tabela 4.5 - Serviços de aplicação do PROFIBUS

3.4.4. A proposta ISA SP-50

3.4.4.1. INTRODUÇÃO

Os projetos FIP e PROFIBUS contribuíram grandemente para o projeto SP-50 (Standards & Practices 50), em elaboração na ISA (instrumentation Society of America), que deverá tornar-se a proposta definitiva para o fieldbus padronizado. Os trabalhos de padronização ainda estão em andamento.

A ISA adotou a terminologia da CCITT, na qual um elemento de comunicação é composto de duas partes:

- DTE (Data Terminal Equipment), que aqui inclui as funcionalidades das camadas de aplicação, de enlace e a parte da camada física independente do meio de transmissão adotado;
- DCE (Data Communication Equipment), que aqui inclui as partes da camada física que dependem do meio.

3.4.4.2. A CAMADA FÍSICA

A camada física da ISA SP-50 se compõe de três subcamadas (figura 4.7):

- Subcamada DIS (Data Independent Sublayer): atua como interface com a camada de enlace, recebendo pedidos de serviço desta e repassando os dados recebidos (DTE);

- Subcamada MDS (Medium Dependent Sublayer): codifica os dados a enviar para um formato compatível com o meio físico a ser adotado. Por enquanto, somente foi especificada a subcamada MDS para o par trançado, onde se adota a codificação tipo Manchester bifásico (DCE);

- Subcamada MAU (Medium Attachment Unit): descreve o transceptor propriamente dito para o meio físico (DCE). Para a camada física foram inicialmente propostos dois tipos de meio:
    - meio H1: para aplicações em controle de processos em áreas de segurança intrínseca, usando par trançado, com taxa de transmissão de 31.25 Kbps, cabo de até 1900 metros, que é também utilizado para a alimentação dos dispositivos de campo. As topologias propostas são barramento e estrela;

- meio H2: para processos de controle de alta velocidade, com taxa de transmissão de 1 Mbps ou 2.5 Mbps, topologia em barramento e distância de 750 m para 1 Mbps ou 500 m para 2.5 Mbps, com 32 estações por segmento.

Propostas alternativas utilizando fibra ótica e sinais de rádio estão sendo estudadas.

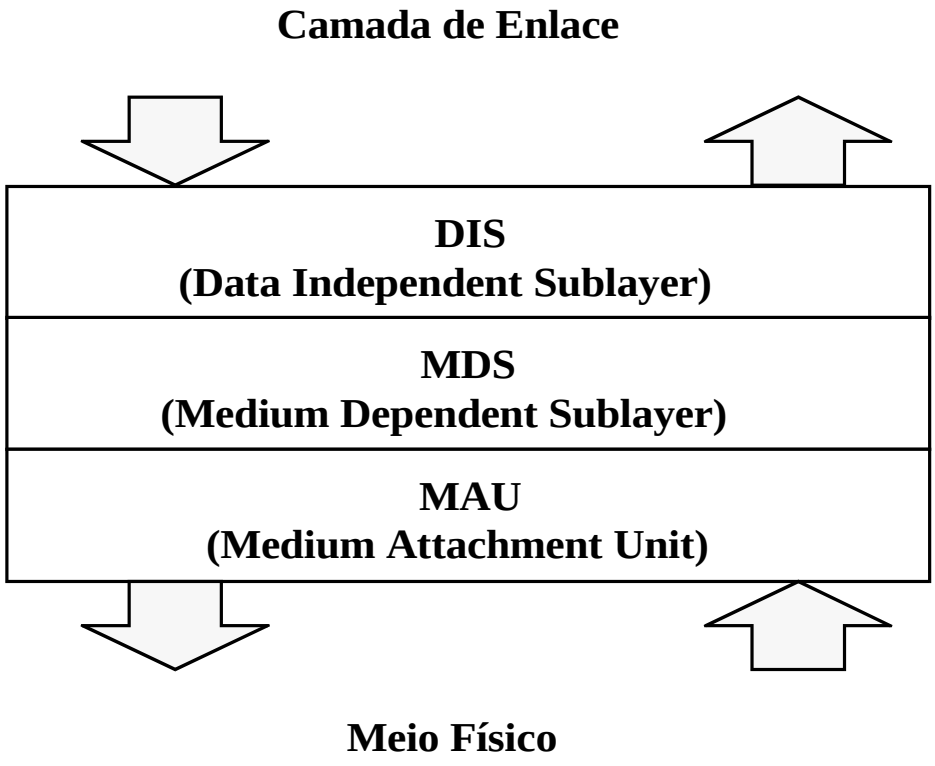


Figura 4.7 - A camada física do SP-50

3.4.4.3. A CAMADA DE ENLACE

A camada de Enlace oferece quatro classes de funções, a serem implementados nas estações de acordo com as necessidades:

- funções de "Responder": estação só transmite dados em resposta a uma solicitação (estação "escrava");
- funções de "Initiator": estação pode se apoderar do direito de acesso ao meio (token), podendo enviar e requisitar dados a outras estações por iniciativa própria;
- funções de "Link-master": inclui as funções de responder e initiator, mas a estação pode exercer o papel de escalonador de enlace (LAS), administrando o token e gerenciando o tempo interno do sistema (semelhante ao papel do árbitro de barramento do FIP);
- funções de "Bridge": estação capaz de interligar entidades de enlace diferentes;

Se há mais de uma estação com as funcionalidades de um "Link-master" no sistema, estas disputam entre si na inicialização o papel de escalonador de enlace (árbitro). A estação vencedora é chamada LAS (Link Active Scheduler). Existem, para isto, três tipos de token (figura 4.8):

- Token de escalonamento (Scheduler Token): este token é disputado na inicialização por todas as estações tipo Linkmaster e define a estação LAS, que o retém.
- Token circulado ou de Resposta (Reply Token): distribuído pela estação LAS às demais estações com funcionalidade de Linkmaster, que formam um anel lógico conforme a norma IEEE 802.4.
- Token delegado (Delegated Token): enviado pela estação LAS a uma estação qualquer por solicitação desta ou para atender às necessidades de um serviço de comunicação escalonado pela LAS.

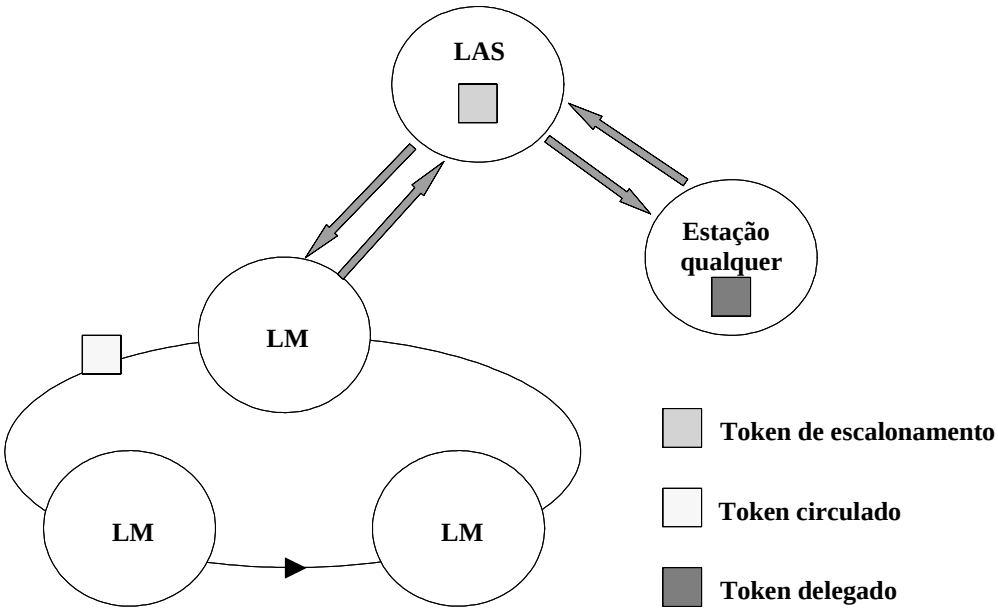


Figura 4.8 - Operação dos tokens no SP-50

Funcionalmente, a camada de Enlace está subdividida em quatro subcamadas, conforme mostrado na figura 4.9:

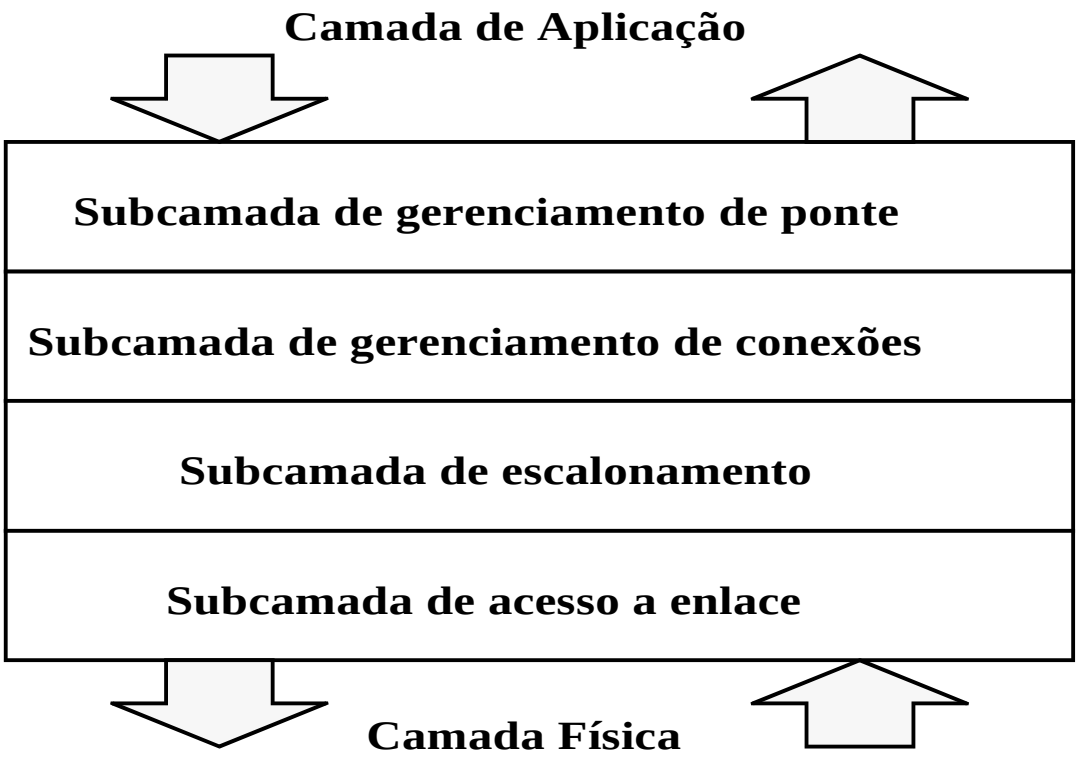


Figura 4.9 - A camada de Enlace do SP-50

- Subcamada de acesso a Enlace: estabelece a interface com a camada física, gerencia o token e serviços de resposta imediata;
- Subcamada de escalonamento de Enlace: providencia as funções de escalonamento de atividades na própria entidade de enlace. Esta subcamada é mais complexa em estações com as funcionalidades de Linkmaster e que podem assumir a função de LAS;
- Subcamada de gerenciamento de conexões: executam todas as funções referentes ao estabelecimento e rompimento de conexões;
- Subcamada de gerenciamento de Ponte: só existe em estações com funcionalidades de Bridge.

A camada de enlace oferece quatro classes de serviços às entidades da camada de aplicação:

- Serviços de gerenciamento de Buffers e filas: através deles, usuários da camada de enlace podem alocar permanentemente buffers ou filas com profundidade especificada, a serem usados posteriormente para a transferência de dados;
- Serviços de transferência de dados com conexão: permitem o estabelecimento de uma conexão e o envio de dados entre estações, garantindo serviços altamente confiáveis;

- Serviços de transferência de dados sem conexão: permitem o envio de dados sem o estabelecimento prévio de uma conexão e que, apesar da menor confiabilidade, são úteis no envio de telegramas de difusão (multicast e broadcast);
- Serviços de escalonamento de transações: permitem programar o Escalonador Ativo de Enlace (Link Active Scheduler, LAS), definindo a seqüência de passagem de token (escalonamento do meio entre as estações).

3.4.4.4. A CAMADA DE APLICAÇÃO

A camada de aplicação do SP-50 ainda se encontra em discussão e procura conjugar as idéias básicas do MMS, para aplicações sem restrições temporais, com serviços tipo READ/WRITE inspirados no FIP e que atendem o tráfego cíclico e acíclico com requisitos de tempo real "duro".

A camada de aplicação prevê, em sua fase atual, os seguintes conjuntos de serviços:

- Serviços MCSE (Message Common Service Element): estabelece e interrompe conexões entre processos de aplicação (Correspondem aos serviços ACSE da ISO).
- Serviços IMSE (Industrial Message Service Element): serviços semelhantes aos oferecidos pelo MMS do projeto MAP.
- Serviços DDM (Distributed Database Maintenance): Serviços de acesso à bases de dados distribuídas.

3.4.4.5. CAMADA DO USUÁRIO

Uma proposta inovadora no SP-50 é a definição de uma camada do usuário, situada acima da camada de aplicação, destinada a aliviar o programador de detalhes de acesso ao sistema de comunicação e a oferecer serviços adequados a diversos tipos diferentes de aplicações (a semelhança dos "companion standards" propostos no projeto MAP).

Os trabalhos atuais estão restritos a definição de funções para aplicações em controle de processos, agrupadas na chamada PCUL ("Process Control User Layer"). Outros trabalhos deverão atender as áreas de automação da manufatura, controle predial, eletrônica embarcada (automóveis), aplicações domésticas, entre outras.

3.4.4.6. SERVIÇOS DE GERENCIAMENTO DE REDE

Além das funcionalidades das camadas já vistas, a proposta SP-50 inclui um conjunto de funções de gerenciamento de rede, como:

- Gerenciamento de configuração de rede: carregamento, inicialização de endereços, configuração de comunicação e aplicação, partida, etc;
- Controle de operação: ferramentas de sincronização, escalonamento, etc;
- Monitoração de desempenho: detecção, diagnose e recuperação de erros, avaliação e otimização de desempenho, etc.

3.4.5. Conclusões

Uma vez definido um padrão internacionalmente aceito, o Fieldbus deverá revolucionar o setor de instrumentação. Esta tecnologia permite que a inteligência seja totalmente distribuída pelo campo e favorece o surgimento de dispositivos com capacidades locais de processamento cada vez mais sofisticadas. A integração total dos equipamentos permitirá alterações nos procedimentos de operação das plantas industriais. O Fieldbus deverá também propiciar a intercambiabilidade a nível de sensores, atuadores, transmissores e controladores, trazendo ao usuário uma maior flexibilidade na compra de produtos e abrindo espaço para novos fabricantes.

3.5. ANÁLISE DE ALGUNS PRODUTOS COMERCIAIS

3.5.1. Introdução

Discutiremos brevemente neste capítulo algumas soluções de mercado. Como existe uma imensa variedade de produtos comerciais, nos limitamos aqui a citar alguns dos mais difundidos, sem pretensões de apresentar uma lista completa.

3.5.2. Redes para Instrumentação

A interface de rede dominante na área de instrumentação (multímetros, osciloscópios e outros instrumentos hoje microprocessados) é a GPIB (General Purpose Interface Bus, barramento de interface de propósito geral). A origem deste sistema é um produto da Hewlett-Packard denominado HP-IB (Hewlet-Packard Interface Bus), que passou a ser norma americana sob a designação IEEE 488.1 e posteriormente norma internacional sob a designação IEC 625-1.

GPIB opera como um barramento paralelo, constituído de 16 linhas com sinal ativo baixo referenciado a um terra comum. Sinais com tensão acima de 2V são considerados como lógico 0 e abaixo de 0.8V como lógico 1. A interface permite portanto o envio de bits em série e de bytes em paralelo. Das 16 linhas que compõe o cabo, 8 são para os dados propriamente ditos, 3 servem para operações de *handshake* e 5 para gerenciamento da interface. A pinagem é descrita na tabela 5.1.

Categoria	**Linha**	**Nome**
8 Data lines	DIO 1-8	Data I/O
3 Handshake lines	DAV	Data Valid
	NRFD	Not Ready For Data
	NDAC	Not Data Accepted
5 Interface Management lines	REN	Remote Enable
	IFC	Interface Clear
	SRQ	Service Request
	EOI	Endo or Identify
	ATN	Attention

Tabela 5.1 - Pinagem do GPIB

As linhas de controle NRFD e NDAC operam no modo "wired-OR", de modo que só assumem o valor lógico TRUE no barramento quando todas as estações ligadas ao GPIB setam a linha correspondente local em TRUE (ativo baixo).

GPIB requer a existência de uma estação controladora (mestre) do barramento, que define em cada instante quem será a estação emissora (fonte dos dados, aqui chamada de "talker" ou falante) e quem serão as estações receptoras (destino dos dados, aqui chamados "listeners" ou ouvintes). A linha ATN distingue mensagens de dados (ATN=0) de mensagens dedicadas de gerenciamento da interface (ATN=1) como, por exemplo, mensagens para definir o talker e os listeners.

Após a definição, pela estação controladora, de quem serão o talker e os listeners, são executados os seguintes passos:

1. Se o talker tem um novo byte de dados a enviar, coloca seu valor nas linhas DIO 1-8;
2. Talker seta linha DAV (Data Valid) em true;
3. Listeners setam NRDF (Not Ready For Data) em true;
4. Listeners recebem o dado e setam NDAC (Not Data Accepted) em false (esta linha só assume o valor false quando todos os listeners receberem o dado, devido ao uso de wired-OR);
5. Talker seta DAV (Data Valid) em false e remove dados das linhas DIO 1-8;
6. Listeners setam NDAC (Not Data Accepted) em true;
7. Se listeners estiverem prontos para receber um novo byte de dados, setam NRFD (Not Ready For Data) em false;
8. Talker pode reiniciar processo do passo 1, enviando o byte de dados seguinte.

Um barramento GPIB pode ter até 15 estações (entre controladora, talkers e listeners), com um comprimento máximo de cabo de 20 metros, operando a uma taxa de transmissão de até 1Mbps.

Apesar da boa aceitação na área de instrumentação, GPIB não é uma interface bem adaptada às necessidades de automação de chão de fábrica (sensores, atuadores, robôs, CLPs, CNCs, etc.), pois os cabos e a própria interface de 16 condutores são caros, o sinal referenciado ao terra é sensível à perturbações eletromagnéticas e o comprimento máximo do barramento é uma limitação física indesejável.

3.5.3. Redes para automação de escritórios

Apesar de uma certa difusão de redes com interfaces paralelas para comunicação de dados na área de instrumentação (por exemplo GPIB), nas redes de comunicação para automação industrial as mensagens são enviadas de modo serial.

As redes locais atualmente mais utilizadas em automação são as redes ETHERNET (produto das empresas DEC, INTEL e XEROX), ARCNET (Datapoint) e TOKEN-RING (IBM). Estes produtos diferem nas velocidades, topologias e protocolos utilizados.

Tipicamente definem as duas primeiras camadas do modelo OSI (física e enlace). Algumas características destas redes são apresentadas na tabela 5.2.

	ETHERNET	**ARCNET**	**TOKEN-RING**
Acesso ao Meio	CSMA/CD	Token-passing	Token-passing
Velocidade	10 Mbps	2.5 Mbps	4 ou 16 Mbps
Número de nós	1024	254	255
Meio de transmissão	Par trançado Fibra ótica Cabo coaxial	Par trançado Fibra ótica Cabo coaxial	Par trançado Cabo coaxial
Topologia	Star/Bus	Star/Bus	Ring

Tabela 5.2 - Redes para escritório grandemente difundidas

Entre estas redes, a ARCNET (Attached Resource Computer Network) apresenta boas características para aplicação em ambiente industrial, devido a sua topologia, técnica de acesso ao meio e preço baixo. Existem várias instalações industriais em operação.

A Ethernet é a mais popular, mas tem como desvantagem o método não determinístico de acesso ao meio (CSMA/CD). Esta rede foi originalmente desenvolvida para aplicação em escritório e possui a maior quantidade de unidades instaladas no mercado. Apesar do método de acesso ao meio não determinístico, existem para ela muitas instalações industriais em operação, principalmente em aplicações com requisitos de tempo relaxados.

A rede Token-Ring é a mais popular entre os produtos da IBM. Apresenta como desvantagem um alto custo de instalação e baixa flexibilidade. Em contrapartida trabalha com elevada taxa de transmissão e inclui um grande número de soluções entre os produtos IBM. Devido ao método de acesso ao meio determinista, é uma boa opção para aplicações industriais.

3.5.4. Softwares para rede

Quanto aos Softwares para redes, existe também uma série de alternativas. Os mais difundidos são: LAN-Manager (Microsoft), LAN-Server (IBM), PC-LAN (IBM), Netware (NOVELL), NetBIOS (IBM), TCP/IP, entre outros.

3.5.4.1. NOVELL NETWARE

O sistema operacional para rede da NOVELL era até bem recentemente o mais difundido no mercado e pode operar sobre diferentes bases de hardware para rede (suporta as

redes ARCNET, Ethernet e Token-Ring). Oferece serviços a nível das camadas de sessão e apresentação.

3.5.4.2. LAN-MANAGER

O Sistema operacional para redes da Microsoft é o maior concorrente de mercado do Novell-Netware. Na versao original, devia operar sobre o sistema operacional OS/2 no servidor. Há versões para MVS (IBM) e VMS (DEC). O pacote foi integrado posteriormente no sistema operacional Windows 95. O LAN-Manager, a exemplo do Netware, oferece serviços a nível da camadas de sessão e apresentação mas engloba também as funcionalidades do protocolo TCP/IP (ver adiante).

3.5.4.3. LAN-SERVER

Sistema operacional para redes da IBM, inicialmente desenvolvido em conjunto com o LAN-Manager da Microsoft. Opera sobre o sistema operacional OS/2.

3.5.4.4. PC-LAN

Sistema operacional que opera sobre a rede Token-Ring da IBM, interligando computadores PC. Requer o NetBios para funcionar.

3.5.4.5. NETBIOS

O NetBios é uma Interface para Programas de Aplicação (API, Application Program Interface) desenvolvido pela IBM que provê serviços a nível das camadas de rede e transporte para PC-LAN e LAN-Server.

3.5.4.6. TCP/IP

Os protocolos TCP/IP foram desenvolvidos pela UCB (University of California at Berkeley) para a ARPA ("Advanced Research Projects Agency") em 1969, muito antes da definição do modelo de referência ISO/OSI. Sua aplicação original era militar (a ARPA é vinculada ao DoD, Departamento de Defesa dos EUA). O par de protocolos conhecido como TCP/IP é uma herança do projeto ARPANET.

Os serviços oferecidos pelo protocolo ARPANET permitem transferir arquivos (ftp), executar comandos remotamente (telnet), enviar e receber correio eletrônico (mail), entre outros.

O protocolo TCP ("Transmission Control Protocol") é usado para implementar o sequenciamento e o controle de fluxo de informações e corresponde aproximadamente à camada de transporte do modelo OSI.

O protocolo IP ("Internet Protocol") é um protocolo não orientado a conexão cujas funções correspondem aproximadamente às da camada de rede, mas engloba algumas funções pertencentes a subcamada LLC da camada de enlace do modelo ISO/OSI.

Este par de protocolos adquiriu uma grande importância, pois é hoje a base de funcionamento da INTERNET. Em função disto, veremos um pouco mais em detalhes os serviços oferecidos.

Estes protocolos costumam ser utilizados em uma arquitetura de rede diferente da proposta pela ISO, conforme veremos a seguir. O TCP/IP se estabeleceu como um padrão de fato para ligações de redes heterogêneas. O TCP/IP foi originalmente desenvolvido para interconectar máquinas de diversos fabricantes, ou seja, se tornar um protocolo universal. Para tanto foram disponibilizados os seguintes serviços:

Gerenciamento de redes:

- **DNS (Domain Name System):** é um esquema de gerenciamento de nomes, hierárquico e distribuído, que define a sintaxe dos nomes usados na Internet. Os endereços TCP/IP são numéricos (com uma formação dividida em classes), compostos uma parte destinada a endereçamento de rede e uma parte destinada a endereçamento de hosts (máquinas). O DNS contém um banco de dados distribuído, mantido por um conjunto de Servidores de Nomes (Name Servers), que permite fazer a resolução de endereços IP (numéricos) para o nome de uma máquina. Cada nível hierárquico de um nome é denominado um domínio (domain). Ex.: ***atlas.lcmi.ufsc.br*** é um nome composto de 4 domínios, que eqüivale ao endereço IP 150.162.14.1.
- **SNMP (Simple Network Management Protocol):** É uma aplicação TCP/IP, que providencia uma maneira de gerenciar objetos dentro de uma rede TCP/IP. Os processos que realizam o gerenciamento são denominados agentes e gerentes e tem por objetivo detectar a presença de falhas no funcionamento dos componentes da rede. O gerente envia comandos aos agentes, solicitando informações sobre o estado dos objetos gerenciados (comandos get e response) ou modificando este estado (comando put). Um agente pode também notificar o gerente da ocorrência de um evento específico (comando trap). Os objetos gerenciados podem ser estações de trabalho, gateways, modems, bridges, concentradores, processos, etc.
- **Finger:** é uma parte do protocolo TCP/IP, que providencia uma maneira de se verificar os hosts e users que estão conectados a um determinado host.

- **Ping:** é uma parte do protocolo TCP/IP , que providencia uma maneira de se verificar se um determinado host está ativo na rede. Funciona enviando uma

mensagem para o host e aguardando uma resposta. Se o host não responde significa que não está conectado à rede.

- **Netstat:** é uma parte do protocolo TCP/IP , que providencia uma maneira de se verificar as conexões que estão ativas na rede TCP/IP. Informa as conexões TCP no host, estado dos servidores TCP/IP neste host, bem como os Sockets utilizados, dispositivos e links, e a tabela de roteamento que está ativa.

Correio:

- **SMTP (Simple Mail Transfer Protocol):** é uma parte do protocolo TCP/IP , que providencia um correio eletrônico entre os usuários UNIX. Este correio permite a troca de mensagens e notas entre dois ou mais hosts. As notas são guardadas no SPOOL do UNIX.

Compartilhamento de arquivos:

- **NFS (Network File System):** é uma parte do protocolo TCP/IP , que providencia uma maneira de se compartilhar arquivos do sistemas através de uma rede TCP/IP. Ele realiza um mapeamento dos discos de um determinado servidor na rede TCP/IP, permitindo que os hosts desta rede enxerguem estes arquivos como locais.

Comunicação:

- **SLIP (Serial Line IP):** é um protocolo muito simples, utilizado para conectar-se dois hosts através de uma linha serial, configurando-se uma ligação ponto-a-ponto. Não providencia endereçamento; cada um dos hosts tem que ser conhecido pelo outro; não identifica pacotes e não possui correção de erros.
- **PPP (Point-to-Point Protocol):** protocolo que foi desenvolvido para substituir o SLIP, e contém as implementações que não são oferecidas pelo SLIP.

Emulação de terminais:

- **Telnet:** é uma parte do protocolo TCP/IP, que providencia um interface padrão através do qual um programa em um HOST (cliente Telnet) acessa recursos em outro HOST (servidor Telnet) como se fosse um terminal local conectado ao servidor de terminais.

APIs (Application Program Interfaces):

- **RPC:** é uma parte do protocolo TCP/IP , que providencia uma interface de aplicação que permite a comunicação entre dois programas, os quais são executados em dois hosts diferentes (processamento cooperativo, processamento cliente/servidor).

Transferencia de arquivos:

- **FTP (File Transfer Protocol):** é uma parte do protocolo TCP/IP, que providencia uma maneira de realizar a transferência de arquivos entre hosts UNIX. O usuário tem que se identificar para o host de onde o arquivo será transferido, ou seja tem que ser um usuário conhecido no host servidor.
- **TFTP (Trivial File Transfer Protocol):** é um simples protocolo para transferir arquivos entre dois hosts. Não leva em consideração nenhuma proteção e autenticação de usuário.

Aplicações gráficas:

- **XWINDOW (X Window System):** é uma parte do protocolo TCP/IP , o qual providencia apresentação em forma gráfica.

Impressão remota:

- **LPR (Line Printer Redirection):** executa o redirecionamento de arquivos de impressão para um host através de uma rede TCP/IP.
- **LPD (Line Printer Daemon):** servidor de impressão para hosts remotos em uma rede TCP/IP.

Execução remota:

- **RSH (Remote Shell Protocol):** executa um shell remotamente em outro host através de uma rede TCP/IP.
- **REXEC (Remote Execution Command Protocol):** é um servidor o qual permite a execução de um comando REXEC de um host remoto através de uma rede TCP/IP.

O servidor realiza um login automático incluindo a verificação do usuário. A parte cliente é realizada pelo processo REXEC.

Servidor de Boot:

- **BOOTP (BOOTstrap Protocol):** é um servidor de boot remoto para hosts através de uma rede TCP/IP. Muito utilizado por terminais gráficos (normalmente Diskless).

A figura 5.1 mostra uma relação aproximada entre o modelo de referência OSI/ISO e a arquitetura usualmente vinculada aos protocolos TCP/IP.

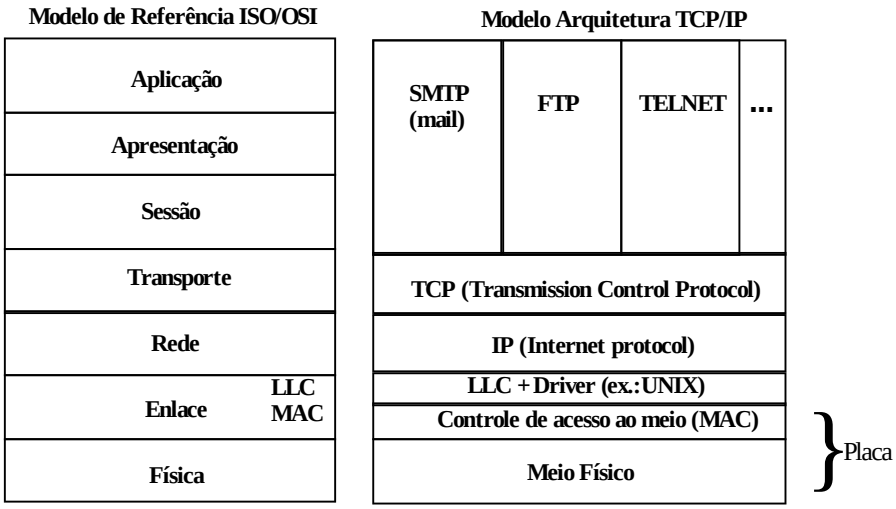


Figura 5.1 - TCP/IP e o RM-OSI/ISO

Uma relação aproximada entre os produtos apresentados e o modelo OSI é representada na figura 5.2.

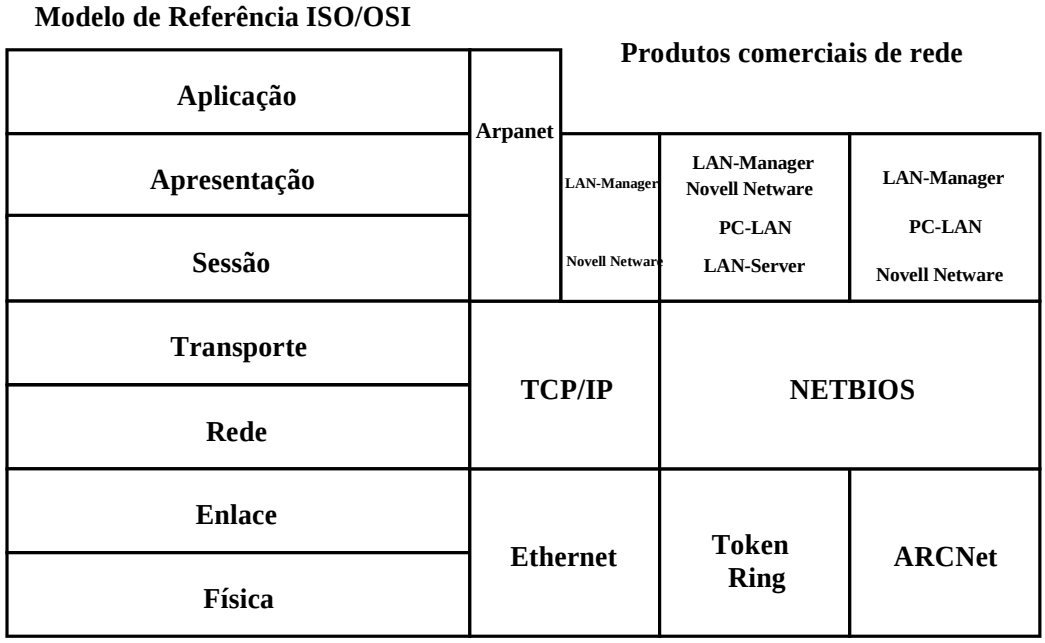


Figura 5.2 - Relação aproximada entre alguns produtos comerciais e o RM-OSI.

3.5.5. redes industriais

3.5.5.1. REDES SINEC (SIEMENS)

As redes SINEC, da Siemens, são um conjunto de soluções de rede englobando praticamente todos os níveis hierárquicos de uma empresa. A família SINEC Inclui:

- SINEC H1: rede compatível com a norma IEEE 802.3 (Ethernet).
- SINEC H2: rede compatível com o padrão MAP.
- SINEC L1: sistema fieldbus proprietário da Siemens.
- SINEC L2: rede fieldbus compatível com a norma alemã PROFIBUS. É oferecida em 3 versões:
    - SINEC L2-DP: sistema desenvolvido para aplicações que exijam respostas rápidas, especialmente aquelas envolvendo sistemas remotos de I/O (como CLPs ligados a sensores e atuadores). Utiliza o padrão RS485 ou fibra ótica na camada física. Com a interface RS485 pode-se ter um cabo de 1200 metros com uma taxa de transmissão de 93.75 Kbps, 1000 metros com taxa de 187.5 Kbps, 200 metros com taxa de 1.5 Mbps ou 100 metros com taxa de 12 Mbps. Suporta até 127 estações em 4 segmentos de rede ligados por repetidores. Prevê uma operação com mestre único (single master) e escravos, adotando portanto somente o protocolo

Mestre/Escravos na subcamada MAC. Usa somente serviços sem conexão e sem reconhecimento (LLC tipo 1). Os serviços de aplicação são basicamente voltados para leitura e escrita de variáveis remotas (READ/WRITE).

- SINEC L2-FMS: sistema concebido para a troca de dados entre sistemas inteligentes autônomos em sistemas de manufatura, como CNCs, CLPs, RCs, PCs, etc (ver figura 5.3). Utiliza também o padrão RS485 ou fibra ótica na camada física. Como as estações podem ser autônomas, utiliza na subcamada MAC os protocolos Token-Passing conjuntamente com Mestre/Escravos, conforme previsto na norma PROFIBUS. Também suporta 127 estações em 4 segmentos de rede, como o DP. Usa serviços LLC tipos 1 e 3. Os serviços de aplicação seguem o padrão FMS (Fieldbus Message Services, subconjunto do MMS da rede MAP).

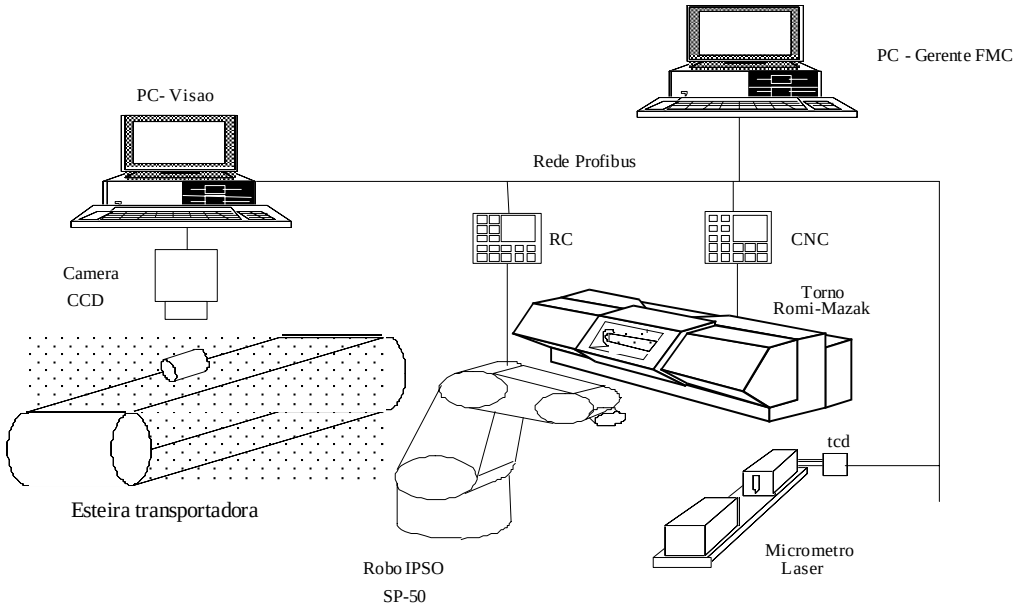


Figura 5.3 - Aplicação típica da Rede SINEC L2-FMS

- SINEC L2-PA: permite interligar instrumentos da área de processos unitários a um sistema de comunicação. Adota o padrão IEC61158-2 na camada física, que utiliza o próprio cabo de transmissão de dados para energizar os dispositivos de campo e opera com uma taxa de transmissão de 31.25 Kbps. Aqui o comprimento máximo do cabo depende do número de estações conectadas e é função de seu consumo de energia. Um segmento (sem repetidores) suporta no máximo 32 estações. A subcamada MAC utiliza o protocolo Mestre/Escravos e usa LLC tipo 1. Os serviços de aplicação são semelhantes aos do DP.

3.5.5.2. BITBUS (INTEL)

O Bitbus é uma rede com topologia em barramento e operando com o método de acesso ao meio mestre/escravos. Foi desenvolvido pela INTEL para a integração de componentes no nível mais baixo da automação industrial, integrando sensores, atuadores, controladores e instrumentos de medição.

O Bitbus é um concorrente comercial largamente difundido para os sistemas atualmente propostos para padronização do Fieldbus e apresenta, a exemplo destes sistemas, uma arquitetura de apenas três camadas.

Na camada física é utilizada a interface padrão RS-485 com par trançado e taxas de transmissão de 62.5 Kbps até 2.4 Mbps (modo síncrono, com 4 fios). No modo assíncrono, a rede requer apenas 2 fios e opera com taxas de transmissão de até 1 Mbps. Cada segmento da rede suporta no máximo 28 estações, mas através de repetidores podem ser conectados até 250 elementos na rede.

Para a camada de enlace o Bitbus emprega o protocolo SDLC (Synchronous Data Link and Control), um sub-conjunto do protocolo HDLC padronizado pela ISO. O processador Intel 8044 implementa este protocolo a nível de hardware, permitindo assim uma execução extremamente rápida do mesmo.

A nível da camada de aplicação, a INTEL definiu um conjunto de serviços denominados RAC (Remote Access and Control), especialmente concebidos para atender aplicações envolvendo sensores e atuadores, que são apresentados na tabela 5.3.

Serviço	**Função de Acesso**	**Função de Comando**
Reset_Slave		X
Create_Task		X
Delete_Task		X
Get_Function_ID		X
RAC_Protect		X
Read_IO	X	
Write_IO	X	
Update_IO	X	
Upload_Memory	X	
Download_Memory	X	
OR_IO	X	
AND_IO	X	
XOR_IO	X	
Status_Read	X	
Status_Write	X	

Tabela 5.3 - Serviços RAC do Bitbus

Em 1986 o sistema Bitbus tornou-se um padrão, sob a designação IEEE 1118. Mais de 2,5 milhões de dispositivos com interface Bitbus estão no mercado, produzidos por mais de 50 empresas diferentes.

3.5.5.3. CAN (CONTROLLER AREA NETWORK)

Uma rede do tipo fieldbus que vem ganhando atenção crescente é a rede CAN (Controller Area Network), desenvolvida originalmente pela BOSCH para integrar elementos inteligentes em veículos autônomos (eletrônica "embarcada"). Um automóvel moderno, por exemplo, pode possuir mais de 200 microprocessadores, controlando funções como carburação eletrônica, frenagem anti-bloqueante (ABS), controle e supervisão da temperatura do óleo e do radiador, pressão de óleo de freio, ajuste automático de espelhos retrovisores, banco do motorista, etc.

O sistema CAN teve sua primeira versão lançada em 1984. Em 1987 foi produzido o primeiro chip que implementa em hardware as funções de comunicação, o 82526, produzido pela INTEL. A partir de 1991 vários outros fabricantes foram licenciados para a fabricação de chips para CAN. Entre eles temos a Phillips (chips 82C200, 87C592, 82CE598 e 82C150), a Motorola (chip 68HC05), a NEC (chip 72005), além da Siemens, da Thompson e da National.

Apesar de ter sido concebido para uso em eletrônica embarcada, os grupos de trabalho da área de automação vislumbraram a adequação do sistema para uso como rede local industrial e formaram uma entidade chamada CiA (CAN in Automation), constituída de usuários e fabricantes de produtos baseados no protocolo. Até abril de 1993, a CiA já contava com 64 associados não ligados a industria automobilística.

CAN tem as seguintes características a nível da camada física (padrão ISO/DIS 11898):

- Topologia: barramento ou estrela (com concentrador);
- Taxa de transmissão: 125 Kbps ou 1 Mbps;
- Comprimento máximo do barramento: 40 m para 1 Mbps; até 1 Km para 125 Kbps;
- Número máximo de nós: 16;
- Codificação de bits: NRZ (Non Return to Zero);
- Meio de transmissão: não especificado na norma, mas usualmente usado par trançado ou fibra ótica.

A nível da subcamada MAC, temos:

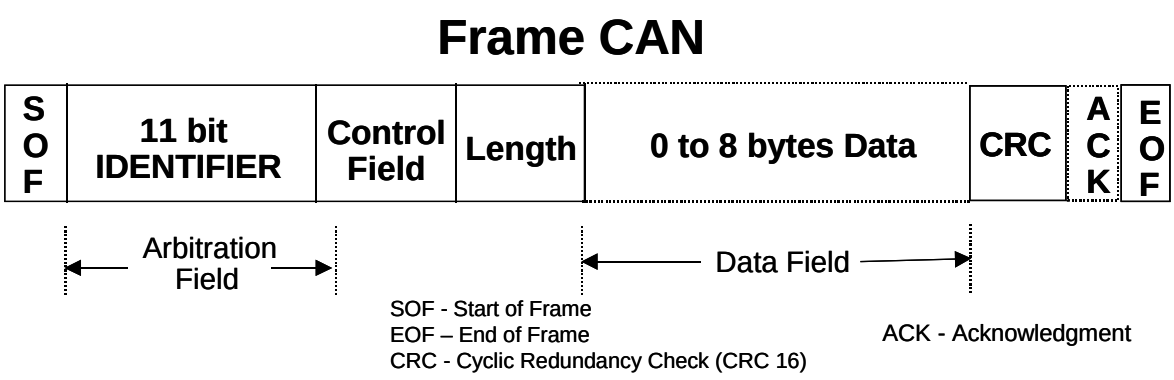
- Método de acesso ao meio: Forcing Headers (ver seção 2.2.1.3) com prioridades para mensagens.

A nível da subcamada LLC, temos:

- Comprimento máximo dos quadros de dados: 8 Bytes;
- Controle de erro por CRC (Cyclic Redundancy Check).

As camadas 3 até 6 do RM-OSI foram suprimidas, a exemplo do que ocorre com as redes tipo fieldbus.

•CSMA/NBA - **Carrier Sense Multiple access with Non-destructive Bitwise Arbitration (Forcing Headers)**



•Qualquer nó pode acessar o meio se estiver livre

•NBA garante 100% de utilização do meio e priorização de mensagens baseada no identificador de 11 bits do frame

•Como na Ethernet, cada nó tenta transmitir se meio livre.

–Diferentemente de Ethernet, não há colisões.

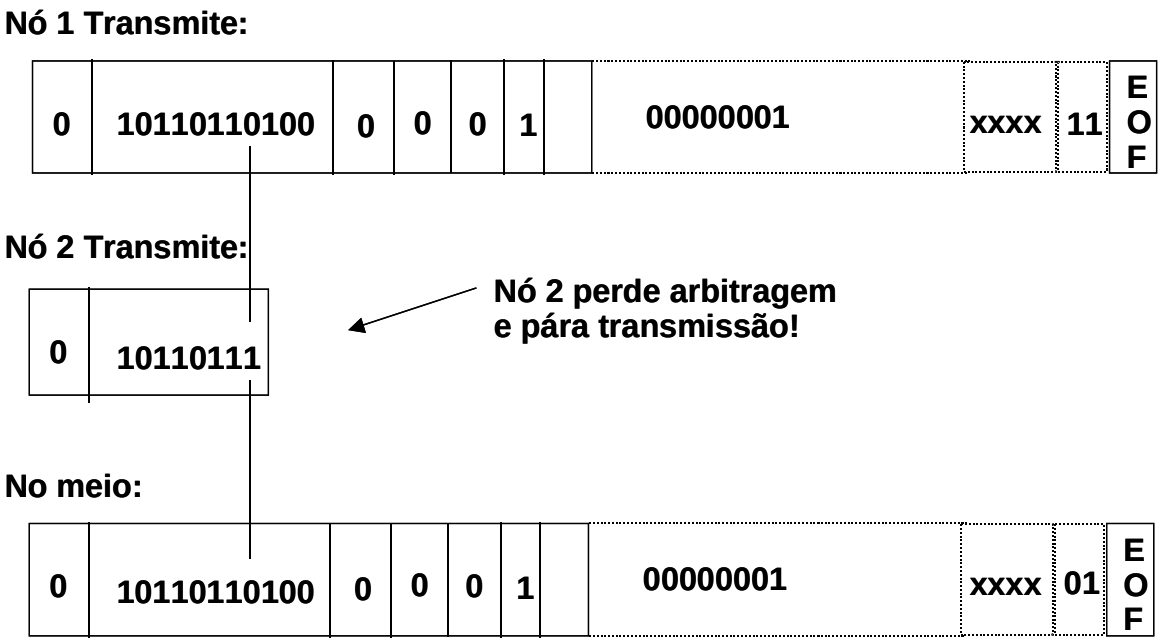
•Se 2 ou mais nós iniciam transmissão simultânea, conflito resolvido por arbitragem bit a bit usando o campo IDENTIFIER.

–"0" é dominante no fio sobre "1" (operação AND binária).

–Se um nó transmite "1", mas escuta "0", ele imediatamente pára transmissão.

–O nó vencedor envia o resto da mensagem.

–Mecanismo garante que não se perde informações nem tempo.



•O valor do campo IDENTIFIER define prioridade durante arbitragem (IDENTIFIER mais baixo "vence"). Isto significa que dois frames não podem ter o mesmo IDENTIFIER.

•Modelos de comunicação:
–Frame não contém campos específicos para endereço destino/origem.
–Campo IDENTIFIER pode conter endereço de uma estação, grupo de estações (multicasting) ou mensagens são difundidas para todas as estações (broadcasting).
–Campo IDENTIFIER pode identificar o conteúdo da mensagem (dados), que é difundida para todas as estações.
»**Gerador da mensagem: PRODUTOR.**
»**Estações interessadas no conteúdo da mensagem: CONSUMIDORES.**

CAN também não define uma especificação para a camada de Aplicação, mas o grupo CiA definiu uma especificação para aplicações em automação, composta dos seguintes elementos de serviço:

- CMS (CAN Message Services): fornece serviços de leitura e escrita de variáveis remotas e tratamento de eventos. CMS é baseado no MMS da rede MAP;
- NMT (Network Menagement): fornece serviços de inicialização e gerenciamento da rede;
- DBT (Distributor): provê uma distribuição dinâmica de nomes definidos pelo usuário para identificar as mensagens.

O sistema suporta até 2032 objetos diferentes, aos quais é associado um número de identificação único na aplicação. O tempo para leitura de dados a nível da camada de enlace é da ordem de 420 µs para o objeto de maior prioridade. CAN tornou-se norma internacional definida pela ISO em 1993 sob a designação ISO 11898.

3.5.5.4. VAN (VEHICLE AREA NETWORK)

A rede VAN (Vehicle Area Network) foi normalizada em 1990 na França pelo *Bureau de Normalisation de l'Automobile* para operar em eletrônica embarcada. A partir de 1992 passou a ser adotada pela Renault e pela Peugeot.

A rede VAN possui as seguintes propriedades a nível da camada física:

- Topologia: barramento;
- Taxa de transmissão: 100 Kbps até 250 Kbps;
- Número máximo de nós: 16;
- Comprimento máximo de barramento: 20 metros;
- Codificação de bits: Manchester ou NRZ.

Na subcamada MAC temos as seguintes características:

- Método de acesso ao meio: Forcing Headers (como CAN);
- Controle de erros: assumido pela subcamada MAC, que usa a técnica de CRC;

A subcamada LLC tem as seguintes particularidades:

- Quadro de dados: 8 bytes ou 28 bytes (versão FullVAN);

Apesar das semelhanças com a rede CAN, não se conhecem aplicações da rede VAN em automação industrial. Os chips disponíveis estão implementados na forma de ASICs projetados especificamente para a indústria automobilística.

Existem várias outras redes concebidas para aplicação em veículos, tais como J1850 (definida nos Estados Unidos pela SAE), C^2D (Chrysler Collision Detection), MIL-STD-1553B (para aviônica militar), mas todas com uso restrito à eletrônica embarcada.

3.5.5.5. DEVICENET

•DeviceNet é uma rede industrial de baixo custo para conectar dispositivos como chaves fim de curso, células fotoelétricas, válvulas, motores, drives, displays de CLP e PC, etc.

•DeviceNet foi desenvolvida tendo CAN como base.

•DeviceNet oferece manipulação robusta e eficiente de dados e é baseada na técnica produtor / consumidor.

•A ODVA (open DeviceNet Vendor Association) é uma organização independente que supervisiona e gerencia as especificações da DeviceNet.

•Seu objetivo é promover a adoção mundial de DeviceNet como rede aberta.

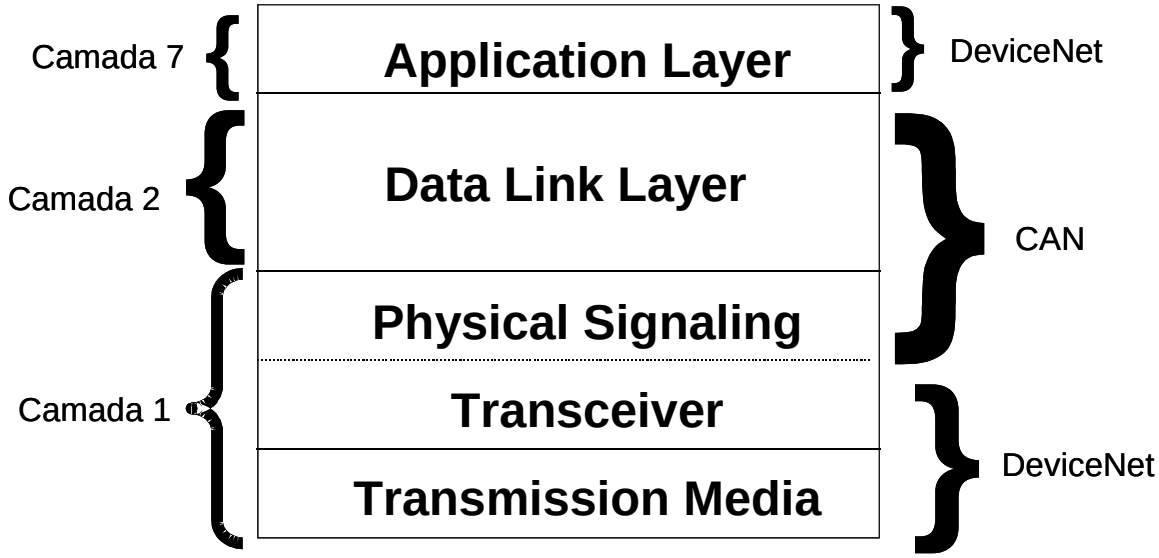
•A ODVA trabalha conjuntamente com os membros vendedores, usuários finais e distribuidores.

•Possui 320 membros (até julho de 2001).

•*Home-pages:*

–*http://www.odva.org*

–*http://www.ab.com/catalogs/b113/comm/dnet.html*



•Configuração em barra (daisy-chain ou ramificações)

•Nós podem ser removidos sem interromper linha

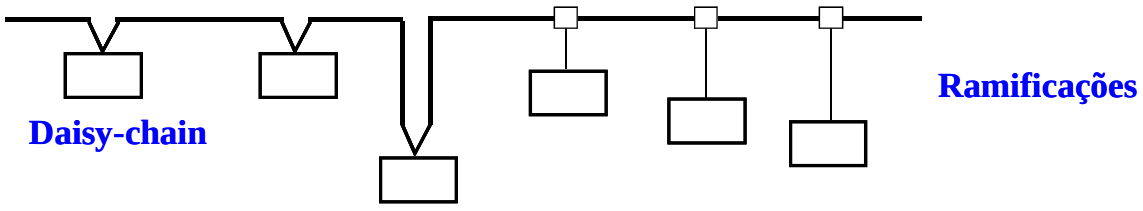
•Até 64 nós endereçáveis

•Sinal e alimentação de 24vdc no mesmo cabo

•Taxas transmissão: 125kbps, 250kbps, 500kbps

•Conectores selados ou abertos

•Terminador de 121 ohms nas extremidades



•Par trançado com dois fios:

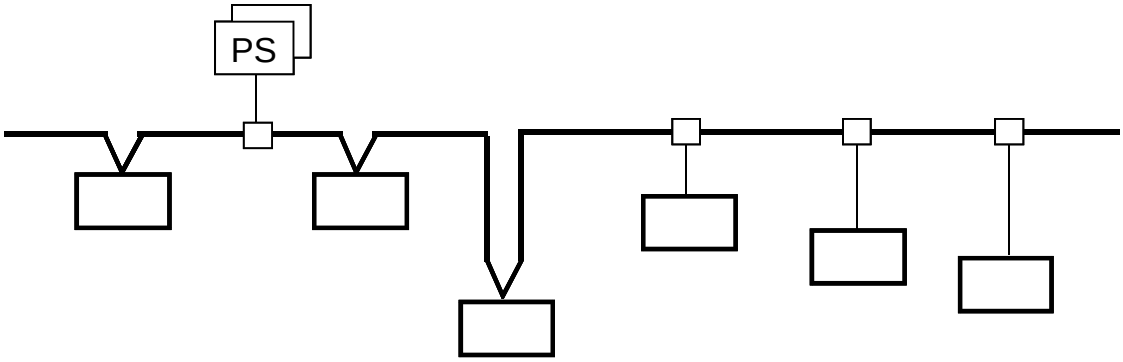
–Par Sinal: baixa perda, alta velocidade.

–Par Alimentação: até 8A corrente.

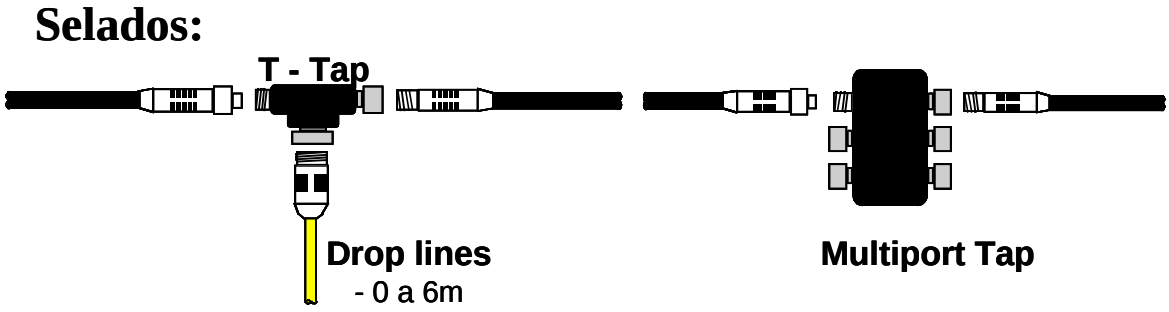
•Sensores alimentados da linha.

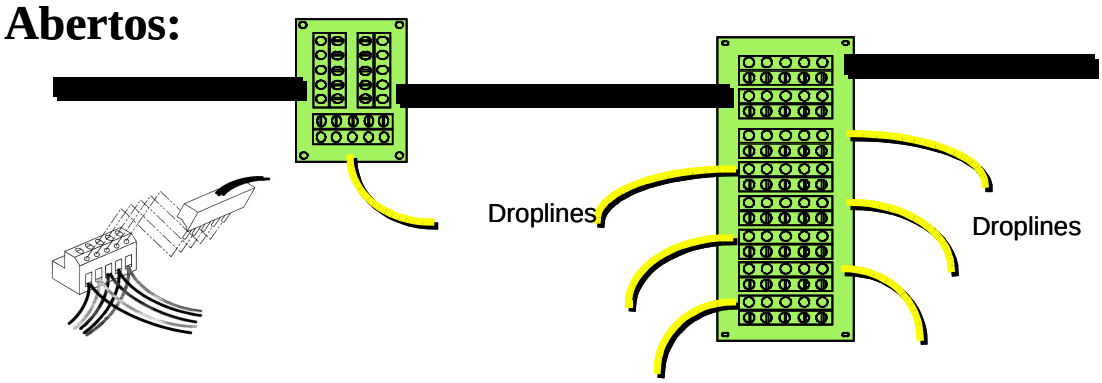
•Opto-isolamento para dispositivos com alimentação própria (Ex.: drive, PLC, etc.).

•Pode-se usar várias fontes de alimentação.



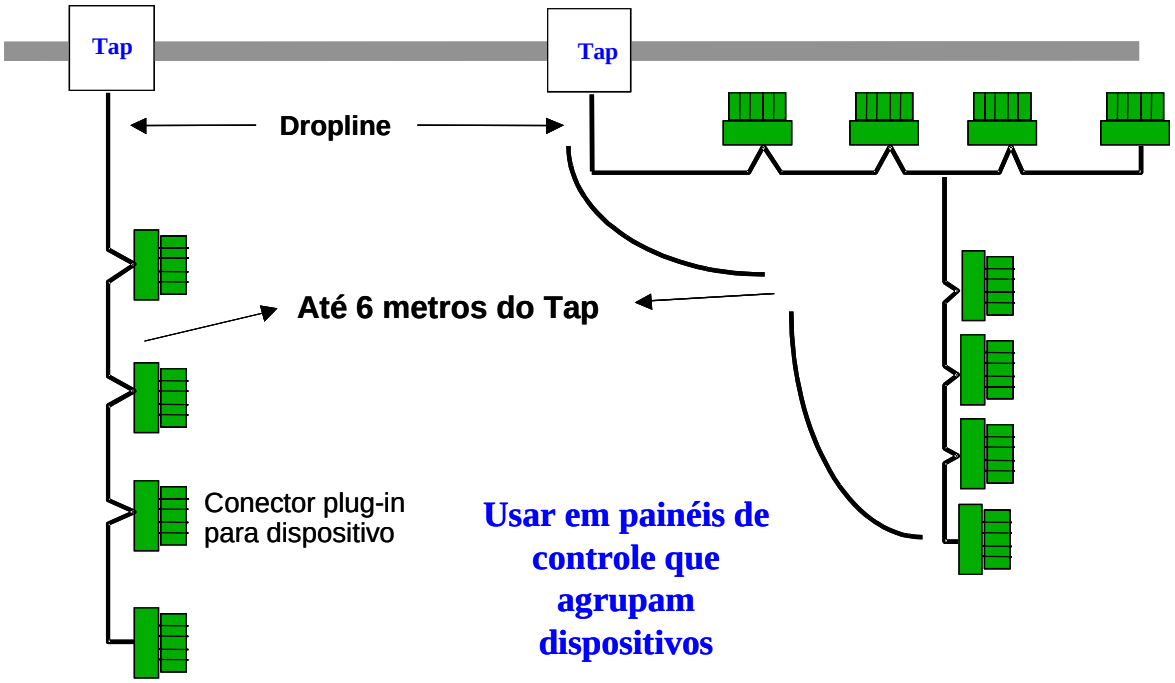
conectores abertos e selados:





Daisy chaining:

Distancias e velocidades:



Data Rate	Barramento	Ramificações	
		Dist. TAP	Cumulativo
125K	500m	26 x 6m	156m
250K	250m	13 x 6m	78m
500K	100m	6 x 6m	36m

•Enlace segue sistema **CAN**.

•Formas de comunicação suportadas através do modelo **produtor/consumidor**:

–**Master/Slave**: escravos só enviam dados em resposta a varredura do mestre.

–**Peer-to-Peer**: comunicação livre entre fontes / destinos quaisquer (par a par).

–**Multi-master**: vários mestres e vários escravos.

–**Mudança de estado dos dados**: envio de dados entre estações predefinidas sempre que houver alteração de estado.

–**Produção cíclica de dados**: estações enviam dados entre si em intervalos fixos de tempo.

•Definição do campo Identifier

–Estabelece prioridade no processo de arbitragem

–usado pelos nós receptores para identificar mensagens

•Dois tipos de mensagens

–Mensagens de I/O para dados de controle críticos no tempo

–Mensagens explicitas para funções cliente/servidor

–Fragmentação para dados maiores que 8 bytes

•Detecção de identificadores duplicados

•Verificação de consistência dos dados de aplicação

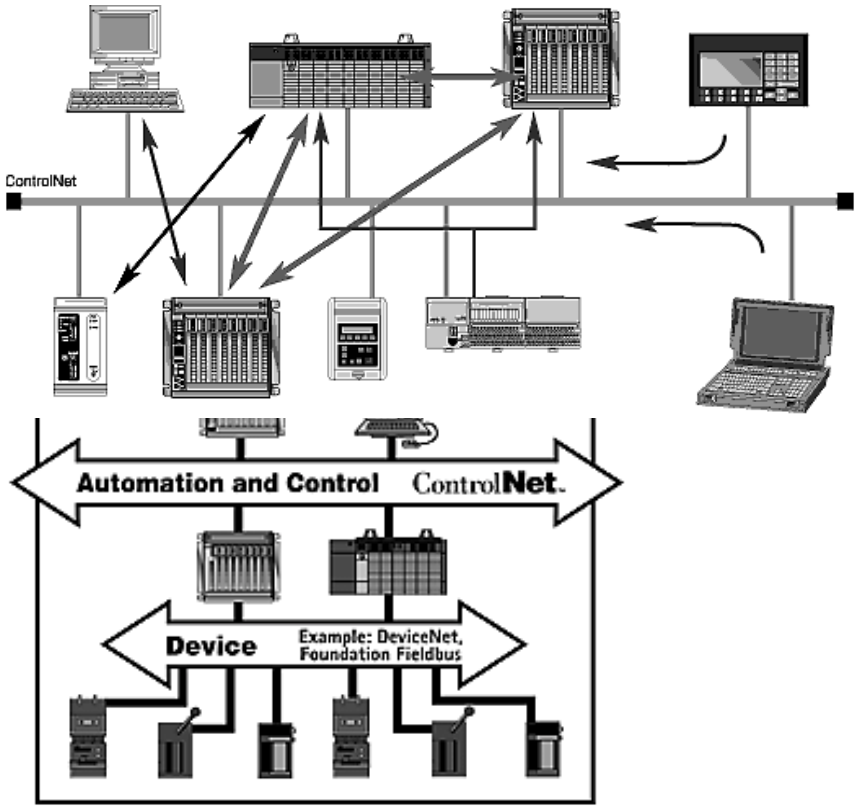
3.5.5.6. CONTROLNET

•ControlNet International é uma organização independente criada em 1997 que mantém e distribui a especificação ControlNet e gerencia is esforços de marketing dos membros associados.

•Home-page: www.controlnet.org

•Mais infos: www.ab.com/catalog/b113/comm/cnet.html

•Onde usar: níveis intermediários (célula, área)



•Camada física:

–Topologias: barramento, árvore, estrela

–Taxa transmissão: 5 Mbps

–Estações endereçáveis: até 99

–Distâncias:

»Cabo coaxial RG-6: 1.000 m com 2 nós, 500 m com 32 nós, 250 m com 48 nós (sem repetidores), máximo de 5.000 m com 5 repetidores

»Fibra: 3.000 m sem repetidores, até 30 km com 5 repetidores

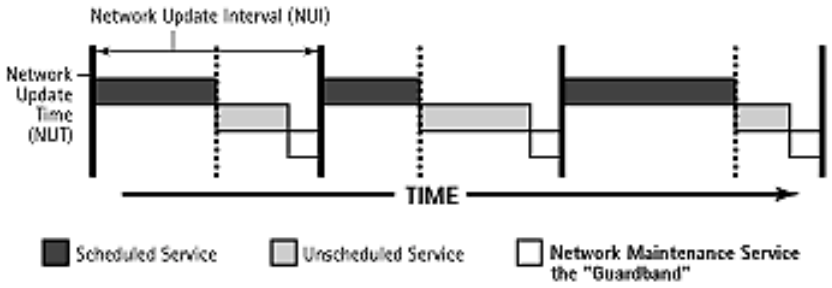
•Camada de enlace:

–Controle de erros no frame por Cyclic Redundancy Check, polinômio CCITT modificado com 16 bits.

–Campo de dados com até 510 bytes.

–MAC: CTDMA (Concurrent Time Domain Multiple Access), que regula a oportunidade de transmitir de cada nó em intervalos de tempo ajustáveis chamados NUT (Network Update Time). A menor NUT é de 2ms.

–Informações com restrições temporais são enviadas na parte escalonada da NUT. Dados sem restrições temporais (ex.: Dados de configuração) são enviados nos intervalos restantes de tempo.



Concurrent Time Domain Multiple Access

•Camada de aplicação:

–Orientação a objetos

–Modos de comunicação:

»Master/Slave

»Multi-Master

»Peer-to-Peer

»Produtor/consumidor

–Leitura de dados:

»Mudança de estado

»Cíclico

»Por solicitação

3.5.5.7. O PROTOCOLO HART

O protocolo HART (Highway Addressable Remote Transducer) é um dos protocolos mais difundidos a nível industrial para a interligação de equipamentos de campo inteligentes. Este protocolo é adotado por empresas como: Siemens, Hitachi, Toshiba, Yokogawa, ABB, Endress+Hauser, Fischer & Porter, Rosemount Inc, Camile Bauer, Smar International e muitas outras, agrupadas em torno do chamado grupo "HUG" (HART User Group).

As principais características do protocolo HART são:

- Meio físico: par trançado;

- Taxa de transmissão: 1200 bps;
- Transmissão assíncrona a nível de caracteres UART (compostos de 1 start bit, 8 bits de dados, 1 bit de paridade e 1 stop bit);
- Tempo médio para aquisição de um dado: 378,5 ms;
- Método de acesso ao meio: mestre/escravos;
- Topologia: barramento ou árvore;
- Modulação: FSK (Frequency Shift Keying, padrão Bell 202), onde o sinal lógico 1 é representado por um sinal de baixa tensão e com freqüência de 1200 Hz e o sinal lógico 0 por uma freqüência de 2200 Hz.

Devido a forma de modulação adotada, é possível transmitir simultaneamente sinais de 4 a 20 mA e quadros digitais pelo barramento. O Chip NCR 20C12 serve como modem de baixa potência para uso em equipamentos de campo. O chip requer a adição de filtros e comparadores para a operação do protocolo.

3.5.5.8. INTERBUS-S

O sistema Interbus-S foi desenvolvido na Alemanha pela empresa Phoenix Contact e obteve ampla aceitação industrial, já tendo sido empregado em mais de 5.000 aplicações de automação (em sua maioria na Europa).

O Interbus-S foi concebido para integração de sensores a atuadores a um elemento de tomada de decisão (CLP, CNC, RC, etc.). O elemento de tomada de decisão opera como uma estação mestre, ao passo que os sensores e atuadores são tratados como estações escravas, que executam essencialmente operações de entrada/saída. Como em todo método de acesso tipo mestre/escravos, a iniciativa de comunicação parte sempre do mestre. No entanto, em lugar de utilizar uma topologia em barramento, na qual o mestre executa uma varredura cíclica dos escravos por meio de quadros específicos para este fim, o Interbus-S adotou uma topologia em anel com um método de varredura denominado "Quadro Concatenado" ou "Quadro Somado" (do alemão "*Summenrahmen-Verfahren*"), que opera de forma análoga a um registrador de deslocamento. Neste método, o mestre monta um quadro único contendo campos reservados para uso de cada um dos escravos. Quando o mestre deseja enviar dados a um dos escravos, preenche o campo reservado àquele escravo com os dados de processo ou parâmetros a enviar. O quadro então é enviado ao primeiro escravo no anel. O primeiro escravo reconhece no quadro o início de sua janela de dados e assim verifica o conteúdo somente do campo reservado a ele, ignorando o resto do quadro. O escravo lê a informação contida no seu campo reservado, copiando-a para um buffer de recepção, e substitui o conteúdo do campo pelos dados de resposta, que estavam a espera em um buffer de envio. Desta forma, o mestre pode enviar novos dados para serem colocados nas saídas do escravo e receber em lugar deles os

dados atualizados das entradas do mesmo escravo. Em seguida, o primeiro escravo envia o quadro completo (agora com seus dados locais no campo reservado) para o próximo escravo no anel. O processo se repete até que o quadro tenha percorrido todos os escravos do anel e retornado ao mestre. A comunicação se dá, portanto, sempre entre o mestre e os escravos. Este princípio é ilustrado na figura 5.4.

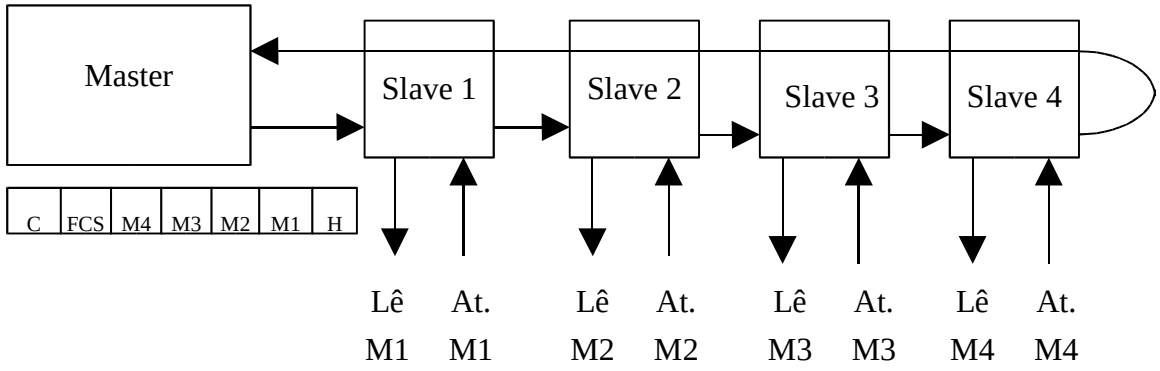


Figura 5.4 - Varredura dos escravos no INTERBUS-S

Para melhor entender o método, pode-se fazer uma analogia com um trem que pára em diversas estações, deixando alguns passageiros e pegando outros. Aqui, os passageiros que desejarem desembarcar do trem (quadro somado) na estação 1 (escravo 1) devem ficar em um vagão reservado para passageiros destinados a esta estação (o campo do quadro somado reservado ao escravo 1). Da mesma forma, todos os passageiros que embarcarem no trem na estação 1 terão que entrar no vagão reservado àquela estação. Terminado o desembarque dos passageiros destinados à estação 1 e o embarque dos passageiros que querem retornar a estação central (mestre), o trem avança até a estação seguinte, até completar uma volta completa (anel).

O tempo que o quadro somado leva para percorrer o anel (tempo de um ciclo de varredura) depende do número de estações escravas e é determinístico. O número máximo de entradas e saídas suportadas pelo Interbus-S é de 2048, que podem ser varridas em 7.2 ms. A distância entre estações consecutivas no anel pode chegar até 400 metros. O número máximo de estações é 256, de modo que o anel do INTERBUS-S pode chegar a ocupar 13 Km sem repetidores. A taxa de transmissão é de 500 Kbps.

As informações que o mestre envia para os escravos podem ser dados de processo, tais como comandos a executar ou valores a colocar em uma saída, ou parâmetros de configuração do escravo. Os dados de processo são em geral sujeitos à restrições temporais (operação em tempo real), ao passo que os parâmetros de configuração em geral só são necessários no inicio de operação, ou em alguns instantes de tempo posteriores, não sendo em geral sujeitos à restrições temporais. Os parâmetros de configuração são enviados em time slots reservados para este fim no quadro somado.

A nível da camada de aplicação, o Interbus-S define um conjunto de serviços denominado PMS (Peripherals Message Services), que consiste de um subconjunto dos serviços do MMS da rede MAP. Os serviços PMS incluem:

- gerenciamento de conexões;
- identificação e verificação de status;
- gerenciamento de objetos;
- acesso a variáveis (read, write, update, etc.);
- gerenciamento de programas (dowload, upload, start, stop, resume, etc.).

Grupos de empresas que oferecem produtos compatíveis com Interbus-S (hoje mais de 100) ou que são usuárias do produto se reuniram em organizações como a DRIVECOM e a ENCOM, ocupadas de definir padrões de utilização e configuração para este sistema. O sistema é também tema de atividades de padronização pela IEC e DIN.

3.5.5.9. ASI-BUS

O sistema ASI (Actuator/Sensor Interface) foi desenvolvido por um consórcio de 11 empresas (Balluf, Baumer, Elesta, Festo, IFM, Peperl+Fuchs, Sick, Siemens, Leuze, Turck e Visolux) e introduzido no mercado em 1993.

ASI foi concebido para interligar via rede elementos periféricos (sensores e atuadores) binários, tais como chaves fim-de-curso, sensores de proximidade indutivos e capacitivos, relês, válvulas, etc. Todos estes elementos requerem em geral uma informação mínima para operar (na maioria dos casos, 1 bit com comandos tipo ON/OFF).

Para atender aos requisitos operacionais deste tipo de dispositivo, ASI foi concebido como um sistema Mestre/Escravos com topologia em barramento. O mestre executa uma varredura cíclica dos escravos, enviando quadros de solicitação de dados e aguardando um quadro de resposta.

Os quadros enviados pelo mestre ASI tem um campo de dados de apenas 4 bits e um campo de parâmetros de mais 4 bits, conforme a figura 5.5. O quadro tem 17 bits no total.

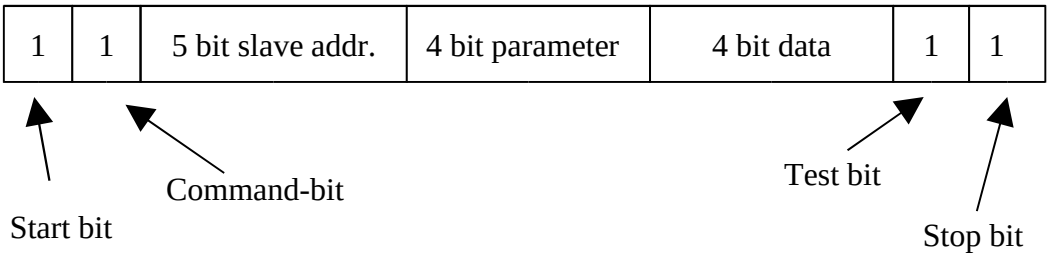


Figura 5.5 - Quadro do mestre ASI

O quadro de resposta do escravo tem o formato da figura 5.6, composto de apenas 7 bits. Como todas as respostas são destinadas ao mestre, não é necessário um campo de endereço neste quadro.

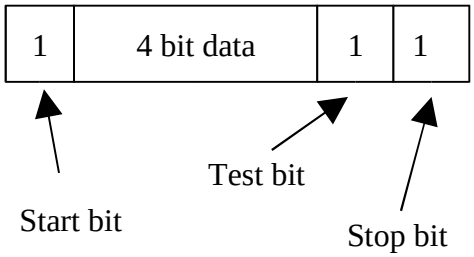


Figura 5.6 - Quadro dos escravos ASI

Como os quadros utilizados são sempre iguais aos acima mostrados, a varredura de cada escravo implica no envio e recepção de um total de apenas 24 bits, o que pode ser feito em um tempo bastante reduzido, como veremos adiante.

Cada escravo recebe portanto 4 bits de dados e 4 bits de parâmetros, e responde, se for o caso, também com 4 bits de dados. Um escravo ASI possui até 4 portas de I/O conectadas a dispositivos periféricos, como mostra a figura 5.7. Cada porta de saída recebe o valor de 1 dos 4 bits do campo de dados do quadro enviado pelo mestre. Se as portas estão configuradas como entradas, seu valor é copiado nos 4 bits correspondentes do campo de dados do quadro de resposta do escravo. Desta forma, o mestre pode ler ou escrever em qualquer uma das portas remotas dos escravos.

ASI suporta até 31 escravos em um barramento. Como cada escravo pode ter 4 entradas ou saídas, o número máximo de elementos binários que podem ser integrados aos 31 escravos é de 124. Esta configuração permite ligar os sensores e atuadores binários convencionais atuais à rede ASI. A varredura completa dos 31escravos, atualizando todas as 124 entradas e saídas, requer cerca de 5 ms.

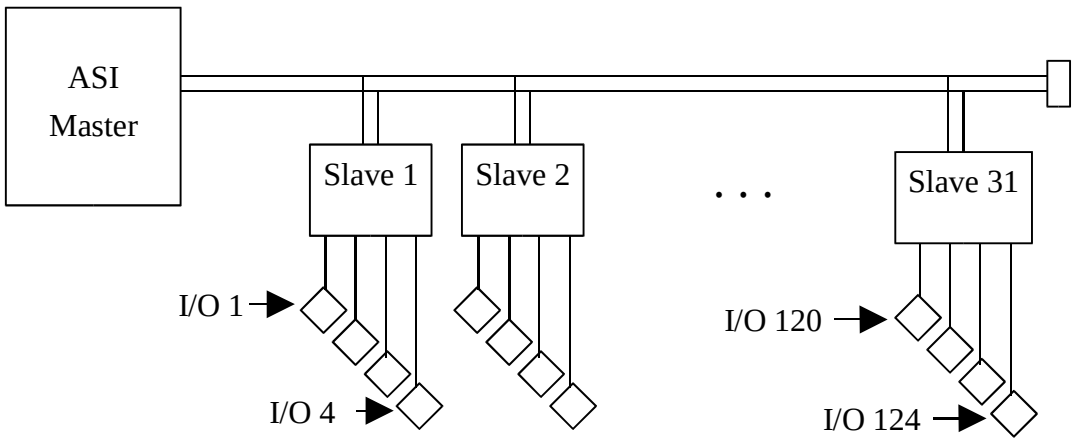


Figura 5.7 - Configuração típica de um segmento ASI

Os 4 bits de parâmetros recebidos do mestre podem ser enviados para 4 portas de saída adicionais, podendo ser utilizados para configurar um dispositivo mais sofisticado conectado ao escravo, como ilustrado na figura 5.8. Esta configuração permite conectar sensores e atuadores inteligentes à rede ASI.

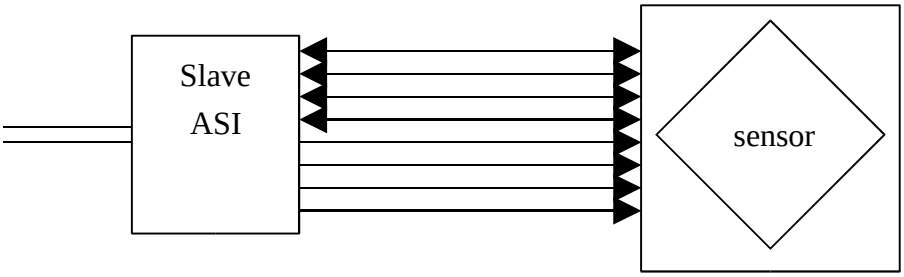


Figura 5.8 - Sensor ou Atuador inteligente conectado a um escravo ASI

O cabo de rede ASI é composto de 2 condutores não blindados e é utilizado também para a alimentação dos escravos (24V DC, 100 mA por escravo). Um segmento de rede ASI pode ter até 100 metros de comprimento.

A grande vantagem de ASI sobre outras rede tipo fieldbus é o custo baixo e simplicidade de implementação, operação e manutenção. Sua aplicação em automação industrial vem crescendo muito desde seu lançamento em 1993.

3.5.5.10. FAIS

A rede FAIS (Factory Automation Interconnection System) foi desenvolvida no Japão por um consórcio de mais de 30 empresas e o International Robotics and Factory Automation Center (IROFA). Os primeiros produtos foram lançados no mercado em 1992.

FAIS consiste basicamente em uma versão atualizada da rede Mini-MAP e foi concebida para uso em automação fabril no nível hierárquico de célula (FMC). Neste nível encontram-se subsistemas como máquinas ferramenta com CNC, Robôs, Sistemas de transporte, CLPs, etc.

A arquitetura FAIS também é composta das camadas 1, 2 e 7, com especificações próximas as do mini-MAP, mas com algumas alterações, como veremos a seguir.

Na camada física está previsto o uso de cabo coaxial com técnica de transmissão em Carrier-Band com 5 ou 10 Mbps ou, opcionalmente, fibra ótica com 10 Mbps.

A camada de enlace de dados prescreve para a subcamada MAC o protocolo Token-Bus, conforme IEEE 802.4. Na subcamada LLC é especificado o serviço sem conexão com reconhecimento (LLC tipo 3), conforme IEEE 802.2.

A camada de aplicação prevê o uso de MMS (Manufacturing Message Sevices) juntamente com serviços de gerenciamento de rede NM (Network Menagement) e dicionário de objetos OD (Object Dicionary).

As alterações básicas em relação a proposta mini-MAP estão portanto na camada física com fibra ótica e nos serviços de aplicação NM e OD.

A figura 5.9 mostra resumidamente a especificação FAIS.

<table>
<tr><td>Aplicação</td><td>MMS</td><td>NM</td><td>OD</td></tr>
<tr><td>Apresentação</td><td colspan="3" rowspan="4">VAZIO</td></tr>
<tr><td>Sessão</td></tr>
<tr><td>Transporte</td></tr>
<tr><td>Rede</td></tr>
<tr><td>Enlace</td><td colspan="3">LLC 802.2 tipo 3
MAC 802.4 Token bus</td></tr>
<tr><td>Física</td><td>Baseband
5 / 10 Mbps</td><td colspan="2">Fibra Ótica
10 Mbps</td></tr>
</table>

Figura 5.9 - Especificação FAIS 2.0

A inclusão do sistema FAIS na proposta MAP/TOP está atualmente em discussão. Algumas incompatibilidades a nível da camada de aplicação estão ainda atrasando a integração das propostas.

3.5.5.11. LON

A rede LON (Local Operating Network) foi desenvolvida pela empresa Echolon em 1990 para atender a um amplo espectro de aplicações, incluindo automação predial (imótica), automação doméstica (domótica), automação de escritórios e mesmo automação industrial.

O protocolo de comunicação da rede LON é conhecido pelo nome LonTalk, que é implementado em firmware em um processador dedicado denominado NeuronChip, produzido pelas empresas Motorola e Toshiba. O NeuronChip é composto dos seguintes elementos:

- Três processadores de 8 bits, dos quais um executa o controle de acesso ao meio, outro é responsável por serviços gerais de comunicação e o terceiro é dedicado a aplicações do usuário em um nó de rede;
- Porta de conexão ao transceiver, através do qual o NeuronChip se conecta ao meio;
- Pinos de entrada e saída, reset, clock e alimentação (5V);
- Acesso a um número de série de 48 bits definido pelo fabricante;

- Um timer programável;
- Sistema completo de memória, contendo 10 Kbyte ROM, 1 Kbyte RAM e 512 Bytes EEPROM para parâmetros de rede;
- 3 temporizadores tipo Watch-Dog (1 para cada processador);

O protocolo LonTalk é baseado no modelo de referência OSI e implementa todas as 7 camadas previstas pela ISO.

Para implementar uma rede LON, necessita-se de um conjunto de ferramentas de software e componentes denominado LonWorks, que incluem:

- NeuronChip;
- Protocolo LonTalk;
- Transceivers que permitem ligar o NeuronChip ao meio físico;
- LonBuilder Developer's Workbench, que é um sistema de desenvolvimento orientado a objetos para projeto, implementação e teste de nós LON.

A nível da camada física, existem transceivers para vários meios de comunicação e diferentes topologias de rede. Os modelos oferecidos pela Echolon são os seguintes:

- FTT-10: meio tipo par trançado, taxa de transmissão de 78 Kbps, suporta 127 nós em um barramento de até 2.700 metros ou segmentos de até 500 metros com topologia em estrela ou anel;
- LPT-10: meio tipo par trançado, taxa de transmissão de 78 Kbps, suporta opcionalmente 32 nós com consumo de 100 mA cada, 64 nós com 50 mA cada ou 128 nós com 25 mA cada, em um barramento de 2.200 metros ou segmentos de até 500 metros com topologia em estrela ou anel. Recebe alimentação pelo mesmo fio em que recebe/envia dados;
- TPT/XF-78: par trançado, taxa de 78 Kbps, barramento com 2.000 metros, 64 nós;
- TPT/XF-1250: semelhante ao anterior, mas com taxa de transmissão de 1.25 Mbps para distâncias de até 500 metros;
- PLT-10A: utiliza como meio físico a própria rede elétrica da casa ou prédio por meio da tecnologia *spread spectrum* (técnica especial de transmissão usada em sistemas com elevados níveis de interferência), operando na faixa de freqüência de 100 KHz até 450 KHz com taxa de transmissão de 10 Kbps;
- PLT-20: idem ao anterior, mas com freqüência de 125 KHz a 140 KHz com taxa de transmissão de 5.4 Kbps;
- PLT-30: idem aos anteriores, mas com freqüência de 9 a 95 KHz e taxa de 2 Kbps.

Além destes, existem ainda transceivers de outros fabricantes, incluindo suporte para:

- RF-300: usa sinais de rádio freqüência de 300MHz, taxa de transmissão de 1.200 bps (rede sem fio);
- RF-450: idem, com 450 MHz e taxa de 4800 bps;
- RF-900: idem, com 900 MHz e taxa de 39 Kbps;
- IR: usa sinais em infravermelho, com taxa de transmissão de 78 Kbps;
- Fibra ótica: taxa de transmissão de 1.25 Mbps;
- Cabo coaxial: taxa de transmissão de 1.25 Mbps.

Para que a rede elétrica de um prédio, fábrica ou residência possa ser configurada como um barramento de rede LON, são usados capacitores para interligar segmentos diferentes de fio, resultando em curto-circuitos somente para sinais de alta freqüência, como os usados para transmitir dados via rede, mas representando linha aberta para o sinal de 60 Hz da rede de energia.

A nível da camada de enlace, subcamada MAC, é utilizado um protocolo de acesso ao meio CSMA preditivo p-persistente com detecção de colisão e atribuição de prioridades às mensagens. Este protocolo é preditivo, ou seja, prevê o tráfego na rede, somente quando é usado serviço com reconhecimento. A subcamada LLC suporta somente serviços sem conexão (com ou sem reconhecimento) e oferece funções de montagem de quadros e checagem de erros com CRC.

LonWorks oferece ainda uma série de elementos para interconexão de subredes LON, incluindo roteadores (por exemplo, RTR-10) e pontes. Uma rede LON pode assim ser composta de várias subredes, com meios físicos diversos, conforme a figura 5.10.

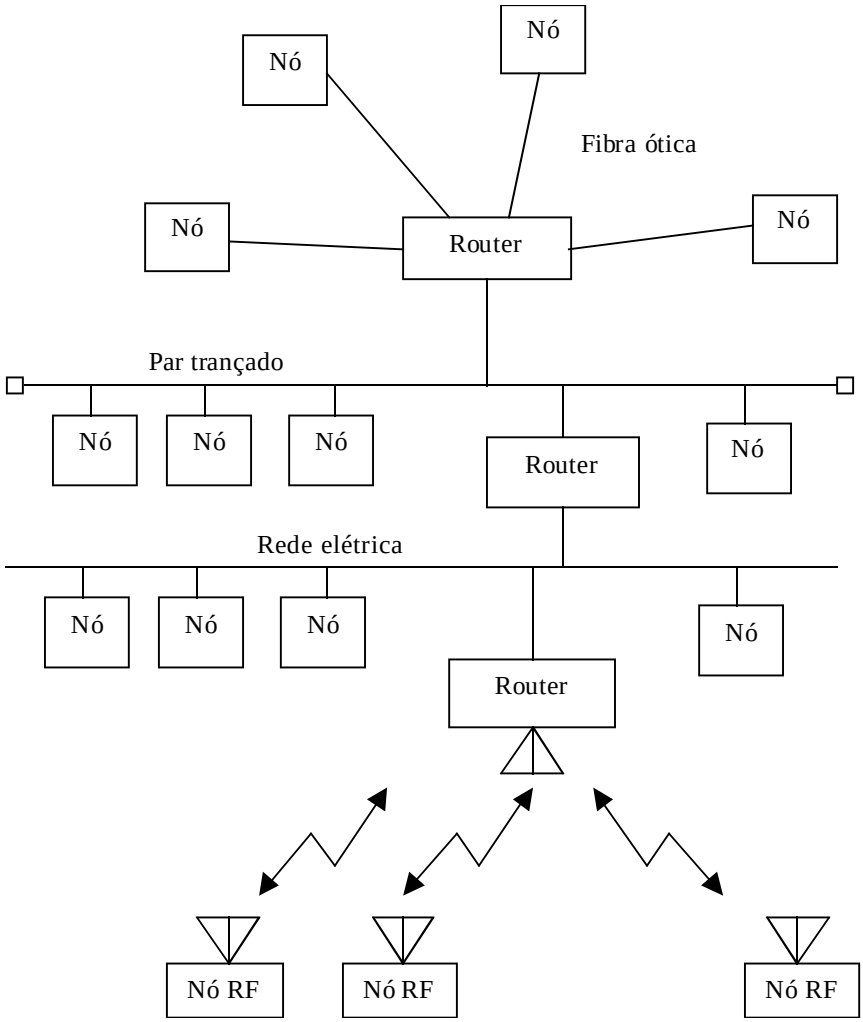


Figura 5.10 - Rede LON com subredes diferentes e até 32.385 nós

O NeuronChip é programado em uma extensão da linguagem C desenvolvida especificamente para ele, chamada Neuron C, que expande o padrão ANSI com orientação a objetos, suporte a programação concorrente, 37 novos tipos de dados definidos na especificação SNVT (Standard Network Variable Types) e mecanismos de passagem de mensagem.

Apesar deste sistema ser ainda pouco conhecido no Brasil, existem mais de 1 milhão de nós de rede LON instalados nos Estados Unidos. Em 1994 e empresa Echolon criou um grupo um grupo de usuários e fabricantes de produtos baseados no NeuronChip chamado "LonMark Interoperability Association", que inclui empresas como a Honeywell, Detriot Edison, IBM, Microsoft e Leviton. Esta associação executa testes e certificação de conformidade para produtos que queiram ter o logotipo LonMark e define diretivas para interoperabilidade.

A maioria dos nós LON instalados estão em aplicações de automação predial e residencial. Existem estações baseadas no NeuronChip para controle de lâmpadas e eletrodomésticos, termostatos, sistemas HVAC (*Heating, Ventilation and Air Conditioning*,

ou calefação, ventilação e ar condicionado), sensores de presença e segurança em geral, sensores de luminosidade ambiente, equipamentos de áudio e vídeo (por exemplo, Home Theaters), gerenciamento de energia, controle otimizado de elevadores, subsistemas de água e gás (válvulas, sensores de nível e outros componentes), etc.

3.5.5.12. P-NET

A rede P-NET foi desenvolvida na Dinamarca pela empresa Ultrakust e tem como aplicação alvo a automação industrial.

A nível da camada física, P-NET prevê uma topologia em anel, com taxa de transmissão de 76.8 Kbps. Em um anel podem estar no máximo 125 estações. Como meio físico é usado um cabo tipo par trançado blindado, com até 1.200 metros de comprimento, sem a necessidade de repeaters.

A nível da subcamada MAC é previsto um método de acesso ao meio tipo Mutimestre / Escravos. Em um anel podem estar até 32 estações mestras. Entre as estações mestras e escravas é realizada uma varredura cíclica através de quadro pré-definidos. A varredura de cada escravo requer 30 slot times, ou cerca de 390µs. Entre as estações mestras, o controle de acesso ao meio é do tipo token-passing, porém de forma diferente da adotada no método token-ring, pois o token não fica em circulação. Cada mestre pode reter o token por um tempo determinado, após o que tem que envia-lo ao próximo mestre do anel. A passagem de token entre mestres requer no máximo 10 slot times, ou cerca de 130µs. Apesar do token passar pelas estações escravas, uma vez que elas estão também no anel, estas não podem retê-lo. Esta configuração é mostrada na figura 5.11.

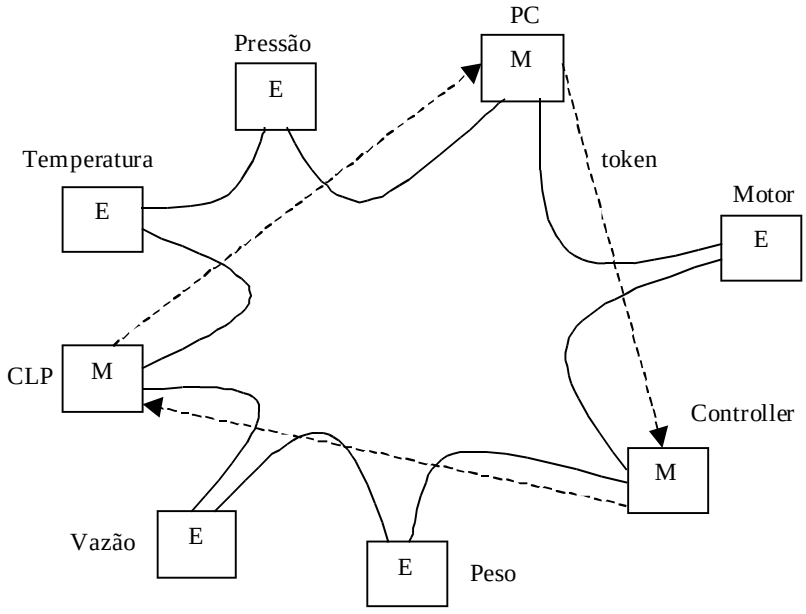


Figura 5.11 - Um anel P-NET

Vários anéis podem ser interligados entre si por meio de estações do tipo P-NET-Controller, que executam a função de roteadores ou gateways, como ilustrado na figura 5.12.

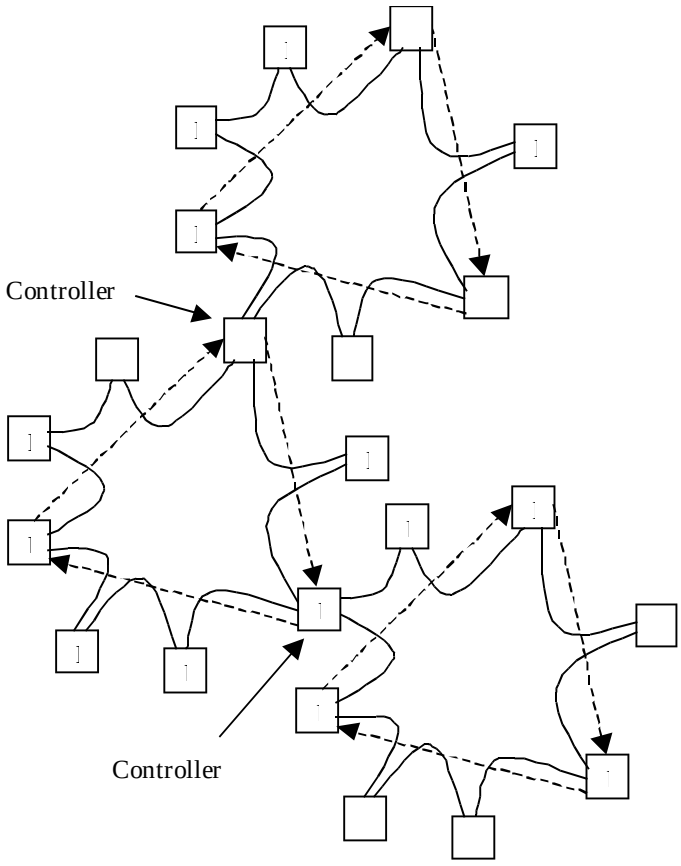


Figura 5.12 - Vários anéis P-NET interligados por Controllers

O P-NET-Controller pode também ser usado para conectar a uma rede P-NET dispositivos não especificamente desenvolvidos para este tipo de rede, mas que possuam uma interface RS-232C, ou ainda estações de outro tipo de rede (por exemplo, Profibus) a P-NET. O Controller é programado em Process Pascal, uma versão modificada da linguagem Pascal que suporta programação concorrente e primitivas de comunicação.

Diversos sistemas baseados em P-NET estão em operação na Europa. A exemplo do que foi feito para outros sistemas, foi criada para a P-NET uma organização de fabricantes e usuários que dão suporte ao produto, denominada "International P-NET User Organization".

3.5.5.13. SERCOS

A rede SERCOS (SErial Real-time COmmunication System) foi apresentada ao mercado na feira de máquinas-ferramenta EMO de 1989. A idéia inicial da rede SERCOS era realizar via rede a conexão de servo-acionamentos a um CNC em máquinas operatrizes, implementando desta forma malhas fechadas de controle.

Uma vez que no interior de uma máquina-ferramenta existem campos eletromagnéticos fortes, gerados principalmente pelos acionamentos de corrente alternada com comando tiristorizado, foi necessário definir um sistema de comunicação pouco sensível a perturbações eletromagnéticas. Para tal, foi proposta uma rede com topologia em anel utilizando como meio físico a fibra ótica, como ilustrado na figura 5.13.

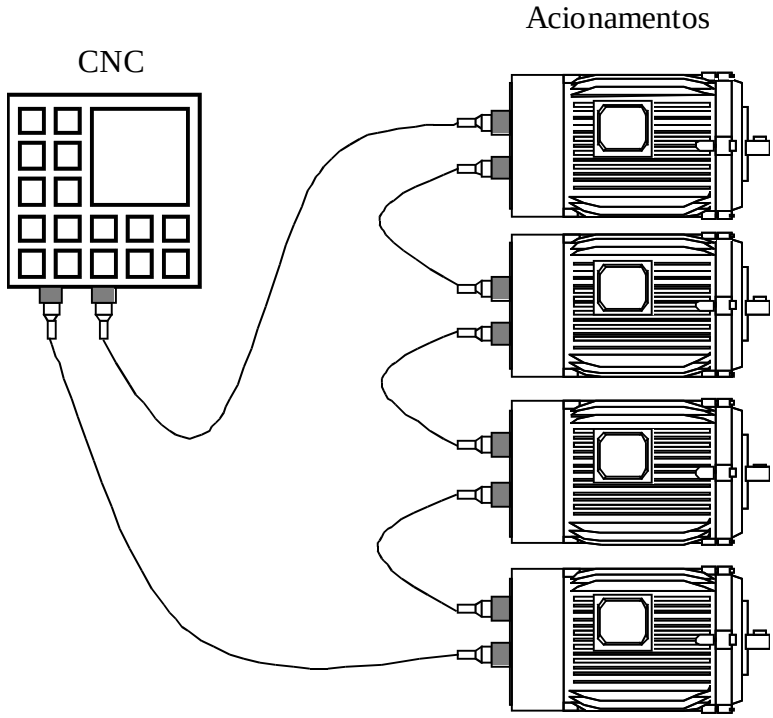


Figura 5.13 - Anel SERCOS

O sistema tem uma estrutura com comando centralizado tipo mestre/escravos, onde o CNC exerce o papel de estação mestre e os servo-acionamentos o papel de estações escravas. Como a fibra ótica suporta elevadas taxas de transmissão e os quadros SERCOS são pequenos, podem ser executados ciclos de varredura de todos os escravos em tempos ajustáveis de 62µs, 125µs, 250µs, 500µs, 1 ms ou múltiplos de 1 ms até o limite de 65 ms, permitindo assim implementar tempo de amostragem diferentes para as malhas de controle. Vale lembrar que não adianta configurar um tempo de varredura menor que aquele que o CNC é capaz de realizar.

SERCOS permite a interligação de até 254 escravos em um anel. O comprimento do cabo é de no máximo 40 metros para fibra ótica plástica e de até 1000 metros para fibra ótica de vidro.

Nas aplicações de controle de máquinas, usualmente o CNC executa o controle de posição enquanto o controle de velocidade e de corrente é executado no próprio acionamento.

Sendo assim, a rede é usada principalmente para enviar valores de referência de velocidade do CNC aos acionamentos e receber valores atualizados dos mesmos.

Na camada de enlace, SERCOS faz distinção entre dois tipos de dados:

- Dados cíclicos, com características de tempo real, usados para controle em malha fechada;
- Dados de serviço, usados para configuração, envio de parâmetros, etc.

A arquitetura da rede é diferente para cada um dos dois tipos de dados acima referidos. Para dados de serviço é usada uma pilha com 3 camadas (física, enlace e aplicação), como é usual nas redes tipo fieldbus. Para os dados cíclicos é ainda incluída uma camada de sincronização abaixo da de aplicação, cuja função básica é realizar a sincronização dos timers de todas as estações, de modo a permitir a aquisição de dados em instantes de tempo idênticos em todas as estações e a aplicação de novos valores de referência para as malhas de controle também de forma sincronizada. A subcamada LLC usa um serviço sem conexão e sem reconhecimento, de forma que quadros errados não são retransmitidos.

A camada de aplicação é única para ambos os tipos de dados e composta de serviços tipo leitura e escrita de variáveis remotas (READ/WRITE).

SERCOS vem sendo utilizada também para interligar dispositivos em outras aplicações além das máquinas-ferramenta. Entre as aplicações mais usuais estão o controle de eixos de robôs industriais e conexão de sensores e atuadores binários.

3.5.5.14. MODBUS

•O protocolo MODBUS® foi criado em 1978 pela Modicon (hoje Schneider Automation).

•O protocolo visava originalmente implementar uma maneira simples de transferir dados entre controladores, sensores e atuadores usando uma porta RS232 (serial convencional).

•Após sua criação, tornou-se padrão industrial “de-facto” adotado por muitas empresas com uma segunda opção para intercâmbio de dados.

•MODBUS® é um protocolo proprietário da Schneider Automation. No entanto, a Schneider Automation optou por uma licença sem royalties e as especificações do protocolo estão disponíveis em seu web-site gratuitamente.

•**Home-Page: http://modbus.org/**

•MODBUS® é uma estrutura de troca de mensagens usada para comunicação tipo mestre/escravos entre dispositivos inteligentes.

•Como o protocolo MODBUS é somente uma estrutura de troca de mensagens, ele é independente da camada física subjacente.

•MODBUS é usualmente implementado usando RS232, RS422, ou RS485 sobre uma variedade de meios de transmissão (fibra, rádio, celular, etc.).

•Algumas variantes do protocolo original foram criadas posteriormente.

–MODBUS PLUS: é um protocolo de maior velocidade baseado em token passing e que usa a estrutura de mensagens do MODBUS original. Os chips MODBUS PLUS são disponibilizados pela Schneider Automation através de um programa chamado MODCONNECT.

–MODBUS TCP/IP: usa TCP/IP e Ethernet para transportar a estrutura de mensagens MODBUS. MODBUS/TCP requer uma licença, mas as especificações são de acesso público e não há royalties. MODBUS TCP está disponível na página: http://www.modicon.com/openmbus.

–

–

•MODBUS suporta dois modos de transmissão:

•

•ASCII: cada byte da mensagem é enviado como 2 caracteres ASCII.

•

•RTU: cada byte da mensagem é enviado como 2 caracteres hexadecimais de 4 bits.

ADDRESS	FUNCTION	DATA	CHECKSUM

•**Address**: contém 2 caracteres ASCII ou 8 bits RTU. Endereços válidos de escravos estão na faixa de 0 a 247 decimal. Endereços individuais estão na faixa de 1 a 247 (0 para broadcasting).

•**Function**: contém 2 caracteres (ASCII) ou 8 bits (RTU). Códigos válidos vão de 1 a 255 decimal. Este campo indica ao escravo que ação este deve executar. Exemplos: ler grupo de entradas; ler dados de um grupo de registradores; ler status do escravo para diagnóstico; escrever em um grupo de saídas ou registros; permitir carregamento, gravação ou verificação do programa no escravo. Quando o escravo responde ao mestre, este campo indica se a operação ocorreu sem erros (ecoa dado recebido) ou se é uma resposta de exceção (ecoa dado recebido com Msb em 1).

•**Data**: usa 2 dígitos hexadecimais na faixa de 00 a FFh. Estes podem ser 2 caracteres ASCII ou um RTU. Contém dados adicionais para uso do escravo (endereços de portas de I/O ou registros, quantidades de itens a manipular, etc.). Se não houverem erros, este campo retorna o valor solicitado ao escravo. Se houver erro, este campo retorna um código de exceção. Este campo pode ser vazio.

•**Checksum**: são usados 2 tipos de checagem de erros (LRC ou CRC), dependendo do modo de transmissão (ASCII ou RTU)

•Mais detalhes: MODBUS protocol guide na página http://www.modicon.com/techpubs/toc7.html.

3.5.5.15. REDES IBM

A IBM oferece uma série de soluções para a interconexão de equipamentos de chão de fábrica, incluindo:

- redes baseadas em uma arquitetura própria denominada SNA (Systems Network Architecture), anterior ao modelo RM-OSI da ISO. A arquitetura SNA é mostrada na figura 5.14;
- redes compatíveis com o modelo OSI, tais como MAP;
- rede Token-Ring (IEEE 802.5);
- rede Token-Bus (IEEE 802.4);
- diversos softwares para rede (NetBios, PC-LAN, LAN-Server, etc).

Figura 5.14 - Arquitetura SNA comparada com RM-OSI

3.5.6. Conclusão e discussões

Nos últimos anos, uma grande variedade de produtos para redes de comunicação foi introduzida no mercado, tornando muito difícil manter uma visão geral atualizada do que é oferecido no setor. Os primeiros produtos comerciais na área de redes foram desenvolvidos para aplicações em automação de escritório.

Há atualmente um consenso sobre a necessidade de definir sistemas de comunicação padronizados, que permitam a interoperabilidade e eventualmente até e intercambiabilidade de equipamentos de diferentes fabricantes.

BIBLIOGRAFIA

[1]Tanenbaum, A. S. : "*Computer Networks*". 3rd Edition, Prentice-Hall, 1996.

[1]Halsall, F. : "*Data Communications, Computer Networks and OSI*". Addison Wesley, 1988.

[2]Soares, L.F.G.; Lemos, G.; Colcher, S. : "*Redes de Computadores: das LANs, MANs e WANs às redes ATM*". Editora Campus, 2ª. Edição, 1995.

[3]Electronics Industries Association (EIA) : "*EIA-RS511 - Manufacturing Message Specification*". Draft 6, May 1987.

[4]Glass, B. : "*Understanding NetBios*". Byte, pp. 301-306, January 1989.

[5]Graube, M. : "*The Carrier-Band Network and Mini-MAP: Low Cost Solution*". Control Engineering, pp. 30-31, October 1986.

[6]IEEE : "*Project 802, Local Networks Standards*". IEEE, New York, 1983.

[7]IBM : "*MAP - Manufacturing Automation Protocol*". IBM, 1990.

[8]ISO : "*Information Processing Systems - Open Systems Interconection - Basic Reference Model*". ISO / TC97 / SC16, 1983.

[9]Leite, J. R. E. : "*O Modelo de Referencia para a Interconexão de Sistemas Abertos*". Telebrás, pp. 11-15, Junho 1985.

[10]MAP/TOP : "*Manufacturing Automation Protocol / Technical Office Protocol, Version 3.0*". Users Group of SME, July 1987.

[11]Mendes, M. J. : "*Comunicação Fabril e o Projeto MAP/TOP*". Editora Kapeluz, Argentina, 1989.

[12]Mohr, H. B. : "*Avaliação de Desempenho de Rede PDV em Ambiente Industrial*". Tese de Doutorado, UFSC / RWTH-Aachen, Julho 1989.

[13]Pimentel, J. R. : "*Communication Networks for Manufacturing*". Prentice-Hall, New Jersey, USA, 1990.

[14]Shoch, J. F. : "*Evolution of the Ethernet local computer network*". IEEE Computer, pp. 10-27, August 1982.

[15]Zimmerman, H. : "*OSI Reference Model - The ISO Model of Architecture for Open Systems Interconnection*". IEEE trans. Commun., Vol. 28, pp. 425-432, April 1980.

[16]Wood, G. G. : "*International Standards Emerging for Fieldbus*". Control Engineering, Vol. 2, pp. 22-25, October 1988.

[17]Stemmer, M. R. : "*Einsatzmoeglichkeiten digitaler Feldbussysteme in geschlossenen, maschineninternen Regelkreisen*". Tese de Doutorado, RWTH-Aachen, Alemanha, Julho 1991.

[18]Arvind, K.; Ramamritham, K.; Stankovic, J.A. : "*A Local Area Network Architecture for Communication in Distributed Real-Time Systems*". The Journal of Real-Time Systems, pp. 115-147, Kluwer Academic Publishers, 1991.

[19]Le Lann, G.; Rivierre, N. : "*Real-time Communications over Broadcast Networks: the CSMA-DCR and the DOD-CSMA-CD Protocols*". Rapport de Recherche, INRIA, França, 1993.

[20]Kurose, J.F.; Schwartz, M.; Yemini, Y. : "*Multiple-Access Protocols and Time-Constrainde Communication*". Computing Surveys 16(1), pp. 43-70, March 1984.

[21]Tindell, K.; Burns, A.; Wellings A.J. : "*Analysis of Hard Real-Time Communications*". The Journal of Real-Time Systems, nr. 9, pp. 147-171, Kluwer Academic Publishers, 1995.

[22]Le Lann, G. : "*On Real-Time and Non Real-Time Distributed Computing*". Workshop on Distributed Algorithms, Springer Verlag, Sep. 1995.

[23]Diversos autores : "*Feldbus*". Edição especial da revista "Elektronik Plus", Franzis-Verlag GmbH, München, Alemanha, 1992.

[24]Echolon : "LonTalk protocol specification". Version 3.0, USA, 1994.