
Universidade de Aveiro Departamento de Electrónica e Telecomunicações
2005

Miguel Alexandre de Bastos Osório

Implementação de IP sobre novas camadas físicas

Miguel Alexandre de Bastos Osório

Implementação de IP sobre novas camadas físicas

dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Electrónica e Telecomunicações, realizada sob a orientação científica do Doutor Rui Luís Andrade Aguiar, Professor Auxiliar da Universidade de Aveiro, Departamento de Engenharia Electrónica e Telecomunicações

o júri

presidente	Doutor José Rodrigues Ferreira da Rocha, Professor Catedrático da Universidade de Aveiro
vogal	Doutor Mário Jorge Moreira Leitão, Professor Associado da Faculdade de Engenharia da Universidade do Porto
vogal	Doutor Rui Luís Andrade Aguiar, Professor Auxiliar da Universidade de Aveiro

agradecimentos

Agradeço em particular à Helena pelo auxílio e pela paciência que teve durante o período em que fiz esta dissertação. Agradeço também aos meus Pais, ao meu orientador e aos meus colegas do Instituto de Telecomunicações de Aveiro por toda a ajuda prestada.

resumo

O presente trabalho propõe-se estudar as metodologias existentes de integração e implementação do protocolo Internet sobre sistemas de telecomunicações. É apresentada a descrição da actual norma do RPR (IEEE 802.17) e em paralelo as opções de desenvolvimento para a implementação dos módulos do MAC (Medium Access Control), para o transporte de IP sobre uma rede em anel, segundo esta norma.

A primeira parte deste documento correspondente ao primeiro capítulo, aborda os serviços existentes actualmente para transporte de dados em pacotes, nas actuais redes de telecomunicações.

A segunda parte correspondente aos segundo e terceiro capítulos, é composta por uma abordagem às principais metodologias utilizadas para mapeamento de IP sobre diversos protocolos para posterior transporte na camada física, seguida de uma apresentação breve de soluções e produtos existentes no mercado, desenvolvidos por alguns fabricantes e operadores.

Os quarto, quinto e sexto capítulos, pertencem à terceira parte deste documento, onde é descrita a norma do IEEE 802.17 e as opções de desenvolvimento dos trajectos de dados do MAC, dos algoritmos de *Fairness* e de descoberta da Topologia e Protecção, e da unidade de Operação, Administração e Manutenção, para um sistema real desta referida norma. São descritas as soluções desenvolvidas para os diversos módulos e para todo o sistema que pressupõe a unidade de MAC do RPR. É também apresentada a implementação e simulação de um dos módulos do MAC do RPR.

Por fim no sétimo capítulo apresentam-se as respectivas conclusões.

São ainda apresentados dois anexos. O anexo A apresenta uma descrição detalhada de tipos e estruturas de tramas do protocolo Resilient Packet Ring - 802.17. No anexo B é descrita a estrutura da base de dados do algoritmo de descoberta de topologia e protecção, do referido protocolo.

abstract

The present work is intended to study the existing methodologies of integration and implementation of the Internet Protocol in telecommunications systems. It also proposes a starting point for the implementation of a module for the MAC of an emerging protocol to transport IP over a ring net. The description of the current standard of the RPR (IEEE 802.17) is presented and in parallel, the options of development for the implementation of the MAC (Medium Access Control) modules, for the transport of IP on a ring network, according to this standard.

The first part of this document, corresponding to the first chapter, currently approaches the existing services for transport of data in packages, in the present telecommunication networks.

The second part, corresponding to the second and third chapters, is composed of an approach to the main methodologies used for IP mapping on diverse protocols for posterior transport in the physical layer, followed of a brief presentation of solutions and existing products in the market, developed by some manufacturers and operators.

The fourth, fifth and sixth chapters, belong to the third part of this document, where it is described the 802.17 standard of the IEEE and the options for the development of the MAC datapaths, of the Fairness and Topology Discovery and Protection, algorithms, and of the Operation, Administration and Maintenance unit, for a real system of this standard. It is described the solutions developed for the several modules and for all the system that compose the unit of the RPR MAC. It is also presented the implementation and simulation of one of the modules of RPR MAC.

Finally in the seventh chapter the respective conclusions are presented.

Still two annexes are presented. The annex A presents a detailed description of frame types and structures of the Resilient Packet Ring – 802.17 protocol. In the annex B it is described the structure of the database for the topology discovery and protection algorithm, of the related protocol.

Índice

Implementação de IP sobre novas camadas físicas

0 Informação Geral

0.1 Acrónimos

ADM add drop multiplexer

ANSI american national standard for information systems

ASP application service provider

ATD attribute discovery

ATM asynchronous transfer mode

ATT attribute type

AUG administrative unit group

AU-N administrative unit level n

BER bit error ratio

B-ISDN broadband isdn

BLSR bidirectional line switched rings

CAD computer aided design

CIR committed information rate

CLB configurable logic block

CN container level n

CRC cyclic redundancy check

CSMA/CD carrier sense multiple acess with collision detection

DCE data circuit-terminating equipment

DCS digital cross connector

dLOC loss of continuity failure defect

DTE data terminating equipment

DUT device under test

DWDM dense wdm

EIR excess information rate

EISS enhanced internal sublayer service

EPL ethernet private line

FCS frame check sequence

FDD fairness differential delay

FIFO first in first out

FOH framing over head

FPGA field programmable gate array

FRTT fairness round trip time

FS forced switch

GBR guaranteed bit rate

GFP generic framing procedure

GMII gigabit media independent interface

GRS gfp reconciliation sublayer

HDL hardware description language

HDLC high-level data link control

HEC header error check

ICMP internet control message protocol

IEC international electrotechnical commission

IEEE institute of electrical and electronics engineers

IETF internet engineering task force

IP internet protocol

IPCP ip control protocol

ISDN integrated services digital network

ISO international organization for standardization

ISP internet service provider

ISS internal sublayer service

ITU-T international telecommunication union

LAN local area network

LAPS link access procedure - SDH

LCP link control protocol

LLC logical link control

LME layer management entity

LOC loss of continuity failure

LRTT loop round trip time

LSB least significant bit

LVDS low-voltage differential signals

LVTTL low-voltage transistor-transistor logic

MAC medium access control

MAN metropolitan area network

MBR maximum burst rate

MCFF multi choke fairness frame

MIB management information base

MII media independent interface

MLME mac layer management entity

MRU maximum receive unit

MS manual switch

MSB most significant bit

MSOH multiplexer section over head

MSPP multi service provision platform

MTU maximum transfer unit

NCP network control protocol

NME network management entity

OAM operations, administration, and maintenance

OC-N optical carrier level n

OEM original equipment manufacturer

OIF optical Internetworking forum

OSI open systems interconnection

OTN optical transport network

PCB printed circuit board

PCS physical coding sublayer

PDH plesiochronous digital hierarchy

PDU packet data unit

PHY physical layer

PICS protocol implementation conformance statement

PLI pdu length indicator

PLME physical layer management entity

PMA physical medium attachment

PMD physical medium dependent

POH path over head

POS packet over sonet/sdh

PPM parts per million

PPP point to point protocol

PRS-1 1 Gb/s 1Gb/s packet phy reconciliation sublayer

PRS-10 10 Gb/s 10Gb/s packet phy reconciliation sublayer

PTQ primary transit queue

PVC permanent virtual connection

QoS quality of service

RFC request for comment

RPR resilient packet ring

RRTT ring round trip time

RS reconciliation sublayer

RSOH regenerator section over head

RTL register transfer level

SAN storage area network

SAPI service access point identifier

SCFF single choke fairness frame

SD signal degrade

SDH synchronous digital hierarchy

SDU service data unit

SF signal fail

SLA service level agreement

SME station management entity

SNMP simple network management protocol

SONET synchronous optical network

SPI system packet interface

SPI-3 system packet interface level 3

SPI-4.1 system packet interface level 4 phase 1

SPI-4.2 system packet interface level 4 phase 2

SRS SONET/SDH reconciliation sublayer sonet/sdh

STM-N synchronous transport module level n

STQ secondary transit queue

STS-N synchronous transport signal level n

TC topology checksum

TCP transmission control protocol

TDM time division multiplexing

TNE transport network element

TP topology and protection

UDP user datagram protocol

UITS unacknowledged information transfer service

VC virtual container

VC-N virtual container level n

VLAN virtual lan

VPL virtual private line

VPN virtual private network

WAN wide area network

WDM wavelength division multiplexing

WIS wan interface sublayer

WTR wait to restore

XAUI 10 gigabit attachment unit interface

XGMII 10 gigabit media independent interface

XGXS xgmii extender sublayer

0.2 Termos e definições

agente: Uma NME usada para configurar uma estação e/ou colectar dados que descrevem a operação dessa estação.

ajuste agressivo da taxa: Método de ajuste da taxa em que as taxas podem ser mudadas sem atraso significativo e as quantias de mudança da taxa não são "altamente amortecidas".

ajuste conservador da taxa: Método de ajuste da taxa no qual a taxa geralmente não é alterada até que tenha decorrido um tempo suficiente para que o efeito de qualquer ajuste prévio tenha sido observado, e cuja quantidade de mudança exiba amortecimento e histerese significativos.

anel aberto: Anel que foi cortado, portanto impedido de completar uma ligação em volta. Um anel aberto tem pelo menos uma estação *edge* detectada.

anel fechado: Anel intacto (sem cortes), que providencia um trajecto completo em todo o caminho à volta do anel. Um anel fechado não contém *edges* detectadas.

atraso de transito: Atraso ocorrido a uma trama que transita por uma estação no *ringlet*.

atraso *extremo-a-extremo*: Tempo requerido para a transferência de uma trama entre as estações de origem e de destino, medido desde o início da transmissão da trama até ao início da recepção da trama, nas respectivas interfaces de serviço.

***best-effort*:** Não associado com uma garantia explícita de serviço.

***bit error ratio* (BER):** Relação entre o número de bits recebidos com erro e o número de bits transmitido.

***bridge*:** Unidade funcional que liga duas ou mais redes na camada de Data Link do modelo OSI.

***bridging* transparente:** Mecanismo de *bridging* que é transparente para as estações de destino.

***broadcast*:** Acto de enviar uma trama endereçada a todas as estações numa rede.

cabeça de congestionamento: É a estação mais a jusante de um domínio de congestionamento. Esta é a estação imediatamente a montante do ponto de congestionamento.

camadas superiores: Conjunto de camadas protocolares acima da camada de *data-link*.

capacidade disponível: Capacidade do anel que não é requerida para suportar o serviço atribuído e está, consequentemente, disponível para suportar serviços oportunistas.

cauda de congestionamento: Estação mais a montante de um domínio de congestionamento.

classe de serviço: Categorização do serviço do MAC em termos de limites de atraso, prioridade relativa, garantias de taxa de dados, ou características similares distintas.

***cleave point*:** Ponto entre estações, no qual o anel é logicamente dividido.

cliente MAC: Entidade de camada que invoca a interface de serviço do MAC.

***committed information rate* (CIR):** Taxa à qual a rede acorda em transportar dados medida sobre um mínimo intervalo de tempo.

congestionamento: Estado de um anel ou estação determinado pelo cálculo do algoritmo de *fairness* e que causa restrições ou regulações na adição ao anel, de tráfego elegível para *fairness*.

conservador: Diz-se de um algoritmo que tende a exibir uma resposta transitória sobre-amortecida.

controlo de fluxo: Método de controlo de congestionamento em que uma estação comunica, a uma outra estação ou estações, informação de congestionamento pretendida para regular a transmissão das tramas.

domínio de congestionamento: Conjunto de ligações contíguas associadas a um anel, que causa congestionamento num ponto comum de congestionamento. É também o conjunto de ligações entre as estações de congestionamento de cabeça e de cauda.

elegível para fairness: Qualidade de uma trama que indica se está sujeita ao algoritmo de *fairness*.

endereço de *broadcast*: Endereço de grupo que consiste em tudo a 1 e identifica todas as estações numa rede.

endereço de grupo: Endereço que identifica um grupo de estações numa rede.

endereço *multicast*: Endereço de grupo que não é um endereço de *broadcast*, i.e., não é igual a tudo a 1s, e identifica algum subconjunto de estações na rede.

evento de comutação de protecção: Trama de controlo de protecção recebida que causa a actualização/modificação da topologia local e a base de dados de *status*.

***excess information rate* (EIR):** Taxa à qual a rede transfere dados em excesso relativamente à informação cometida, quando não existe congestionamento da rede.

***fairness* ponderado:** Classe do algoritmo de *fairness* que permite a atribuição de partes diferentes da capacidade disponível do anel.

***fairness round trip time* (FRTT):** Tempo que toma a um valor de *fairness* para se propagar desde a cabeça de um domínio de congestionamento até à cauda do mesmo, e para que a primeira trama afectada seja enviada desde a cauda do domínio de congestionamento e seja recebida pela cabeça do mesmo.

***fairness*:** Propriedade que para uma qualquer ligação no anel, cada estação origem recebe uma proporção igual de capacidade elegível para *fairness*. Se todas as estações de origem tiverem pesos iguais, então as mesmas têm o acesso substancialmente igual à capacidade disponível de todas as ligações.

fila de trânsito: Fila mantida para o propósito de armazenar numa estação, uma trama em trânsito, até que essa trama tenha permissão para continuar a transitar no *ringlet*.

***flooding* bidireccional:** Transferência de trama que envolve a emissão de duas tramas de *flooding*. Uma em cada *ringlet* (*ringlet0* e *ringlet1*), onde cada trama é dirigida a estações adjacentes distintas.

***flooding scope*:** Número de *hops* que uma trama pode viajar (em torno do anel) desde uma dada estação de origem até uma estação de destino associadas a um dado *ringlet.*

***flooding* unidireccional:** Transferência para a frente de uma trama envolvendo o envio de uma trama de *flooding* na direcção a jusante de um *ringlet*, com essa trama dirigida para a sua estação de envio.

***flooding*:** Acto de transmitir uma trama tal que todas as estações no anel recebam essa trama uma vez.

garantia de serviço: limites de atraso ou jitter, ou garantias de largura de banda para uma unidade de classe de serviço.

***hop-count*:** Número de *hops* atravessados pelos dados que circulam entre estações no anel.

***in flight*:** Transmitido pelo MAC de origem e ainda não recebido pelo MAC de destino.

***in transit*:** Recebido por um PHY do MAC e ainda não transmitido pelo PHY de outro MAC.

inversão de maior prioridade: Qualquer efeito causa-reclamação que viola quaisquer garantias de classe de serviço igual ou superior.

inversão de menor prioridade: Qualquer efeito causa-reclamação numa classe de serviço igual ou superior que não viola quaisquer garantias dessa classe de serviço.

***jitter*:** Variação no atraso associado a uma transferência de tramas entre dois pontos.

***keepalive*:** Troca de mensagens permitindo a verificação de que está activa uma comunicação entre estações.

largura de banda atribuída: Largura de banda que pode ser sujeita a reclamação, mas que deve estar disponível à estação para a qual está atribuída, quando requisitada. Uma classe de tráfego é dita estar atribuída quando os limites associados a essa classe são determinados pelo operador de engenharia da rede ou departamento de operações. Há dois tipos de largura de banda atribuída: reservada e reclamável. A largura de banda atribuída é referente à quantidade de largura de banda num *ringlet* que fornece

a capacidade necessária para classes de serviços de taxa cometidos.

largura de banda não atribuída: Largura de banda que não está atribuída para qualquer serviço aprovisionado.

largura de banda reclamável: Subconjunto de largura de banda atribuída que é dinamicamente reclamada pelo algoritmo de *fairness*.

largura de banda reservada: Quantidade de largura de banda que deve ser mantida disponível (i.e., não sujeita a reclamação). Representa o subconjunto da largura de banda atribuída que não é dinamicamente reclamada pelo algoritmo de *fairness*.

latência: Tempo requerido para transferir informação desde um ponto até outro.

***layer management entity* (LME):** Entidade numa camada que executa a gestão local de uma camada. O LME fornece a informação sobre a camada, efectua controlo sobre a mesma, e indica a ocorrência de determinados eventos dentro da mesma.

ligação: Canal unidireccional que liga estações adjacentes num *ringlet*.

***logical link control* (LLC):** Parte da camada de ligação de dados que suporta funções de ligações de dados independentes do meio, e usa os serviços da sub-camada do MAC para fornecer serviços à camada de rede.

***loop round trip time* (LRTT)**: Tempo que decorre para que uma trama de controlo ou de dados, seja enviada de uma estação a outra e de volta à estação original.

***management information base* (MIB):** Repositório de informação que descreve a operação de um equipamento específico de rede.

***maximum transfer unit* (MTU):** A maior trama (cabeçalho, *payload* e *trailer*) que pode ser transferida através da rede.

modo *non-revertive*: Uma estação está no modo *non-revertive* se, quando do cancelamento de todas as falhas numa das suas extensões, a estação permanece num estado de protecção *wait-to-restore* até que esta condição seja interrompida por uma condição de protecção de mais alta prioridade em qualquer lado do anel.

modo *revertive*: Uma estação está no modo *revertive* se, quando do cancelamento de todas as falhas numa das suas extensões, essa estação entra num estado de *wait-to-restore*, seguido de uma entrada no estado de *idle* após expiração do tempo de *wait-to-restore*.

***multicast*:** Transmissão de uma trama para estações especificadas por um endereço de grupo.

***multi-choke*:** Característica de um algoritmo, relacionada com a observação de múltiplos pontos de congestionamento num *ringlet*.

não reservado: Designação para capacidade de tráfego que não é reservada. É também, a designação para o tráfego que ocupa essa largura de banda. Adicionalmente, a largura de banda não reservada pode ou não ser largura de banda atribuída.

***packet*:** Termo genérico para uma PDU associada com uma camada-entidade acima da sub camada do MAC.

***passthrough*:** Condição de um MAC tal que as tramas passam através, como se estivesse a operar como uma ligação entre as estações em ambos os lados do mesmo.

***physical layer* (PHY):** Camada responsável por fazer a interface com o meio de transmissão.

***plug-and-play*:** Requisito para que uma estação desempenhe as actividades de trânsito, *strip* e controlo do anel sem a intervenção do operador. A estação pode adicionalmente copiar e inserir tramas.

ponto de congestionamento: Ligação de transmissão de uma estação que experimenta congestionamento mas que não contribui para congestionamento a jusante. Esta ligação está imediatamente a jusante da cabeça do congestionamento.

***protocol data unit* (PDU):** Informação enviada como uma unidade entre pares de entidades de camada que contém informação de controlo e opcionalmente, dados.

reutilização espacial: Transferência concorrente de tráfego independente em porções não sobrepostas de um anel.

***ring round trip time* (RRTT):** Tempo que decorre para que uma trama de controlo ou de dados seja enviada à volta do anel completo.

***ringlet*:** Ligação na qual o tráfego de dados circula unidireccional entre estações num anel composto por duas ou mais ligações.

***ringlet* oposto:** *Ringlet* cujo tráfego circula na direcção oposta a um determinado *ringlet*.

***service data unit* (SDU):** Informação entregue como uma unidade entre entidades de camadas adjacentes, possivelmente contendo também uma PDU da camada superior.

***shaper*:** Unidade que converte um fluxo de tráfego arbitrário para um fluxo de tráfego "suave" a uma taxa de dados especifica.

***single-choke*:** Característica de um algoritmo, relacionado com a observação de apenas um ponto de congestionamento num *ringlet*.

***source stripping*:** remoção das tramas pela estação de origem após terem circulado no anel.

sub camada de adaptação: Sub camada protocolar que tem como objectivo converter dados de um formato para outro.

sub camada de reconciliação (RS): Sub camada que providencia um mapeamento entre a interface de serviço PHY e a interface independente do meio da PHY.

sub camada *medium access control* (MAC): Porção da camada de ligação de dados que controla e medeia o acesso ao meio da rede.

taxa da ligação: Taxa de dados com que um MAC se comunica com a sua entidade MAC adjacente.

taxa da linha: Taxa em que um PHY transfere dados no meio físico ao qual está ligado.

taxa justa: Taxa que tem a propriedade de que uma estação é impedida de utilizar uma parte desproporcional da capacidade disponível do *ringlet* relativamente a outras estações no mesmo.

taxa: Número de bytes transferido por unidade de tempo.

topologia: Arranjo das ligações e estações que formam uma rede, conjuntamente com informação de atributos da estação.

trama de dados estrita: Trama que tem activado o seu bit *so* (*strict order*).

transferência: Movimento de uma SDU de uma camada para outra camada adjacente ou de um ponto a outro na rede.

transmissão: Acção de uma estação colocar uma trama no meio.

***unicast* desconhecido:** Trama *unicast* para a qual a localização do seu destino é desconhecida.

***unicast*:** Acto de enviar uma trama endereçada a uma única estação.

***virtual* LAN (VLAN):** Subconjunto da topologia active de uma LAN *bridged*.

1 Introdução

1.1 Enquadramento

Este trabalho enquadra-se na área das redes de telecomunicações para comunicação de dados (MAN/WAN). Actualmente o tráfego de dados baseado na *Internet* está a expandir-se a um ritmo elevado, e as tecnologias utilizadas nestas redes estão a evoluir de forma a tornar mais eficiente o transporte desses mesmos dados. Torna-se portanto necessário analisar as tendências da actual evolução de tráfego de dados e qual o seu impacto nas infra-estruturas de telecomunicações, com o objectivo de investigar e desenvolver novas tecnologias que permitam satisfazer da melhor forma todos os requisitos e necessidades dos utilizadores dessas referidas infra-estruturas.

Em face ao actual desenvolvimento do mercado (o tráfego de dados baseia-se essencialmente no protocolo IP [11-a][11-b]), alguns fabricantes e operadores começam a apresentar soluções e produtos que permitem transportar IP utilizando a infra-estrutura de telecomunicações já existente e fornecendo IP ao cliente de uma forma eficiente e económica. Na integração e implementação do protocolo IP para transporte em sistemas de telecomunicações, são utilizadas técnicas e metodologias que efectuam o mapeamento do mesmo sobre diversos protocolos para posterior transporte na camada física. No entanto algumas dessas metodologias e técnicas, impõem limitações no transporte deste mesmo tráfego.

Entre outras opções para transporte de tráfego IP nas MAN/WAN, a futura norma do RPR (Resilient Packet Ring - IEEE 802.17) [7] apresenta-se como uma solução particularmente atractiva, eficiente e económica para o transporte desse tráfego sobre os actuais anéis SONET/SDH [14][15] que são a arquitectura dominante nas redes dos operadores de telecomunicações.

Assim e em face às actuais tendências dos fabricantes e operadores de telecomunicações, neste documento serão descritas de uma forma sucinta, algumas das técnicas já existentes e em fase de implementação para o transporte de IP e, em mais pormenor, as opções de desenvolvimento dos módulos constituintes do MAC da norma do IEEE 802.17 – RPR, onde se incluem os trajectos de dados do MAC, os algoritmos de *Fairness* e de descoberta da Topologia e Protecção, e a unidade de Operação, Administração e Manutenção (OAM), necessários para a implementação de um sistema real desta referida norma.

Este documento apresenta também a metodologia de desenvolvimento utilizada, recorrendo à linguagem SystemC [22][26][34] e suportada sobre ferramentas de desenvolvimento da empresa Synopsys [24][25][28][30]. Este desenvolvimento foi efectuado usando adequadamente as metodologias *top-down* e *bottom-up* através da descrição em SystemC comportamental e ao nível da lógica de registos, de todos os módulos do sistema. O projecto cobre todos os módulos da camada MAC do RPR e interfaces para o cliente e para a camada física. Isto inclui as sub-camadas de *datapath* e de controlo do MAC, contendo esta última, os módulos de *Fairness*, Topologia e Protecção, e OAM. Foram também desenvolvidas interfaces MII para um cliente 100Mb/s Ethernet e GMII para a camada física 1000Mb/s Ethernet [5]. Estes módulos foram desenvolvidos tendo como objectivo a sua implementação em FPGA, para posteriormente ser integrada num sistema usando memória externa para armazenamento da base de dados da Topologia, um processador externo para cálculo de estatísticas, e interfaces externas MII e GMII. As simulações dos módulos foram desenvolvidas usando o ambiente em SystemC, fazendo a co-simulação de vectores de teste em SystemC com os módulos do MAC do RPR compilados de SystemC para Verilog e fazendo testes num FPGA da Xilinx [35], permitindo comparações directas entre as simulações e os resultados práticos.

1.2 O transporte de dados em pacotes sobre a camada física

A evolução de tipos de tráfego (figura 1.1) mostram que tem vindo a ocorrer um aumento no tráfego de dados IP. Estas tendências são o corolário do desenvolvimento do mercado que gera mais tráfego de dados. Os principais fornecedores de *Internet* têm vindo a relatar a duplicação,

aproximadamente, da largura de banda nos seus *backbones* em cada seis a nove meses. Este enorme crescimento do tráfego *Internet* baseado em IP tem vindo a evidenciar algumas limitações na infra-estrutura de transporte existente. A adopção de *Intranets* e de *Extranets* para o comércio em rede requer mudanças adicionais à infra-estrutura de serviços IP, devido às respectivas necessidades de largura de banda e às exigências nos requisitos.

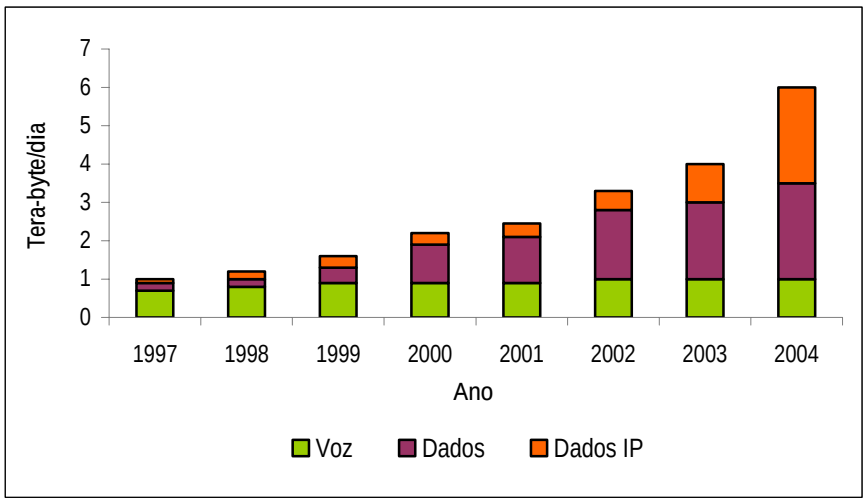


Figura 1.1 – Evolução de tipos de tráfego [1]

Para que haja eficiência no transporte deste constante aumento de tráfego IP, é necessária uma tecnologia também eficiente para transportar os pacotes de IP até à camada física.

No final dos anos 80 as tecnologias SDH e SONET foram introduzidas na rede de transporte para obviar os problemas com a tecnologia PDH. As tecnologias SDH e SONET são baseadas em transmissão TDM e a norma SDH/SONET tem vindo a evoluir existindo actualmente sistemas a funcionar acima de 10 Gbit/s, optimizados principalmente para a transmissão de voz. Ao nível do transporte, a tecnologia WDM está também a ter um crescimento constante. Estão actualmente em operação sistemas até 160 canais de comprimentos de onda, cada um transmitindo 10 Gbit/s numa fibra. Ao combinar os benefícios do DWDM, e das tecnologias TDM e SDH/SONET é possível actualmente, transportar informação até uma escala de Terabit por segundo.

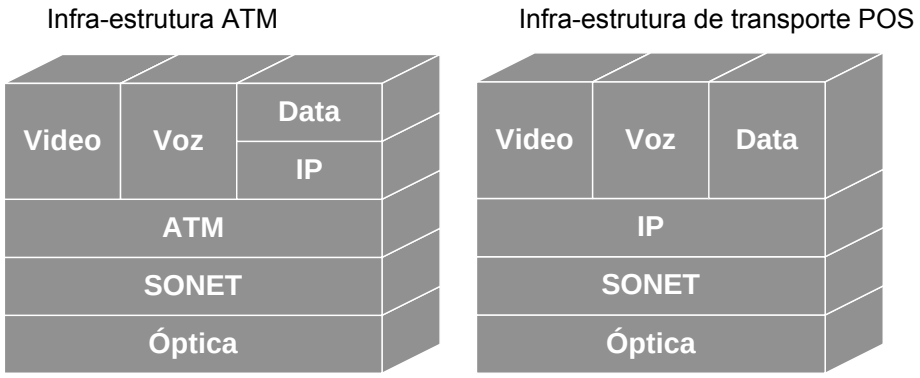


Figura 1.2 – Infra-estruturas de transporte ATM e POS

A tecnologia SDH/SONET existe desde 1986, e o advento de tecnologias tais como o ATM sobre SONET/SDH (e mais recentemente, o mapeamento directo de IP sobre tramas de SONET/SDH) estendeu a vida útil desta tecnologia.

Até recentemente, o ATM parecia ser o único método viável de agregar a voz e o tráfego de dados em redes de serviços múltiplos e de alta velocidade. No entanto a abordagem POS (Packet over SONET/SDH) [1][9][10], oferece uma arquitectura para o *backbone* que preserva os investimentos existentes na infra-estrutura SONET/SDH e suporta a distribuição de aplicações de vídeo e de voz baseadas em IP. O POS coloca a camada de IP directamente em cima da camada de SONET/SDH, e ao oferecer garantias de QoS elimina o *overhead* decorrente de transportar o IP sobre o ATM e por sua vez sobre SONET/SDH (figura 1.2).

Estas duas infra-estruturas de transporte de tráfego IP são actualmente as principais técnicas existentes para "entregar" pacotes IP à camada física.

No entanto actualmente o tráfego de dados em pacotes não é efectuado somente em IP na sua forma nativa (i.e., datagramas IP). Este tráfego de pacotes, nas redes de telecomunicações, é feito maioritariamente através de datagramas IP mapeados em Ethernet. Há portanto a necessidade de estudar quais os serviços Ethernet existentes, quais os impactos destes na eficiência das actuais redes de telecomunicações e qual a gestão necessária para estes mesmos serviços.

1.3 Um novo paradigma para a gestão de serviço

O paradigma mais comum da gestão de serviço em telecomunicações baseia-se em aprovisionamento e gestão de ligações ponto-a-ponto. Mas quando os serviços se tornam mais sofisticados, começam a levar ao limite as potencialidades deste paradigma. O paradigma da gestão de serviço necessita actualmente de um modelo novo de gestão da ligação, que controle também as ligações multi-ponto e serviços de *multicast*. Necessita ainda ter conhecimento da classe de serviço da aplicação.

Reconhecida extensamente como a tecnologia dominante no mundo empresarial, a Ethernet é tida frequentemente como estando abaixo da classe de transporte. À Ethernet é considerado faltar a escalabilidade e a fiabilidade esperadas para as infra-estruturas de classe de transporte da rede pública. Esta deficiente concepção vem do facto de se confundir a Ethernet como um serviço, da Ethernet como uma tecnologia de comutação.

A Ethernet como uma tecnologia de comutação não é uma tecnologia de classe de transporte. Mas os serviços de Ethernet podem ser entregues através de um número de diferentes tecnologias, incluindo o SONET/SDH, o DWDM e o RPR, que são todos bastante fiáveis. Além disso, tecnologias mais recentes tais como o RPR fornecem uma escalabilidade sem precedentes quando se trata de fornecer serviços Ethernet.

É conveniente uma abordagem de gestão de serviço que incorpore um gestor da ligação Ethernet (dada a sua importância) e um gestor de serviço baseado em policiamento (dada a sua flexibilidade).

Gestores da ligação Ethernet oferecem arquitecturas sofisticadas, tais como pontes virtuais e VLANs para gerir as exigências de ligação de serviços Ethernet. Por exemplo, para criar uma VPN empresarial, um administrador de rede necessita somente especificar os portos que dão forma à VPN, não necessita criar e controlar uma rede em malha de circuitos ponto-a-ponto. O sistema de gestão de serviço tem a tarefa de configurar os comutadores individuais de pacotes para implementar a arquitectura.

Analogamente, o gestor de serviço baseado em policiamento simplifica o aprovisionamento e a gestão de serviços da aplicação específica. Uma política de gestão de serviços tem diversos componentes, os mais importantes dos quais são as regras de classificação e especificação de QoS. As regras de classificação identificam o tipo de tráfego. A especificação do QoS define o nível de serviço a ser atribuído a este tipo de tráfego. Por exemplo, para criar um serviço de *best-effort* de largura de banda baixa para todas as aplicações de *e-mail* e aplicações *web*, o administrador da rede cria apenas uma política de serviço. O sistema de gestão de serviço tem a

tarefa de ajustar as tabelas de configuração dos comutadores no anel para se assegurar de que as aplicações de *e-mail* e *web* obtenham sempre o QoS especificado.

1.4 Serviços Ethernet

A maioria das infra-estruturas de redes de negócios é hoje constituída por uma mistura de serviços de rede que incluem linhas privadas, ATM e Frame Relay. Apesar de no passado terem sido adequados para as necessidades de tráfego de dados, actualmente apresentam algumas deficiências. Por exemplo, o Frame Relay não oferece a possibilidade de ser escalável, e o ATM não consegue alcançar os níveis de custo requeridos para se atingir as necessidades do mercado da economia em rede.

Com a procura crescente por parte do cliente de um acesso económico à rede para velocidades de 10 Mb/s, 100 Mb/s e mais, há uma necessidade óbvia para uma nova classe de serviços de dados que seja mais escalável e económica do que a que está actualmente disponível. O desenvolvimento de serviços Ethernet (serviços que são oferecidos como ligações Ethernet ao exterior) por parte dos operadores preenche este requisito.

A largura de banda e o custo, no entanto, não são as únicas motivações para os serviços de Ethernet. Um serviço de Ethernet é ideal para um cliente de serviços de dados, por questões de convergência tecnológica, simplicidade de evolução e integração de acesso. A maioria das empresas já opera internamente com Ethernet. A única razão para que operem com outras tecnologias, sejam Frame Relay, SONET/SDH ou ATM, é a comunicação inter-escritórios. Se esta necessidade fosse cumprida através da Ethernet, estes clientes seriam poupados ao incómodo de gerir múltiplas tecnologias. O *router* de acesso do cliente poderia ligar directamente a redes remotas usando um dos seus portos nativos Ethernet.

Uma outra vantagem da Ethernet é a escalabilidade. Um fornecedor de serviços pode fornecer fisicamente apenas uma vez um porto de Fast Ethernet (100 Mb/s) ou Gigabit Ethernet (1000 Mb/s) a um subscritor, e actualizar várias vezes o serviço fornecido, sem adição de equipamento para além da instalação inicial. A largura de banda e outras mudanças de serviço podem ser geridas remotamente, simplificando e acelerando o aprovisionamento do serviço.

Finalmente, não se pode negligenciar a capacidade da Ethernet em suportar aplicações integradas de acesso, quando entregues sobre a infra-estrutura correcta. Os fornecedores de serviços procuram maneiras de diferenciar os seus serviços. Oferecer serviços integrados é uma solução. Com tantas aplicações baseadas agora nas técnicas e nas tecnologias de pacote (isto é, IP sobre Ethernet), a Ethernet torna-se assim uma interface natural de serviço para estes grupos de serviços (figura 1.3).

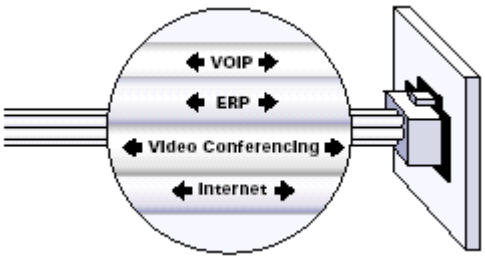


Figura 1.3 – Serviços agrupados

1.4.1 Linha Privada Virtual

Uma linha Ethernet privada virtual é um circuito ponto-a-ponto, tal como uma linha alugada Frame Relay ou ATM PVC. Os SLAs para VPLs são também similares a SLAs tradicionais e incluem atributos tais como o CIR, a taxa de pico, a duração permitida de *burst*, a latência *extremo-a-extremo* e a interface de entrega dos dados.

As VPLs podem ser usadas em várias aplicações incluindo a interligação de LAN-a-LAN e as VPNs. Para tais aplicações, que requerem tipicamente a ligação entre mais do que duas posições de cliente, a solução requer a criação de linhas confidenciais virtuais múltiplas em estrela ou em malha. A figura 1.4 mostra uma rede típica de empresa, com as redes satélite ligadas à rede principal através de circuitos ponto-a-ponto.

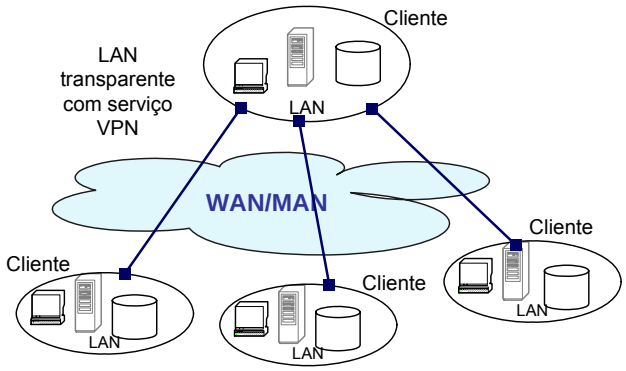


Figura 1.4 – Arquitectura de rede tipo *circuit-oriented*

1.4.2 Serviço Ethernet Bridged

Com tecnologias centradas na comutação de circuitos, tais como SONET/SDH e DWDM, tudo o que se pode entregar são circuitos ponto-a-ponto. As arquitecturas de comutação de pacotes, por outro lado, suportam um mais rico e mais "amigável" modelo de dados de serviço Ethernet e consequentemente maior flexibilidade, tais como os serviços suportados de Ethernet ponto-a-multiponto construídos numa *Bridge*.

Com um serviço Ethernet *Bridged*, os fornecedores de serviços fazem o *drop* de um porto Fast Ethernet ou Gigabit Ethernet em cada posição de cliente, e então providenciam uma ponte virtual para ligá-los. A ponte virtual tem o dever de encaminhar tramas individuais Ethernet entre os *sites*. Em consequência, um *site* de cliente pode comunicar com qualquer outro *site* na empresa sem a necessidade de uma malha de circuitos ponto-a-ponto (figura 1.5).

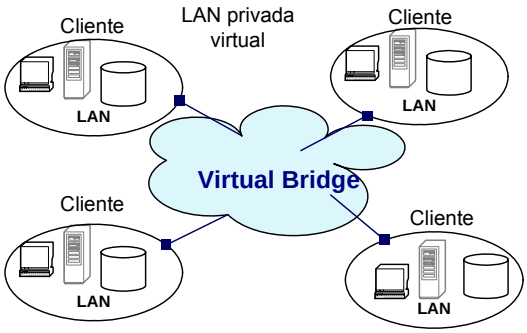


Figura 1.5 – Serviço Ethernet Bridged

Um serviço Ethernet *Bridged* constrói literalmente uma ponte entre as diferentes LANs numa empresa e uma LAN filial (*corporate*) privada virtual.

1.5 Resilient Packet Rings – a fundação para serviços avançados de dados

Presentemente, os fornecedores de serviços de aplicação têm duas opções para fornecer o acesso de rede aos subscritores para os seus serviços: circuitos comutados tais como T1 ou Frame Relay PVCs, ou VPNs baseadas em protocolo IP, sendo ambos pouco optimizados do ponto de vista de utilização dos recursos do operador. A primeira aproximação é muito cara, especialmente quando uma empresa subscreve serviços com mais do que um ASP; ao segundo falta garantias determinísticas de serviço.

Nem o SONET/SDH nem a comutação Ethernet estão projectados para se dirigirem às necessidades de uma camada de MAC projectada para o ambiente da área metropolitana (figura 1.6). O SONET/SDH emprega técnicas da camada 1 para a gestão da largura de banda e para a protecção do serviço. Os comutadores Ethernet baseiam-se em fazer o *bridging* da Ethernet ou em encaminhar o IP para a gestão da largura de banda, e prestam serviços de manutenção e protecção. Consequentemente, a rede é sub utilizada no caso do SONET/SDH ou não determinística no caso de comutadores de Ethernet.

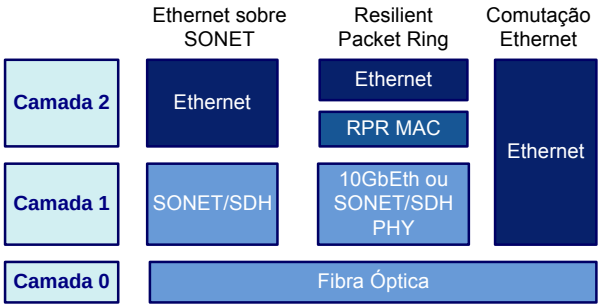


Figura 1.6 – Perspectiva de camadas das MAN

A natureza *circuit-oriented* do SONET/SDH é um excelente meio de adaptação para tráfego TDM, como por exemplo a voz ou circuitos comutados. A mesma natureza *circuit-oriented* é também o seu defeito quando é necessário transportar tráfego de dados em pacotes (*burst*).

Aos produtos Ethernet de comutação e encaminhamento, por outro lado, embora *data-friendly*, falta-lhes a robustez, escalabilidade e o serviço determinístico requerido para a infra-estrutura da rede pública em termos de serviços tradicionais.

A distribuição de conteúdos multimédia pela arquitectura de rede actual é similarmente deficiente. *Streaming* media, vídeo no *desktop* e vídeo-conferência, todos requerem ligações de largura de banda elevada e com um baixo *jitter*.

Há assim necessidade de uma nova tecnologia para as redes de pacotes metropolitanas. A nova arquitectura deve ser de comutação de pacotes como os comutadores e *routers* Ethernet, mas deve também ser fiável, escalável e determinística como o SONET/SDH. Além disso, deve incorporar características tais como o *multicast*, o conhecimento da classe de serviço da aplicação e uma gestão mais sofisticada desse serviço.

O RPR é uma arquitectura de rede emergente projectada para alcançar as exigências de uma rede MAN baseada em tráfego de pacotes. Uma rede RPR é uma arquitectura baseada em anel que consiste em nós de comutação de pacotes que se ligam aos nós adjacentes sobre um único par de fibras. A topologia da rede é baseada em anéis duplos contra-rotacionais.

Controlando o acesso ao meio e arbitrando os pedidos para o uso do meio, a camada MAC do RPR pode garantir a qualidade de serviço (isto é, atraso e *jitter*) e a gestão justa da largura de banda. Além disso, o MAC do RPR executa um mecanismo de protecção de serviço para proteger de falhas no anel e um esquema para evitar o congestionamento que permite ao sistema operar perto da capacidade máxima ao assegurar a qualidade de serviço a todos os serviços configurados.

Um anel é também a melhor topologia para a entrega de tráfego *multicast*. Em vez de ter centenas de filas de tráfego ponto-a-ponto a transportar os mesmos dados, há apenas uma fila de tráfego que atravessa o anel. Se os clientes servidos por um comutador individual de pacotes subscreverem este serviço particular de *multicast*, o comutador copia os dados e entrega-os localmente. O MAC do RPR, na camada 2, faz o processamento de pacotes para estes serem transportados num anel duplo SDH/SONET na camada 1.

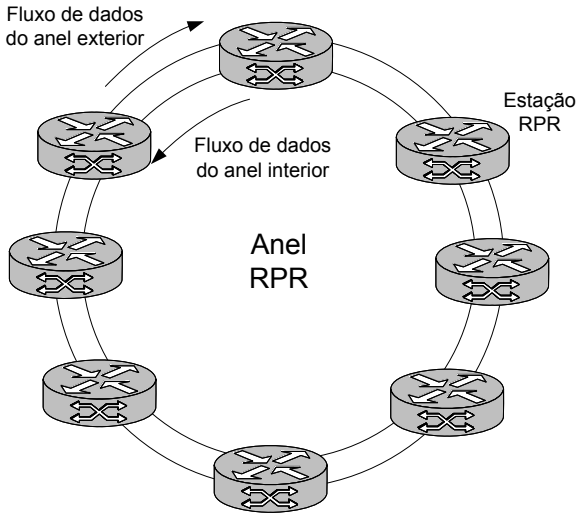


Figura 1.7 – Arquitectura de um anel RPR

1.5.1 O modelo de serviço do RPR

O RPR pode ser visto como um comutador distribuído, com os portos distribuídos geograficamente em que os anéis contra-rotacionais formam o *backplane* deste comutador. A ligação entre um conjunto de portos é estabelecida através da criação de uma ponte virtual entre eles. Tal como uma ponte de uma empresa liga vários segmentos da LAN a uma LAN homogénea da empresa, uma ponte virtual liga segmentos geograficamente distribuídos da LAN numa LAN homogénea, mas distribuída. Assim os conceitos de redes virtuais também são explorados no RPR.

Cada ponte virtual tem uma base de dados de encaminhamento que permite dirigir aos seus destinos apropriados, as tramas Ethernet recebidas. A ponte virtual toma as decisões de encaminhamento baseadas no endereço de destino do MAC – o endereço Ethernet codificado em cada cabeçalho da trama Ethernet. Além de interligar vários postos da empresa, uma rede virtual pode também ser usada para ligar uma empresa a um fornecedor de serviços, tal como um ASP ou um ISP. Uma empresa pode assim ter pontes virtuais múltiplas, uma para a VPN da companhia, uma segunda para a sua ligação de acesso à Internet e uma terceira para a ligação do seu ASP. Além disso, um porto físico pode fazer parte de múltiplas pontes virtuais. Para enviar o tráfego que entra nesse porto, é necessário mais do que o endereço do MAC de destino.

Associado a cada ponte virtual há um identificador único, conhecido por *Tag* da LAN virtual (ou VLAN). O tráfego que entra num porto é enviado baseado numa combinação do *Tag* da VLAN e do endereço do MAC de destino; o *Tag* da VLAN identifica a ponte virtual correcta, e o endereço do MAC de destino dirige a decisão de encaminhamento dentro da ponte.

A identificação virtual da ponte não é o único uso de um *Tag* de VLAN. As VLANs são usadas por vezes dentro das empresas para segregar redes locais em diferentes domínios de difusão. Cada VLAN é identificada por um único *Tag* dessa VLAN. As pontes virtuais preservam estes *Tag* da VLAN ao enviarem pacotes através das diferentes localizações, permitindo a criação de VLANs geograficamente distribuídas.

Um benefício importante do modelo virtual da ponte é que conserva portos de comutação. A combinação de pontes virtuais e de VLANs pode também ser usada para criar redes lógicas

seguras e separadas sobre um porto físico. O ISP pode criar uma ponte virtual e VLANs por subscritor e entregar o acesso de Internet a todos os subscritores fora de um porto de Ethernet no *router* de ponto extremo. O que consumiria centenas de portos no *router* de extremo pode agora ser feito com apenas um (figura 1.8).

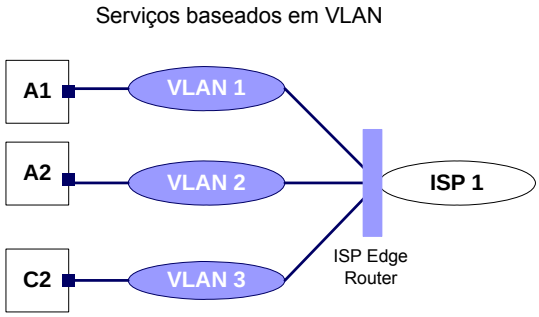


Figura 1.8 – Redes lógicas sobre um porto físico

O modelo de serviço do RPR suporta vários níveis de qualidade de serviço, identificados por parâmetros tais como a CIR, taxa de pico, potencialidade de *burst* e atraso no anel. Uma politica de serviço define então a qualidade de serviço atribuída ao tráfego quando transita através do anel de pacotes. Os diferentes níveis de serviço podem ser atribuídos aos diferentes tipos de tráfego. Uma política de serviço pode ser muito genérica, por exemplo, "todo o tráfego numa ponte virtual começa com um mínimo de largura de banda de 1Mb/s até 10 Mb/s", ou pode ser altamente específica, por exemplo, "tráfego HTTP começa com uma taxa cometida de 100 Kb/s até 100Mb/s".

1.6 Objectivos

Juntos, o novo paradigma de gestão de serviço, o modelo de serviço da Ethernet e a tecnologia RPR, dão forma aos três blocos de construção da arquitectura de rede *packet-based* da próxima geração na área metropolitana.

Neste contexto, este trabalho de mestrado irá contribuir para demonstrar a possibilidade de implementação de um sistema baseado na tecnologia RPR, em circuitos de lógica programável (FPGA). As unidades constituintes do MAC RPR foram desenvolvidas em linguagem SystemC passível de ser sintetizada em FPGA, e foram simuladas usando também código em SystemC.

A implementação daquelas unidades pressupôs o acompanhamento do desenvolvimento da norma IEEE 802.17. Durante esta fase foi feita a análise e estudo dos algoritmos das unidades do RPR e consequentemente a implementação em SystemC desses mesmos algoritmos.

1.7 Estrutura da dissertação

Neste capítulo foi feita uma breve introdução aos conceitos de gestão e modelo de serviço, e à tecnologia RPR.

O capítulo seguinte (capítulo dois) é composto por uma revisão das principais metodologias utilizadas para mapeamento de IP sobre diversos protocolos para posterior transporte na camada física. São também sumariamente abordados o sistema OSI, o protocolo SDH para o transporte de dados na camada física, e os protocolos IP e Ethernet.

No capítulo três é feita uma introdução à tecnologia RPR, seguida de uma previsão futura sobre alguns possíveis métodos e novas soluções para o transporte de IP. Seguidamente é feita uma apresentação breve de algumas soluções e produtos existentes no mercado, desenvolvidos por alguns fabricantes e operadores.

No capítulo quatro é apresentada uma breve descrição da futura norma do RPR (IEEE 802.17) e descrevendo também as opções de desenvolvimento das unidades do MAC RPR, segundo esta norma.

Nos capítulos cinco e seis é feita a descrição detalhada das opções de desenvolvimento dos trajectos de dados do MAC, dos algoritmos de *Fairness* e de descoberta da Topologia e Protecção, e da unidade de Operação, Administração e Manutenção, para um sistema real desta referida norma. São ainda discutidos os resultados de testes.

No capítulo sete são apresentadas as devidas conclusões.

2 Tecnologias habituais para transporte do protocolo IP

Neste capítulo são descritas algumas tecnologias usadas para o transporte de pacotes de dados IP/Ethernet sobre as diversas redes de telecomunicações.

A abordagem descrita nas seguintes secções está estruturada segundo o modelo OSI de forma a analisar primeiro o protocolo IPv4 [11-a] e IPv6 [11-b], em segundo os protocolos que o IP transporta, para posteriormente se analisar algumas das técnicas existentes para o transporte do IP nas camadas *Data Link* e *Physical*, em particular o POS e o LAPS [17].

Tendo o RPR como tecnologia alvo deste trabalho, são também analisados a Ethernet e o GFP [16]. A norma IEEE 802.17 especifica concretamente as interfaces destes dois protocolos para a camada física do MAC RPR. A Ethernet com diversas interfaces físicas completamente definidas, é actualmente um protocolo estabelecido no mercado até uma taxa de 1000Mb/s e com alguns fabricantes a iniciar o lançamento de produtos a 10Gb/s. Estas interfaces são definidas no IEEE 802.3 e usadas no IEEE 802.17 para adaptar as tramas RPR à camada física. O GFP por sua vez, é um protocolo que adapta genericamente protocolos à camada física e está referido no IEEE 802.17 como a adaptação das tramas RPR directamente ao SONET/SDH.

2.1 O Modelo OSI e o protocolo Internet

A figura 2.1 ilustra as sete camadas do modelo OSI [13] e uma instanciação do mesmo, orientada ao conjunto de protocolos IP. Neste caso na camada de *Data Link* é ilustrada a descrição do POS. Existem no entanto outras possibilidades de transportar IP abaixo da camada de *Network* e que serão abordados mais à frente.

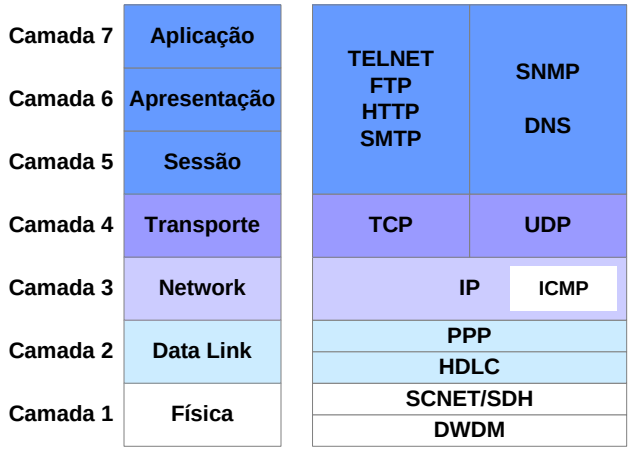


Figura 2.1 – Modelo ISO-OSI para transporte de tráfego *packet* sobre uma rede óptica

2.1.1 Camadas de Aplicação, Apresentação e Sessão

Na Tabela 2.1 as três camadas superiores, Aplicação, Apresentação e Sessão são consideradas uma única camada no caso dos protocolos IP, por uma questão de simplificação. As aplicações podem ser divididas num grupo que usa o TCP e noutro grupo que usa o protocolo UDP, para transporte dos datagramas IP. Existem correntemente múltiplas aplicações de comunicação no conjunto do protocolo IP.

Aplicação do Utilizador	Protocolo	IETF
E-mail	Simple Mail Transfer Protocol - **SMTP**	RFC821
Para cópia e transferência de ficheiros	File Transfer Protocol - **FTP**	RFC959
Login a um terminal remoto	**Telnet**	RFC854
Troca de informação WWW	Hyper Text Transfer Protocol - **HTTP**	RFC1945
Para aplicações de gestão simples de rede	Simple Network Management Protocol – **SNMP**	RFC1157
Identificação do endereço IP de um *Host*	Domain Name System - **DNS**	RFC2929, RFC1591

Tabela 2.1 – Protocolos de aplicação e seu uso

2.1.2 Camada de Transporte

Tal como referido acima na camada de transporte há dois protocolos principais que se distinguem principalmente pela sua fiabilidade e por menor *overhead* no transporte, o TCP e o UDP.

O UDP é o mais simples dos dois. É tipicamente usado quando a fiabilidade e a segurança são menos importantes do que a velocidade e o tamanho do *overhead*. É um protocolo de ligação *extremo-a-extremo* que adiciona apenas os endereços, a soma de controlo de erro e a informação de comprimento dos dados vindos da camada superior.

O TCP fornece serviços completos de fiabilidade e controlo de fluxo, da camada de transporte às aplicações. É um protocolo *connection-oriented*, o que significa que deve ser estabelecida uma ligação entre dois pares antes que qualquer um possa transmitir dados. Ao efectuar isto o TCP estabelece uma ligação virtual entre o remetente e o receptor, e que permanece activa durante toda a transmissão. O TCP é um protocolo fiável, significa que o receptor reconhece cada transmissão de dados. Se o remetente não receber o reconhecimento dentro de um tempo especificado, então retransmite esses mesmos dados.

	UDP	TCP
Características	• Protocolo *Connectionless* • Não recuperação de erros • Não fiável • Alta velocidade • Baixo *overhead*	• Protocolo *Connection-oriented* • Detecção e recuperação de erros • Fiável • De menor velocidade que o UDP • Elevado *overhead*
Aplicações	• Voz • Broadcasting • Etc.	• HTTP • FTP • Etc.

Tabela 2.2 – Comparação entre os protocolos UDP e TCP

2.1.3 Camada Network

Desde a camada de transporte os datagramas TCP ou UDP são encapsulados em pacotes IP para serem entregues à camada de *Network*.

2.1.3.1 O Protocolo Internet – versão 4 – Ipv4

O IP é projectado para o uso em sistemas interligados de redes de comunicações *packet-switched*. Providencia a transmissão de blocos de dados (datagramas), desde uma origem a um destino, onde as fontes e os destinos são *hosts* identificados por endereços de comprimento fixo.

É um protocolo funcionalmente limitado ao âmbito do fornecimento das funções necessárias para entregar um datagrama de uma fonte a um destino, sobre um sistema interligado de redes. Não há qualquer mecanismo para aumentar a fiabilidade dos dados *extremo-a-extremo*, controlo de fluxo, sequenciamento, ou outros serviços existentes geralmente em protocolos *host-to-host*. Não

possuindo competências ao nível da camada de transporte, pode no entanto basear-se nos serviços das suas redes de suporte para fornecer vários tipos e qualidades de serviço.

Este protocolo é transportado por protocolos *host-to-host* num ambiente *internet*. Transporta protocolos de rede locais que por sua vez transportam o datagrama Internet à *gateway* ou ao *host* de destino seguintes.

O IP implementa duas funções básicas: endereçamento e fragmentação.

Os módulos de rede usam os endereços transportados no cabeçalho do IP para transmitir os datagramas até aos seus destinos. Usam também, quando necessário, os campos no cabeçalho do IP para fragmentar e reconstruir os datagramas IP, para a transmissão através de redes *small packet*. A selecção de um trajecto para a transmissão é chamada *routing*.

Os erros detectados podem ser relatados através do ICMP [12]. O ICMP é usado por *hosts* e por *gateways* para informar o remetente de problemas no datagrama, e para trocar mensagens de controlo. O ICMP usa o IP para entregar estas mensagens. Embora o IP seja um protocolo não fiável e *connectionless*, no entanto, o ICMP usa o IP para informar o remetente se um datagrama não foi entregue. O ICMP apenas relata problemas, não corrige erros.

Por exemplo: destino inalcançável, redireccionamento do pacote ou tempo excedido, são os tipos de mensagens de erro que poderiam ser emitidas e, por exemplo, um pedido e resposta de eco (ping) é um tipo de mensagem de informação que pode ser emitido.

As redes IP são projectadas geralmente numa estrutura em malha, de modo que se ocorrer uma interrupção na transmissão, possa ser encontrado rapidamente um trajecto alternativo. O processo de distribuição é um procedimento dinâmico, que permite que o sistema envie um pacote através do trajecto mais eficiente da origem ao endereço de destino.

Os *routers* são elementos de rede, que operam na camada de rede do modelo OSI. Têm acesso aos endereços da camada de rede e contêm *software* que lhes permite determinar qual de diversos trajectos possíveis dentro daqueles endereços, é o melhor para uma transmissão em particular. Direccionam os pacotes entre as múltiplas redes inter-ligadas. Distribuem pacotes de uma rede a qualquer número de redes de destino numa *Internet*.

Os *switches* são elementos da camada 2 da rede. São responsáveis por estabelecer ligações estáticas provisórias entre dois ou mais dispositivos ligados ao *switch*. Cada *switch* é ligado às múltiplas ligações existentes e usado para complementar as ligações entre os pares comunicantes. Assim que os *switches* forem ajustados, ficam estáticos para um processo específico de transmissão e podem então ser reconfigurados mais tarde.

2.1.3.2 Protocolo Internet – versão 6 – IP-v6

O IP versão 6 (IPv6) é uma nova versão do protocolo Internet, projectado como sucessor do IP versão 4 (IPv4).

As diferenças do IPv4 para o IPv6 são englobadas essencialmente nas seguintes categorias:

No IPv6 o tamanho do endereço IP é aumentado de 32 bits para 128 bits, para suportar mais níveis de endereçamento hierárquico, um número muito maior de nós endereçáveis e uma auto configuração mais simples dos endereços. A escalabilidade de *routing multicast* é melhorada, adicionando um campo de *scope* aos endereços de *multicast*. E é definido um novo tipo de endereço chamado *anycast address*, usado para emitir um pacote a qualquer nó de um grupo de nós.

Alguns campos do cabeçalho do IPv4 foram abandonados ou tornados opcionais, para reduzir o custo do processamento do pacote e para limitar o custo da largura de banda do cabeçalho do IPv6. As alterações na forma em como as opções do cabeçalho IP são codificadas, permitem um envio mais eficiente, limites menos restritos no comprimento das opções e uma flexibilidade maior para introduzir novas opções no futuro.

Foi adicionada uma nova potencialidade para permitir etiquetar os pacotes que pertencem ao tráfego particular tipo *flows*, para o qual o remetente requer manipulação especial (tal como qualidade de serviço *non-default* ou serviço *real-time*).

IPv6 header Next Header = TCP	TCP header + data

IPv6 header Next Header = Routing	Routing header Next Header = TCP	TCP header + data

IPv6 header Next Header = Routing	Routing header Next Header = Fragment	Fragment header Next Header = TCP	fragmento de TCP header + data

Figura 2.2 – Exemplo de *extension headers* com *next headers*

No IPv6, são especificadas (opcionalmente) extensões para suportar a autenticação, a integridade e a confidencialidade dos dados. Esta informação opcional é codificada em cabeçalhos separados que podem ser colocados entre o cabeçalho IPv6 e o cabeçalho da camada superior, num pacote. Há um número pequeno de cabeçalhos de extensão, cada um identificado por um valor distinto de "cabeçalho-seguinte". Um pacote IPv6 pode transportar nenhum, um, ou mais cabeçalhos de extensão, cada um identificado pelo campo cabeçalho-seguinte do cabeçalho precedente (figura 2.2).

Com uma excepção, os cabeçalhos extensão não são examinados nem são processados por qualquer nó ao longo do trajecto de entrega de um pacote, até que o pacote alcance o nó (ou cada um do conjunto de nós, no caso de *multicast*) identificado no campo de endereço de destino do cabeçalho IPv6.

Assim, a desmultiplexagem normal no campo *Next Header* do cabeçalho do IPv6 invoca o módulo para processar o primeiro cabeçalho da extensão, ou o cabeçalho da camada superior se nenhum cabeçalho extensão estiver presente. Os índices e a semântica de cada cabeçalho extensão determinam se deve ou não prosseguir para o cabeçalho seguinte. Consequentemente, os cabeçalhos extensão devem ser processados estritamente na ordem em que aparecem no pacote; um receptor não deve, por exemplo, percorrer através de um pacote à procura de um tipo particular de cabeçalho extensão e processar esse cabeçalho antes de processar todos os precedentes.

A excepção referida acima é o cabeçalho das opções de *Hop-by-Hop*, que transporta a informação que deve ser examinada e processada por cada nó ao longo do trajecto de entrega de um pacote, incluindo a fonte e os nós de destino. O cabeçalho das opções de *Hop-by-Hop*, quando presente, deve seguir imediatamente o cabeçalho IPv6. A sua presença é indicada pelo valor zero no campo *Next Header* do cabeçalho do IPv6.

Uma implementação completa do IPv6 inclui a execução dos cabeçalhos extensão de opções de *hop-by-hop*, de *routing* (Tipo 0), de fragmentação, de opções de destino, de autenticação e de encapsulamento seguro do *payload*.

2.1.4 Transporte nas camadas *Data Link* e Física

Desde a camada de *Network*, os pacotes IP necessitam ser entregues às camadas subjacentes de *Data link* e Física. Esta transmissão até à camada física pode ser conseguida através de trajectos alternativos. Na figura 2.3 são ilustrados dois possíveis trajectos de mapeamento de datagramas IP em SDH/SONET e que são transportados posteriormente por fibra óptica.

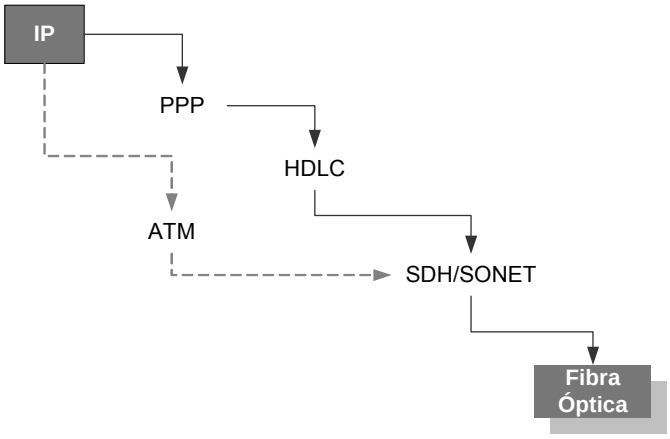


Figura 2.3 – Trajectos alternativos para entregar pacotes IP à fibra

O trajecto descrito pela linha contínua na figura 2.3 é o recomendado pelo IETF e é também um dos possíveis trajectos definidos no esboço do protocolo 802.17 do IEEE. Este trajecto, tal como ilustrado na figura 2.3, será portanto analisado neste documento.

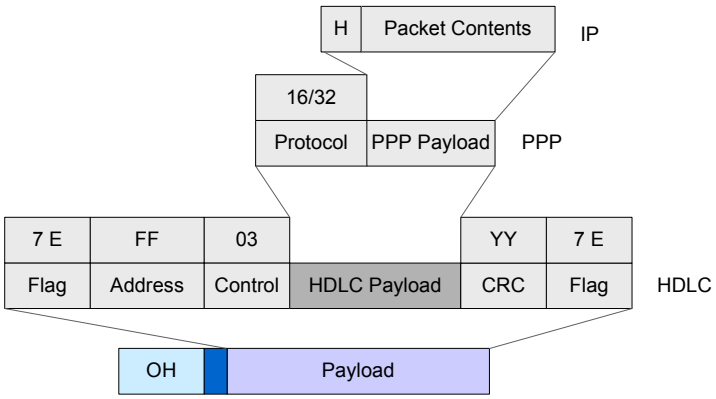


Figura 2.4 – Mapeamento de pacotes IP em tramas SONET/SDH

Neste trajecto definido como POS os pacotes IP são encapsulados no protocolo PPP, que por sua vez é mapeado numa trama HDLC-like. A trama HDLC-like por sua vez é mapeada numa trama SONET/SDH (figura 2.4).

2.2 SDH - Synchronous Digital Hierarchy

Um dos motivos para o desenvolvimento do SDH foi o de providenciar uma trama com uma estrutura que permitisse a transmissão dos sinais PDH com canais tributários de 64 Kb/s, especificados pelas hierarquias ANSI (1,5 Mb/s, 6 Mb/s e 45 Mb/s) e CEPT (2 Mb/s, 8 Mb/s, 34 Mb/s e 140 Mb/s), [36].

A camada SDH fornece ligações do tipo comutação de circuitos extremo-a-extremo e possui um mecanismo eficiente para multiplexar ligações de baixo ritmo (ex. 155 Mb/s), de modo a obter ligações de ritmo mais elevado (ex. 10 Gb/s) para que estas sejam transportadas de um modo mais eficiente pela rede. Os elementos de rede do SDH incluem terminais de linha, ADMs, regeneradores e DCSs. Os terminais de linha multiplexam e desmultiplexam os fluxos de tráfego de dados. Os ADMs são utilizados em configurações de rede lineares e em anel. Estes dois elementos de rede disponibilizam um método eficiente para extrair parte do tráfego incidente num nó, permitindo simultaneamente a passagem directa do restante tráfego para a saída. Os

regeneradores regeneram o tráfego SDH sempre que necessário. Os DCSs são utilizados em nós maiores para comutar um grande número de fluxos de tráfego.

Nos nós intermédios, o SDH disponibiliza um método eficiente para extrair fluxos de dados de ritmos mais baixos, a partir de fluxos de dados a ritmos mais elevados. Esta característica é conseguida com a utilização de um mecanismo de ponteiros, isto será analisado mais à frente.

O SDH garante um elevado grau de fiabilidade e disponibilidade. Este objectivo é conseguido com a implementação de mecanismos que garantem o restauro rápido do serviço no caso de avarias na rede.

O SDH inclui um considerável *overhead*, que acrescenta funcionalidade de monitorização e gestão de rede.

2.2.1 As hierarquias SONET/SDH e PDH

As comunicações eram, tradicionalmente, efectuadas em hierarquias PDH. A figura 2.5 ilustra a comparação entre as diversas hierarquias PDH existentes na Europa, EUA e Japão. Na Europa o PDH parte de um ritmo síncrono básico de 2Mb/s passando a três níveis assíncronos até um máximo de 140Mb/s. Nos EUA o PDH parte de um ritmo síncrono básico de 1,5Mb/s tendo três níveis assíncronos até 45Mb/s, no entanto continua a subir de Nx45Mb/s (N número inteiro). No Japão para o PDH há dois ritmos síncronos básicos, um de 1,5Mb/s e outro de 4x1,5=6,3Mb/s, no entanto o PDH sobe três níveis até um máximo de 400Mb/s. Estes três ritmos básicos de 1,5Mb/s, 2Mb/s e 6,3Mb/s são encapsulados numa hierarquia síncrona com um ritmo básico de 155Mb/s subindo de Nx155Mb/s (N número inteiro).

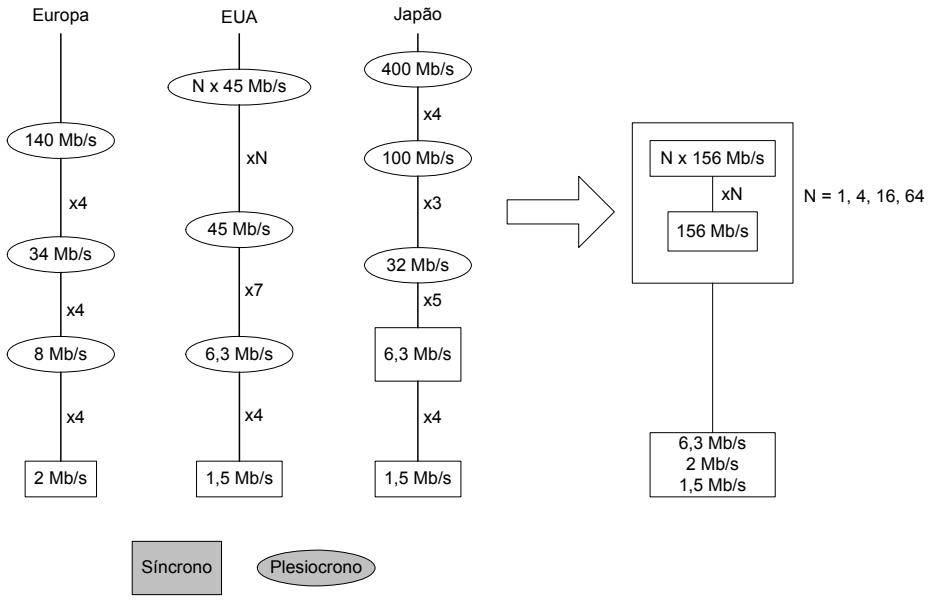


Figura 2.5 – Hierarquias PDH existentes na Europa, EUA e Japão

A tabela 2.3 ilustra a comparação das hierarquias do SONET com as do SDH. A hierarquia europeia SDH é funcionalmente equivalente a parte da norma norte-americana SONET. Para o SDH as designações STM-N correspondem a uma divisão por três das designações STS-N e OC-N que designa um sinal STS-N depois de baralhado e convertido para o domínio óptico, do SONET.

Ritmos (Mb/s)	SONET (STS/OC)	SDH (STM)
51,28	1	-
155,52	3	1
622,08 (4x155,52)	12	4
1244,16 (2x622,08)	24	8
2488,32 (2x1244,16)	48	16
9953,28 (4x2488,32)	192	64
39813,12 (4x9953,28)	768	256

Tabela 2.3 – Comparação entre as hierarquias do SONET e do SDH

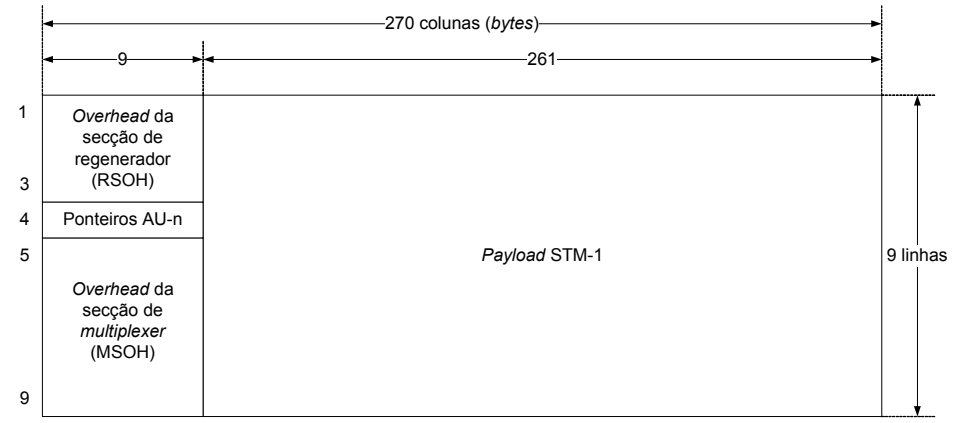
A tabela 2.4 ilustra as três hierarquias digitais PDH existentes na Europa, EUA e Japão. Observa-se que os tributários de baixa largura de banda, para o utilizador, são todos em PDH.

Europa		EUA		Japão
Designação	Ritmo (Mb/s)	Designação	Ritmo (Mb/s)	Ritmo (Mb/s)
E-0	0,064	DS0	0,064	0,064
		DS1	1,544	1,544
E-1	2,048			
		DS1C	3,152	3,152
		DS2	6,312	6,312
E-2	8,448			
E-3	34,368			
		DS3	44,736	
		DS3C	91,053	
				97,728
E-4	139,264			
		DS4	274,176	
				397,2

Tabela 2.4 – Hierarquias digitais PDH existentes na Europa, EUA e Japão

2.2.2 A trama do SDH

A figura 2.6 ilustra a estrutura da trama STM-1 que é a trama mais básica de SDH.



RSOH – *Regenerator Section Overhead*

MSOH – *Multiplexer Section Overhead*

AU – *Administrative Unit*

Figura 2.6 – Estrutura da trama STM-1

A figura 2.7 ilustra a formação do sinal STM-4, usando o método de multiplexagem *standard* do SDH. Quatro sinais STM-1 são agrupados byte a byte num sinal STM-4, método designado por *byte-interleaving* (entrelaçamento de *bytes*).

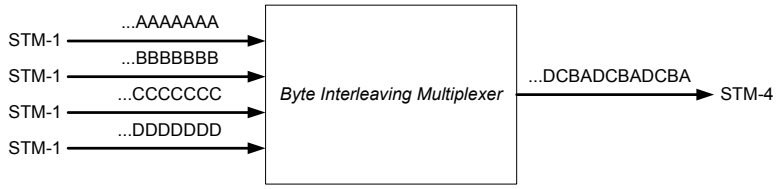


Figura 2.7 – Formação do sinal STM-4 a partir de um STM-1

Além da multiplexagem de dados, é também usada uma nova trama STM-4. A figura 2.8 ilustra a estrutura da trama STM-4, que é construída de novo, com um formato semelhante ao descrito acima, mas com um número maior de colunas.

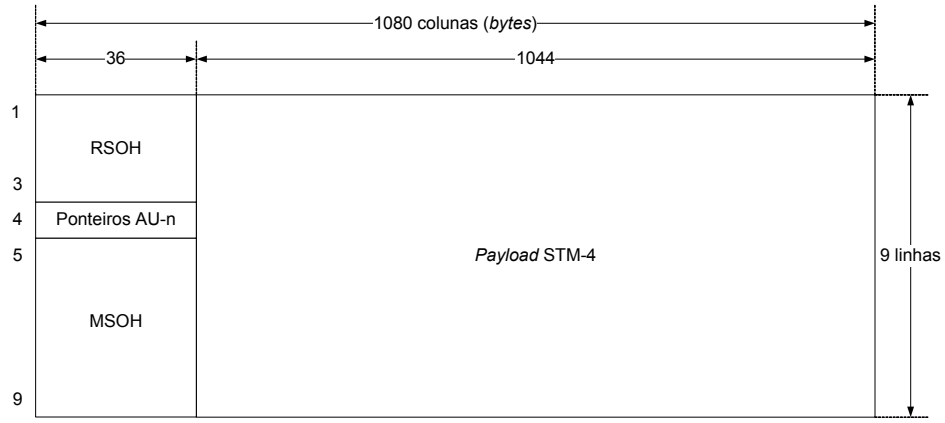


Figura 2.8 – Estrutura da trama STM-4

De uma forma geral a estrutura de uma trama STM-N é composta por NxSTM-1, construída pelo mesmo método de *byte-interleaving* tal como ilustrado na figura 2.9.

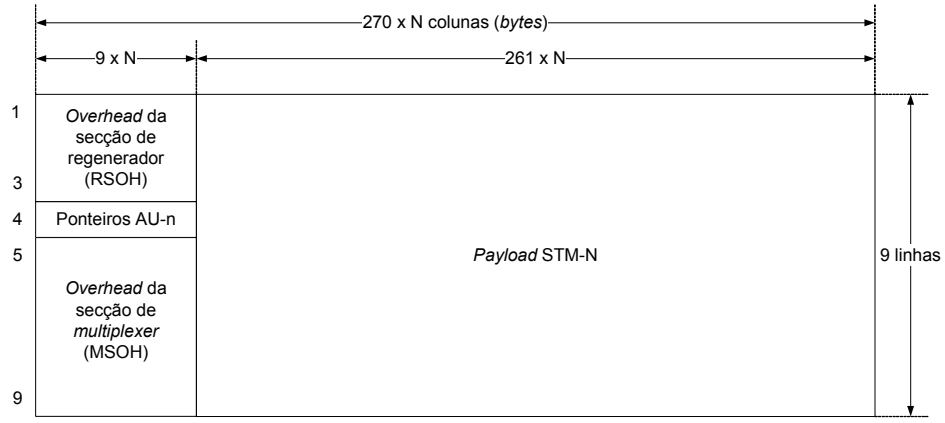


Figura 2.9 – Estrutura da trama STM-N

2.2.3 Multiplexagem Sub-SDH

Na relação entre os elementos de multiplexagem o SDH utiliza VCs para acomodar sinais não SDH, de mais baixo ritmo. Na estrutura de multiplexagem para sinais sub STM-1 são definidos cinco tamanhos de VCs como ilustrado na tabela 2.5.

Tipo de sinal SDH / assíncrono	Designação do VC
E4 (139,264 Mb/s)	VC-4
E3 (34,368 Mb/s) DS3 (44,736 Mb/s)	VC-3
E2 (8,448 Mb/s)	VC-2
E1 (2,048 Mb/s)	VC12
DS1 (1,544 Mb/s)	VC-11

Tabela 2.5 – Contentores virtuais para sinais sub-SDH

O SDH define uma hierarquia com duas etapas. Os sinais VC-2, VC-11 e VC-12 são primeiramente multiplexados em VC-3 ou em VC-4. Em seguida os VC-3 e VC-4 são multiplexados para um sinal STM-1.

Na figura 2.10 está ilustrada a relação entre os elementos de multiplexagem. Observa-se que um AUG é equivalente a um AU-4. Designa-se a mesma entidade por dois nomes diferentes pois um AUG pode ser transportado por um AU-4 ou três AU-3. Sob o ponto de vista desta última aplicação, o AUG é formado por um grupo de três AU-3.

Definem-se portanto dois AUs. AU-4 → VC-4 mais ponteiro AU e AU-3 → VC-3 mais ponteiro AU. O VC-*n* associado a cada AU-*n* não tem uma posição fixa dentro da trama STM-*N*.

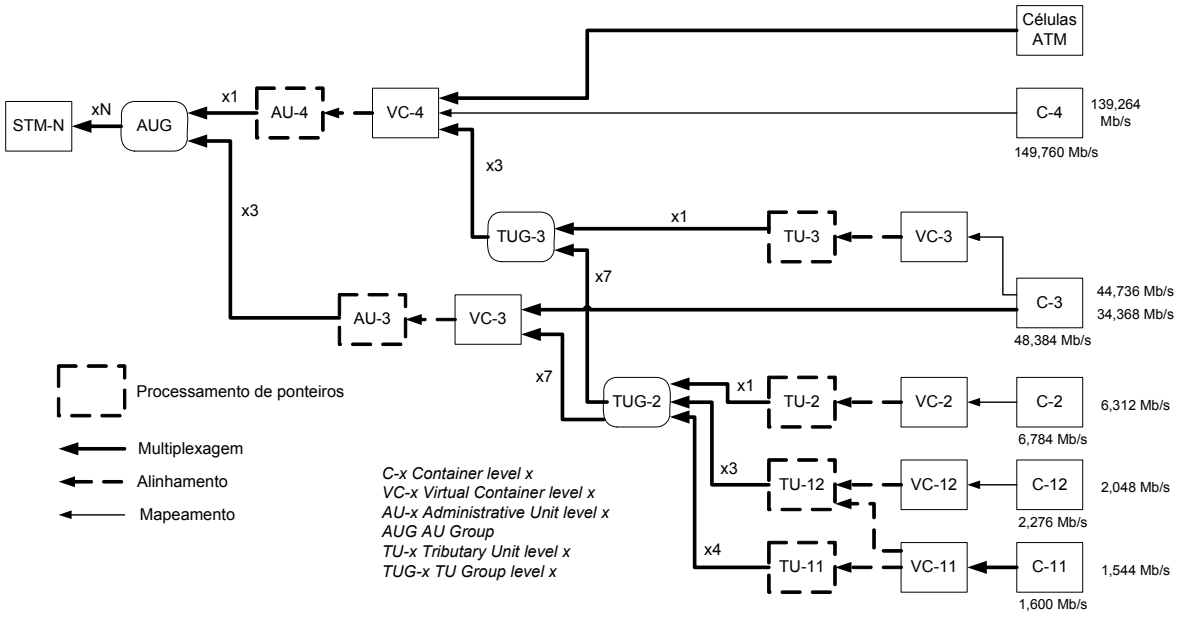


Figura 2.10 – Relação entre os elementos de multiplexagem – tributários associados com recipientes (*containers*) *C-x* (Norma ITU-T G.702)

Por exemplo na formação de uma trama STM-1 (figura 2.11) a partir de um sinal PDH E-4, o contentor C-4 foi concebido para transportar o sinal PDH E-4 de 139,264 Mb/s.

Ao contentor C-4 são adicionados *overheads* de caminho para formar o contentor virtual VC-4.

Na formação da trama STM-1 a partir de um sinal PDH E-4, os ponteiros AU-4 são adicionados ao contentor virtual VC-4 para formar uma AU-4. Esta unidade administrativa é equivalente a um AUG.

Por fim na formação da trama STM-1 a partir de um sinal PDH E-4, adicionam-se os RSOH e MSOH ao AUG para formar a trama STM-1.

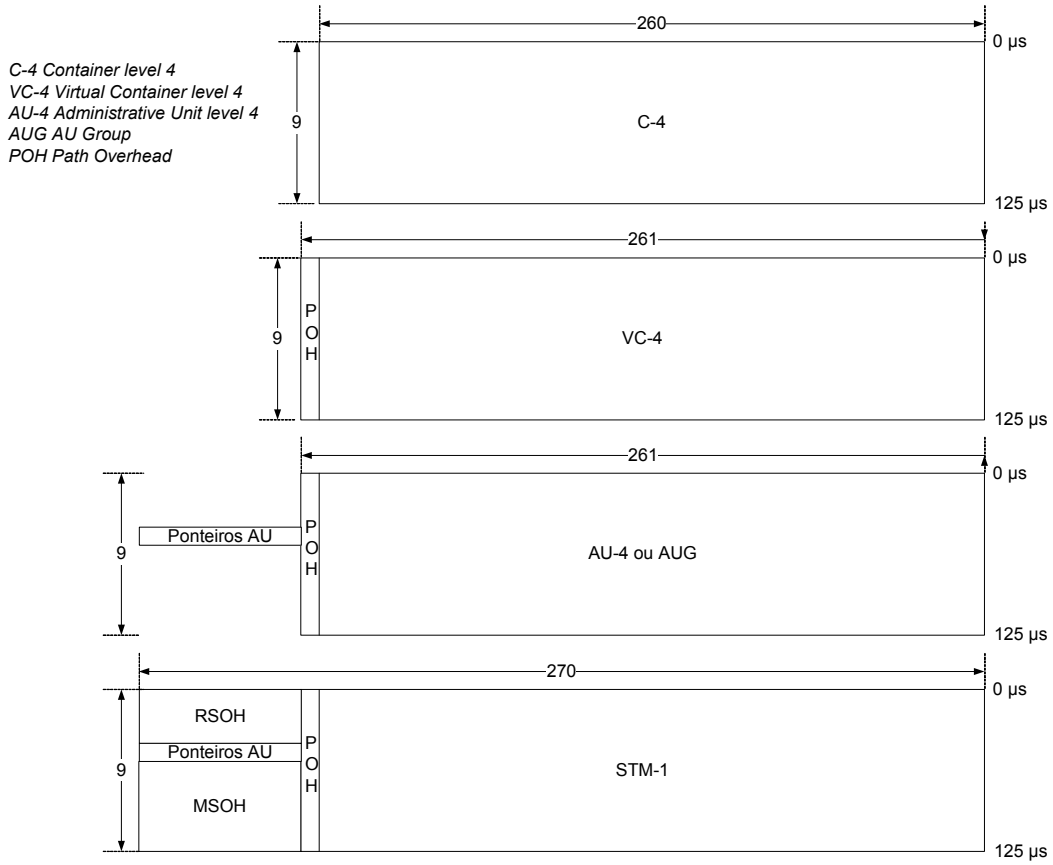
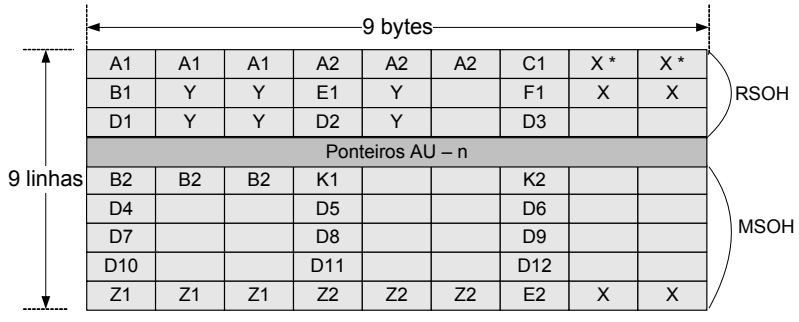


Figura 2.11 – Formação de uma trama STM-1 a partir de um sinal PDH E-4

2.2.4 *Overhead* de Secção STM-1 - Ponteiros

As primeiras nove colunas de uma trama STM-1 têm a seguinte composição como ilustrado na figura 2.12.



X – bytes reservados para uso nacional
Y – bytes dependentes do meio
* - bytes não baralhados
Os bytes não definidos destinam-se a utilização internacional futura

Figura 2.12 – Composição dos RSOH, MSOH e AUG

O ponteiro AU indica o *offset* do início do *payload* em relação ao início da trama da secção de *multiplex*. Como se viu anteriormente, uma AU é constituída por um VC de ordem superior mais um ponteiro AU.

O ponteiro AU-*n* indica a localização do primeiro byte do VC-*n*, e situa-se numa posição fixa na trama STM-*N* (linha 4 da trama STM-*N*). Com o sistema de ponteiros, o VC-*n* pode começar em qualquer ponto do *payload* da AU-*n*. De facto, em geral um datagrama de dados inicia-se numa trama SDH e termina na trama seguinte.

É usado o mecanismo de ponteiros porque em geral, as tramas que chegam a um nó provenientes de outros nós, estão desfasadas devido aos diferentes atrasos de transmissão. Para alinhar estas tramas podem-se utilizar memórias de trama. Este método tem desvantagens pois são introduzidos atrasos médios de meia trama e atrasos máximos de uma trama. Consequentemente para ritmos de transmissão elevados, as memórias de trama deverão ter grande capacidade.

Ao permitir a localização do VC-*n* em qualquer ponto da trama SDH, não é necessário o uso de memórias significativas.

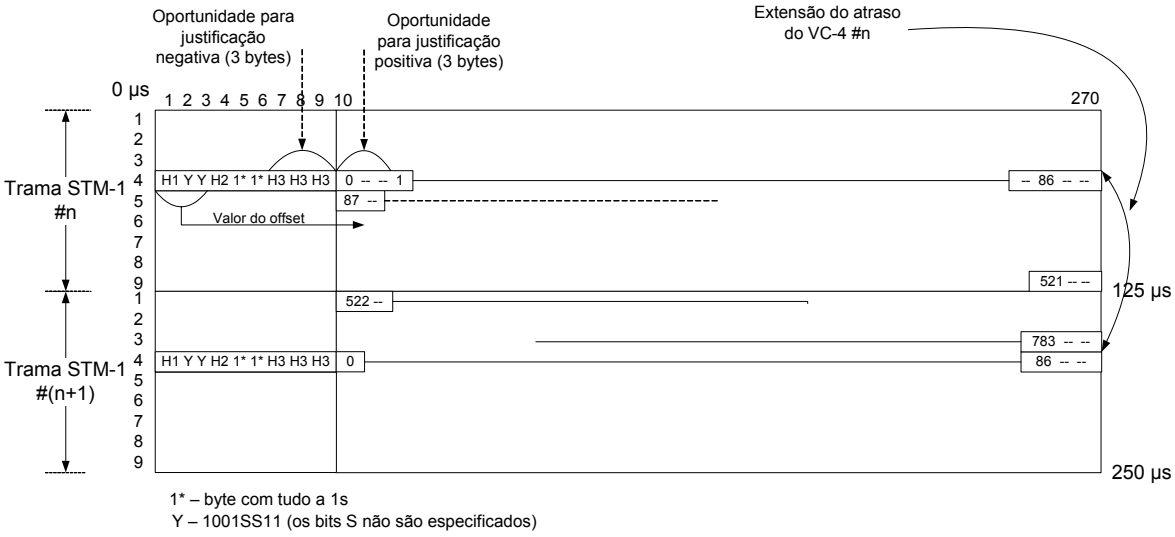


Figura 2.13 – Mecanismo de ponteiros do SDH

O ponteiro AU-4 está contido nos dois bytes H1, H2 indicados na figura 2.14. O valor deste ponteiro vai de 0 a 782, podendo então indicar *offsets* com incrementos de três bytes,

[(261 colunas * 9 linhas) bytes – 3bytes]/3=782

Quando há uma diferença de frequência entre o ritmo das tramas e o ritmo dos VC-*n* o valor do ponteiro varia para acomodar essa diferença. Utiliza-se os bytes adequados da trama para justificação positiva/negativa.

Com os ponteiros AU-4 e TU-3 quando um sinal STM-1 é utilizado para transportar um sinal C-4 (transportando um sinal E4 ou equivalente, através de um VC-4), é apenas necessário um conjunto de bytes para o ponteiro (H1 e H2). Isto porque apenas se pode mapear um contentor C-4 num *payload* STM-1. No entanto, se o sinal STM-1 for utilizado para transportar três sinais E-3, são necessários três ponteiros. Isto porque um VC-4 pode ser também utilizado para transportar três contentores C-3 (contendo três sinais E-3).

Os três conjuntos de ponteiros são agora designados por ponteiros TU-3. A sua localização na estrutura do VC-4 é indicada na figura 2.14-B.

Cada conjunto de ponteiros (H1, H2), situados nas colunas 3, 4 e 5, indica a posição de cada um dos C-3s dentro do VC-4.

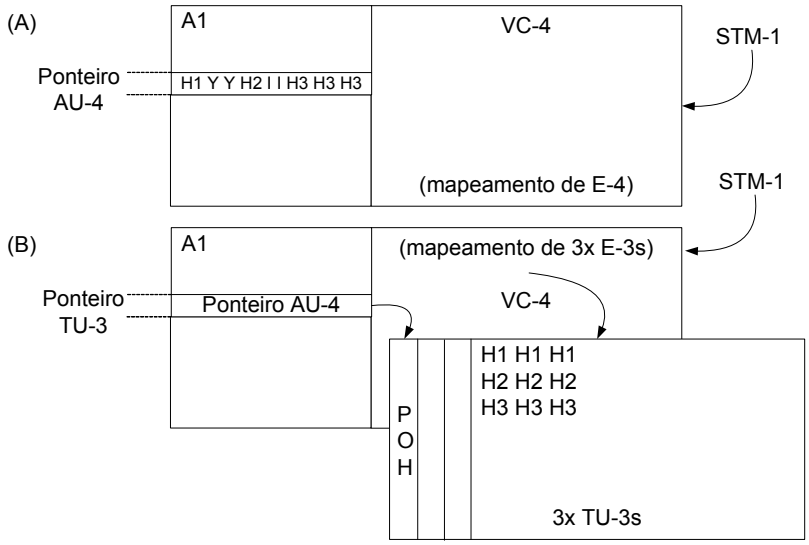


Figura 2.14 – Ponteiros do AU-4 e TU-3

Para a trama STM-1 os *overheads* de transporte têm a estrutura indicada na figura 2.14. Os bytes A1 e A2 destinam-se a garantir o sincronismo de trama e têm a forma fixa (A1, A2)=(1111 0110 0010 1000). O byte B1 é utilizado para a monitorização de erros pela camada da secção do regenerador e é aplicada paridade par. O B1 é calculado sobre todos os bits da trama STM-N anterior depois de baralhada. O valor de B1 é colocado apenas na posição B1 da primeira trama STM-1 (pertencente a um sinal STM-N), antes de baralhada, assim, as outras colunas de um sinal STM-4 que deveriam transportar a informação B1, encontram-se vazias.

Para a trama STM-4 será então como ilustrado na figura 2.15.

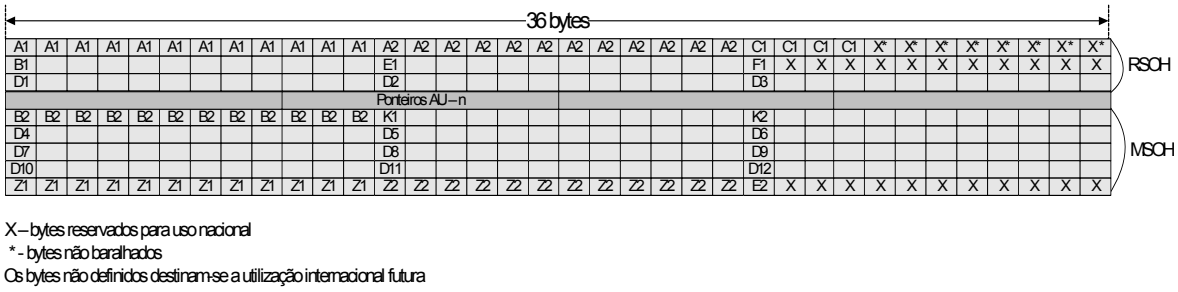


Figura 2.15 – *Overheads* de transporte para a trama STM-4

Alguns campos de *overhead* do STM-4 são em número igual ao do caso STM-1. Contudo, muitos desses bytes quadriplicam no caso do STM-4.

Se se necessitar de capacidade de *payload* superior à capacidade de 149,76 Mb/s, do VC-4, podem-se concatenar vários AU-4s para formar um AU-4-Xc. Nesse caso o ponteiro AU-4 contém indicação de concatenação e o *payload* C-4 múltiplo, transportado num único VC-4-Xc, é tratado como uma entidade única através da rede.

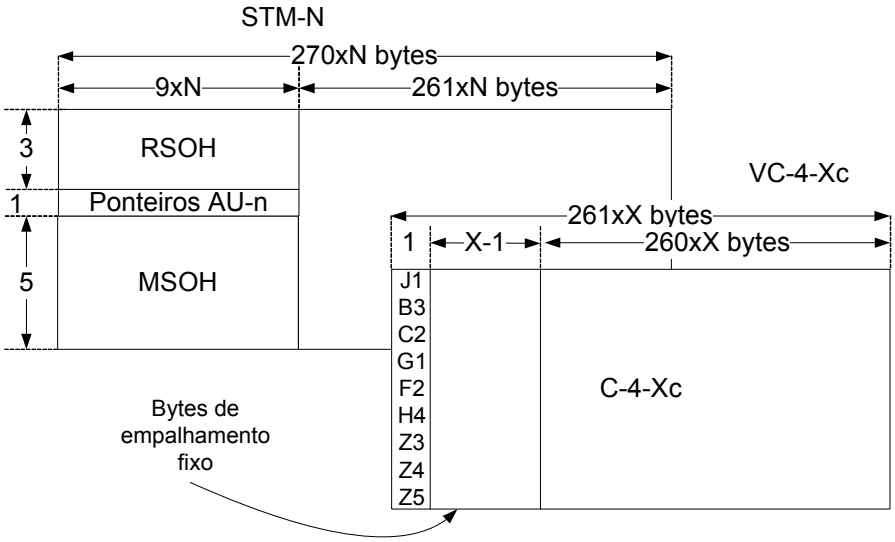


Figura 2.16 – Formação de um VC-4-Xc

Os VC-4-Xc possuem um *overhead* de caminho (POH), contido na primeira coluna.

O contentor único transporta também bytes de enchimento fixos em (X-1) colunas

2.3 POS - Packet over SONET/SDH

O POS permite o transporte de datagramas IP sobre o SONET/SDH. Providencia um serviço eficiente de tráfego IP ponto-a-ponto para altas velocidades e pode suportar voz, vídeo e tráfego de dados simultaneamente. Usa a infra-estrutura SONET/SDH existente, com todas as suas vantagens tais como fiabilidade, escalabilidade e versatilidade. Oferece também vantagens significativas ao providenciar uma utilização mais eficiente da largura de banda devido a um menor *overhead* relativamente ao ATM sobre SONET/SDH, porque não tem esta camada extra. Reutiliza também os protocolos usuais da *Link-Layer*.

2.3.1 PPP - Point-to-Point Protocol

O PPP fornece um método normalizado para transportar datagramas de diferentes protocolos (por exemplo pacotes IP) sobre ligações ponto-a-ponto. Como a tecnologia SONET/SDH é orientada a ligações ponto-a-ponto, o PPP/HDLC é adequado para ser usado sobre estas linhas.

O PPP fornece uma solução comum para a fácil ligação de uma grande variedade de *hosts*, *bridges* e *routers*. A estrutura do PPP permite a multiplexagem de diferentes protocolos da camada de Network (IP, IPX, AppleTalk, etc.) simultaneamente sobre a mesma ligação.

Algumas das características da ligação PPP são a de permitir a operação *full-duplex* e simultânea, a não imposição de qualquer limitação relativamente às taxas de transmissão, a operação através da maioria dos DTE, o suporte de múltiplas interfaces (por exemplo RS232, RS422, V.35, etc.) para o equipamento DTE/DCE e não requerer quaisquer sinais de controlo de *hardware* (por exemplo Request to Send, etc.).

O PPP consiste em três componentes principais. i) Um método para o encapsulamento de datagramas de diferentes protocolos (encapsulamento do PPP) e emulação do PPP; ii) Um protocolo de controlo da ligação (LCP) para estabelecer, configurar e testar a ligação do canal de dados; iii) E uma família de protocolos de controlo da rede (NCPs) para estabelecer e configurar protocolos de diferentes camadas da mesma.

Para permitir que datagramas de diferentes protocolos sejam transmitidos através de uma ligação, tem de ser usada uma estrutura que permita ser tomada uma decisão não ambígua entre os vários

protocolos. Para tal, o encapsulamento PPP (figura 2.17) inclui um campo do protocolo que consiste em 1 ou 2 octetos, e cujo valor identifica o datagrama encapsulado.

Protocolo 8/16 bits	**Payload/Dados**	**Padding**

Figura 2.17 – Encapsulamento PPP de acordo com a RFC1661

O campo de dados tem um comprimento de zero ou mais octetos e contém o datagrama para o protocolo especificado no campo de protocolo. O comprimento máximo, incluindo *padding* (mas não o campo de protocolo) é de 1500 octetos. Isto significa que, se por exemplo for transportado um datagrama IP com um *payload* maior, este *payload* é fragmentado em diversos pacotes para caberem no tamanho do campo de dados. O campo de enchimento (*Padding*) tem a função de no começo da transmissão poder ser enchido (*padded)* com um número arbitrário de octetos (até 1500 octetos) e é da responsabilidade do protocolo distinguir o *padding* da informação real.

Para se estabelecer uma ligação PPP são usados dois protocolos. Um protocolo de controlo da ligação (LCP) e um protocolo de controlo da rede (NCP).

O LCP é responsável por estabelecer correctamente, configurar e testar o PPP. Antes que os datagramas IP possam ser transportados através de uma ligação PPP, cada uma das interfaces PPP ligadas têm que trocar entre si uma série de pacotes LCP.

A informação do LCP é emitida na forma de datagramas PPP. Estes datagramas são classificados como (RFC1661): pacotes de configuração da ligação que são usados para estabelecer e configurar uma ligação; pacotes de terminação da ligação que são usados para terminar uma ligação; e pacotes de manutenção da ligação que são usados para controlar e eliminar erros de uma ligação.

O NCP permite a preparação e a configuração de protocolos diferentes para funcionamento nas várias camadas de rede. Este protocolo permite a activação, a configuração e a desactivação dos módulos do protocolo IP em ambos os lados da ligação ponto-a-ponto. Como no LCP, estas funções são conseguidas através da troca de pacotes especiais de dados.

Quando os protocolos LCP e NCP são executados com sucesso então os dados podem ser transmitidos através da ligação ponto-a-ponto.

2.3.1.1 HDLC - High-Level Data Link Control

O PPP fornece um método padrão para transporte de datagramas de diferentes protocolos sobre ligações ponto-a-ponto. Os pacotes PPP são então encapsulados em tramas HDLC-like [9] que por sua vez são mapeadas na trama SONET/SDH.

A única imposição aos requisitos do PPP é a implementação de circuitos *full-duplex*, tanto dedicados ou comutadas por circuito, que podem operar tanto assincronamente (start/stop), sincronizadas ao bit ou ao octeto, transparente às tramas da camada de *Data Link* do PPP. A especificação do HDLC-like permite o mapeamento de tramas sobre ligações síncronas orientadas ao bit ou ao octeto, e sobre ligações assíncronas com dados de 8 bits e sem paridade. Estas ligações devem ser *full-duplex*, mas podem ser dedicadas ou comutadas por circuito.

2.3.1.2 Mapeamento dentro da trama SONET/SDH

O processo de mapeamento do PPP sobre o SONET/SDH consiste inicialmente na geração do FCS da trama PPP/HDLC, seguido da adição de bits de *stuffing*, seguido do baralhar dos bits e finalmente o mapeamento sobre as tramas SONET/SDH (figura 2.18).

O PPP usa o FCS para a detecção de erros. Isto é geralmente implementado em *Hardware*. A extremidade dos campos de informação e de *padding* é encontrada ao localizar a sequência da *Flag* de fecho e removendo o campo do FCS.

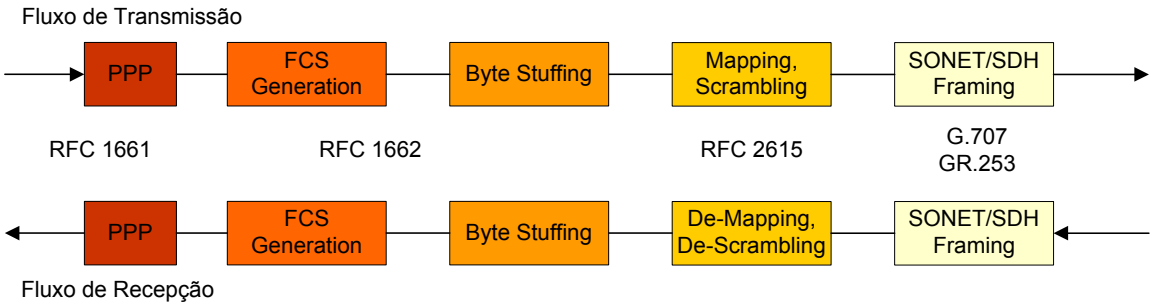


Figura 2.18 – Diagrama do processo de transmissão e recepção de um pacote IP na trama SONET/SDH

O protocolo PPP não coloca quaisquer limitações ao uso de determinados padrões de bits no pacote da camada de rede. É transparente a qualquer tipo de dados. Como internamente o protocolo PPP usa padrões específicos (por exemplo o campo 0X7E de *Flag*, que indica o começo e a extremidade de uma trama), é necessário usar *byte stuffing* através de um *byte* especial de controlo no qual a *Flag* 7E é substituída por 7D5E.

No procedimento de *scrambling/descrambling*, o *scrambler* descrito na RFC2615 usa um processamento de dados com o polinómio $1+x^{43}$. O FCS do HDLC é calculado com o *scrambler* a operar continuamente através dos *bytes* do *payload* contornando os *bytes* do trajecto de *overhead* do SONET/SDH e de toda a informação fixa. O *scrambling* descrito acima é executado durante a inserção do *payload* do SONET/SDH.

Para que a trama SONET/SDH reconheça qual será o conteúdo do *payload*, é usado o identificador de trajecto do sinal C2. A norma do SONET/SDH define o byte C2 do *Path Overhead* como o *path signal label*. O objectivo deste byte é de comunicar qual o tipo de *payload* encapsulado pelo FOH do SONET/SDH.

Value signal label C2	**Scrambler state**
0x16	PPP com $1+x^{43}$ *scrambling*
0xCF	PPP com *scrambler* desligado

Tabela 2.6 – Identificadores para *scrambling* do *Path Signal label* C2

Os pacotes PPP são introduzidos no *payload* do SONET/SDH sequencialmente linha a linha. Adaptam-se no envelope do *payload* como filas de octetos alinhadas na trama SDH.

Como as tramas podem ser de comprimento variável, é permitido às mesmas cruzar os limites do SONET/SDH. As tramas de HDLC são mapeadas dinamicamente dentro da trama de SONET/SDH. Quando não há qualquer tráfego, a trama de SONET/SDH está preenchida com as tramas de *idle* 0x7E, como ilustrado na figura 2.19.

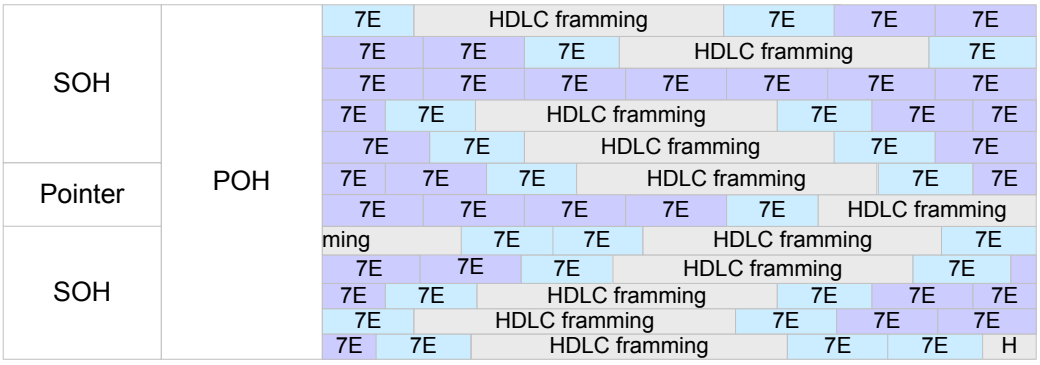


Figura 2.19 – Mapeamento de pacotes HDLC/PPP/IP dentro do payload SONET/SDH

2.3.2 LAPS - Link Access Procedure SDH

O LAPS surge principalmente para obviar a alguns dos problemas inerentes ao “PPP over SONET/SDH” (IETF RFC 2615). Algumas dificuldades em usar a RFC 2615 resumem-se aos seguintes pontos: o pesado *overhead* no envio de pacotes dado que se usa o LCP, o IPCP e números mágicos (no caso do LCP são necessários 10 pacotes de configuração, 16 eventos e 12 acções); demasiado tempo (3s) para estabelecer uma ligação PPP devido às diferenças entre as taxas do VC-12 (2Mb/s) e OC-192 (10Gb/s); existência do campo e função de *padding*; o elevado custo do HW e SW quando se usa ambos IP e PPP; não suporta contentores virtuais de baixa ordem o que seria útil para o SONET/SDH existente. Alguns destes pontos dão origem a um desempenho pouco eficiente e outros a um custo alto.

O grande objectivo do LAPS é o de remover os protocolos do PPP (o LCP e, o NCP no caso do POS).

O LAPS (ITU-T Recommendation X.85/Y.1321) é assim uma especificação HDLC para adaptar o IP directamente sobre SDH. É introduzido um campo designado SAPI para encapsular IPv4, IPv6, PPP e outros protocolos de camada superior. O LAPS é inteiramente compatível com a RFC 1662 quando o campo de endereço é ajustado a "11111111" e a ligação de dados é ajustada para operar como definido pela RFC2615. Isto significa que o LAPS pode operar da mesma forma que o “PPP over SONET/SDH” tende em consideração as diferenças existentes da trama PPP.

Actualmente o IPv4 é amplamente transportado sobre a maior parte dos sistemas ou canais de telecomunicações para suportar protocolos IP e para fornecer aplicações relacionadas com IP. Um dos canais usado frequentemente é o SDH. O modelo de IP directamente sobre o SDH é particularmente adequado para o IPv4 existente. O SDH e a rede óptica de transporte WDM associada, são considerados a fundação para a camada física de banda-larga de IP e de B-ISDN.

O IP sobre SDH usando o LAPS implementa o modelo simples do protocolo HDLC para IP sobre SDH (ITU-T X.200). Especifica as múltiplas ligações lógicas definidas pelo SAPI para encapsular protocolos de pacotes, baseados em IPv4, IPv6, PPP e outros protocolos de pacotes superiores. Define as várias interfaces físicas e primitivas a serem usadas numa rede "IP sobre SDH usando LAPS".

2.3.2.1 Estrutura do protocolo

O IP sobre SDH usando o LAPS é um tipo de arquitectura de comunicação de dados de combinação do protocolo Internet com a rede SDH. As camadas Física, *Data Link* e *Network* ou outros protocolos são especificados como SDH, LAPS, e IPv4/IPv6, PPP, etc., respectivamente, como a *layer/protocol stack* para IP sobre STM-N e sobre sSTM-N (figura 2.20).

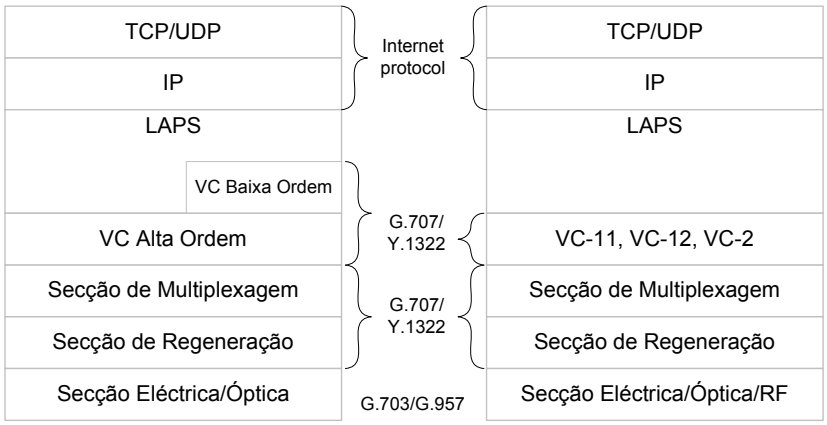


Figura 2.20 – *Layer/Protocol Stack* para IP sobre STM-N e sSTM-N usando LAPS

Na figura 2.20 está ilustrado o mapeamento de IP desde a camada *Network* sobre SONET/SDH, na camada Física, que é feito usando o LAPS na camada de *Data Link*, dentro de contentores virtuais (VC) segundo especificação da norma ITU-T G.707/Y.1322.

Na figura 2.21 está ilustrado um exemplo de uma rede IP usando a estrutura de uma rede de transporte que implementa o protocolo IP sobre SDH usando LAPS. Os pacotes IP são entregues desde a camada de Network até à camada Física, usando o LAPS na camada de *Data Link* para mapeamento em SDH.

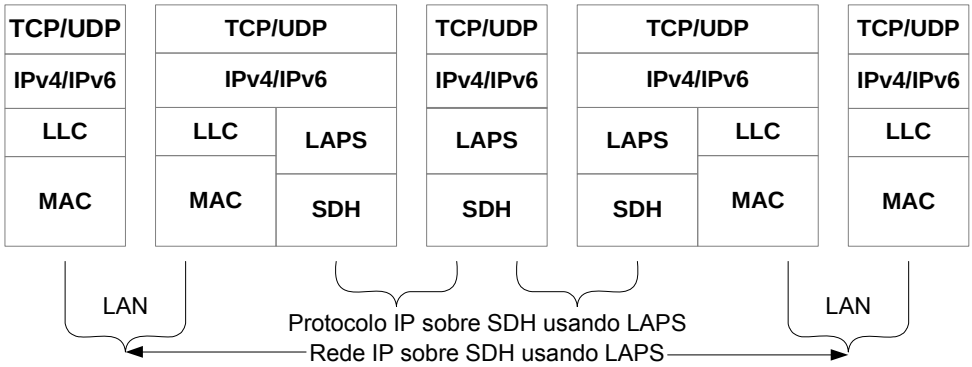


Figura 2.21 – Configuração do protocolo IP sobre SDH usando LAPS

Esta recomendação trata o transporte do SDH como ligações ponto-a-ponto síncronas, orientadas por octetos (as tramas do SDH, como já descrito, são uma estrutura síncrona, multiplexada e orientada que especifica uma série de taxas de ritmo padrão, formatos e métodos de mapeamento). Os serviços de comunicação entre as camadas de *Data Link* e Física são efectuados por meio de primitivas de acordo com o ITU-T X.211. O protocolo *Data Link* é o LAPS, que providencia transferências ponto-a-ponto sobre contentores virtuais e interfaces de taxas de SDH. O UITS suportado é um serviço de modo *connection-less*. As comunicações entre as camadas *Data Link* e *Network* ou os protocolos superiores associados são providenciados por meio de primitivas de acordo com o ITU-T X.212.

2.3.2.2 Compatibilidade com a RFC 2615

O PPP é usado para ser encapsulado através do SAPI para ser compatível com a RFC 2625. Neste caso ambos os FCS-32 e FCS-16 podem ser ajustados por aprovisionamento e não é negociado. Deve implementar-se o FCS-32 para ser usado com todas as taxas do SDH apesar de ser possível usar o FCS-16 para STM-1c/VC-4. Relativamente à etiqueta de sinal de caminho (C2) do SDH, esta é *scrambled* usando (x^{43}+1) e alterada de 0X18 para 0X16. Adicionalmente o LAPS providencia a etiqueta de sinal com valor 0XCF, para indicar o PPP sem *scrambling*. O Data Link terá a mesma operacionalidade da RFC 2615 em que o campo de endereço é igual a 0XFF, o campo de *padding* está a seguir ao campo de informação e as funções de LCP e NCP são também incluídas.

2.3.3 Principais diferenças entre LAPS e PPP/HDLC

O LAPS e o PPP/HDLC, apesar de serem bastante similares, têm diferenças importantes. Observando a figura 2.22, verifica-se que as principais diferenças estão na inserção da trama PPP e homologamente os campos SAPI e respectivo PDU que corresponde ao campo de informação (pacote IP) na trama LAPS.

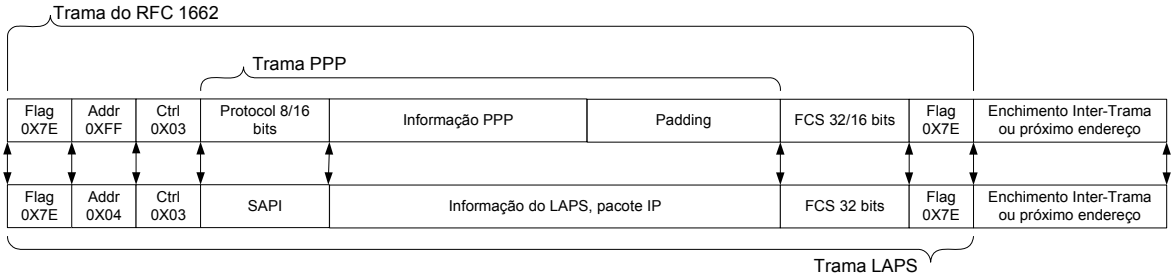


Figura 2.22 – Formatos das tramas LAPS (ITU-T X.85/Y.1321) e PPP/HDLC (IETF RFC 1662)

Em face disto pode-se discutir as diferenças entre ambos.

No caso de encapsulamento multi-protocolo o LAPS usa um conjunto de SAPIs e o PPP/HDLC usa o PPP.

No campo de endereço da trama HDLC que tem o comprimento de um octeto, no PPP o endereço de *all-station* é sempre 0XFF e endereços de estação individual não são atribuídos; no LAPS o endereço de *all-station* é 0XFF que será usado para ser compatível com a RFC 2615 e os endereços de estação são configurados a 0X04.

O PPP/HDLC usa um campo de protocolo 16/8 bits enquanto que o LAPS usa o SAPI de dois octetos.

O LAPS não usa campo de *Padding* e, na transmissão, no PPP/HDLC o campo de informação pode ser preenchido com um número arbitrário de octetos até ao máximo MRU, que é por defeito 1500 octetos. No entanto, implementações reais do PPP podem usar outros valores para o MRU.

O LAPS não usa qualquer número mágico e facilidades de configuração associadas mas, no caso do PPP/HDLC, esta opção de configuração fornece um método para detectar as ligações *looped-back* e outras anomalias da camada de ligação.

O LAPS não usa, ao contrário do PPP/HDLC, qualquer controlo de inicialização e ligação. O tempo de reinicialização é usado no PPP, mas não é especificado na RFC 1619, o seu valor depende do tempo *round-trip* da ligação que é mudado juntamente com as diferentes taxas do SDH (este tempo é necessário para uma comunicação *peer-to-peer* para o PPP).

O LAPS usa um campo FCS de 32 bits, enquanto que no PPP/HDLC o comprimento deste campo varia de 16 a 32 bits, e é ajustado por aprovisionamento.

O LAPS usa os valores 0X7E e 0X7D como octetos de controlo de escape; no caso do PPP/HDLC as implementações de envio devem fazer o escape dos octetos de controlo de escape.

Relativamente a tramas inválidas, no LAPS as tramas que são demasiado curtas (menos de 6 octetos ao usar o FCS 32-bit), ou que terminem sem um campo de controlo, ou nas quais o *octet-framing* é violado, são rejeitadas discretamente, e não são contadas como um erro do FCS. No caso do PPP/HDLC as tramas que são demasiado curtas (menos de 4 octetos ao usar o FCS 16-bit), ou que terminem com um octeto de escape de controlo seguido imediatamente por uma sequência *Flag* de fecho, ou nas quais o *octet-framing* é violado, são rejeitadas discretamente, e não são contadas como um erro do FCS.

No LAPS é somente usado *Scrambling*. O *Path Signal Label* do trajecto de ordem elevada (C2) é ajustado a 24 (0X18) ao usar $x^{43}+1$ para *scrambling*. O *Path Signal Label* de mais baixa ordem (V5) é ajustado ao código (101 binário) ao usar $x^{43}+1$ para *scrambling*. No PPP/HDLC ambos *scrambling* e *non-scrambling* são usados. O *Path Signal Label* (C2) é ajustado a 22 (0X16) ao usar $x^{43}+1$ para *scrambling*. Se o *scrambling* for configurado para estar desligado, então é usado o valor 207 (0XCF).

2.4 GFP - Generic Framing Procedure

Vários protocolos importantes para LANs de alta velocidade usam um código de bloco da camada 1 para comunicar dados e informação de controlo. O código de bloco mais comum é o código de linha 8B/10B usado para Gigabit Ethernet, ESCON, SBCON, *Fiber Channel*, FICON e Infiniband. Estes códigos de linha tornaram-se importantes com o aumento de popularidade das SANs. O código de linha 8B/10B faz o mapeamento de 256 valores nos 1024 valores possíveis para o espaço de codificação de 10-bit. De modo a transportar eficientemente protocolos que usem o código de linha 8B/10B através da rede pública de transporte, como por exemplo o SONET/SDH ou a OTN, é necessário transportar ambos os dados e a informação de controlo 8B/10B. No entanto usar a codificação 8B/10B faz aumentar em 25% a largura de banda usada, o que é indesejável para a rede de transporte. Dentre as possíveis alternativas de protocolos para transportar estes sinais da LAN através das redes SONET/SDH e OTN, o GFP [16] providencia algumas vantagens sobre o ATM e o POS. O ATM necessita de maior complexidade do que o GFP, nos processos de adaptação. O POS necessita terminar o sinal cliente[1] da camada 2 e re-mapear o sinal dentro do PPP sobre HDLC. Seguidamente este mapeamento necessita fazer o *escape* dos caracteres de controlo do HDLC existentes na trama de dados, resultando numa expansão não determinística da largura de banda usada.

Neste contexto a recomendação G.7041/Y.1303 do ITU-T define um procedimento genérico de encapsulamento (GFP) para delinear cargas alinhadas em octetos e de comprimento variável, desde clientes de camadas superiores, para mapeamento subsequente em trajectos síncronos por octetos, como os definidos nos G.707 e G.709 do ITU-T.

2.4.1 O protocolo GFP

O GFP fornece um mecanismo genérico para adaptar tráfego de dados de clientes de camadas superiores sobre uma rede de transporte. Os sinais do cliente podem ser orientados por pacotes (IP/PPP ou Ethernet) referido como *Frame-Mapped* GFP (GFP-F), ou orientados por blocos com taxa de ritmo constante (*Fibre Channel* ou ESCON/SBCON) referido como *Transparent* GFP (GFP-T).

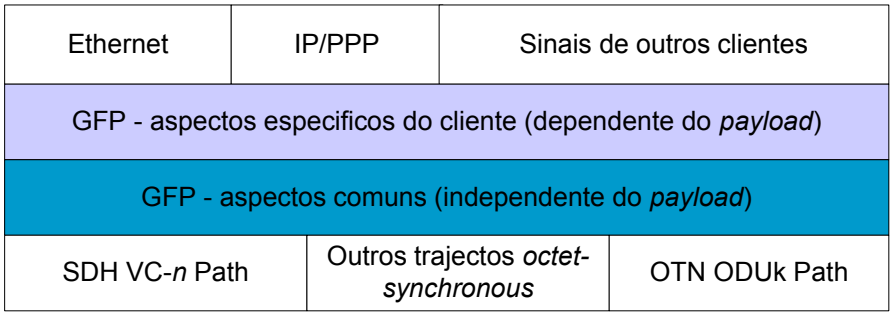


Figura 2.23 – Relações GFP para sinais de clientes e trajectos de transporte

O GFP usa uma variação do mecanismo de delineação de trama baseado no HEC definido do ATM (ITU-T Rec. I.432.1). São definidos dois tipos de tramas GFP: tramas GFP de cliente e tramas GFP de controlo. O GFP também suporta um mecanismo flexível de extensão de cabeçalho (*payload*) para facilitar a adaptação do GFP para usar com diversos mecanismos de transporte.

[1] O termo “sinal cliente” ou mais genericamente “cliente” é entendido aqui nesta secção sobre o GFP, como sendo um protocolo de uma camada superior que é transportado por um outro protocolo da camada imediatamente abaixo daquela.

No modo de adaptação GFP-F a função de adaptação Cliente/GFP pode operar na camada de *Data Link* (ou mais alta) do sinal do cliente. Neste caso é necessária a visibilidade do PDU do cliente. Esta visibilidade é obtida quando os PDUs do cliente são recebidos de ambas as redes da camada de dados (IP *router fabric* ou *Ethernet switch fabric* (C/C' na figura 2.24)), ou por exemplo uma função de *bridge*, *switch* ou *router* num TNE. Noutro caso os PDUs clientes são recebidos por exemplo via uma interface Ethernet (A/A' na figura 2.24).

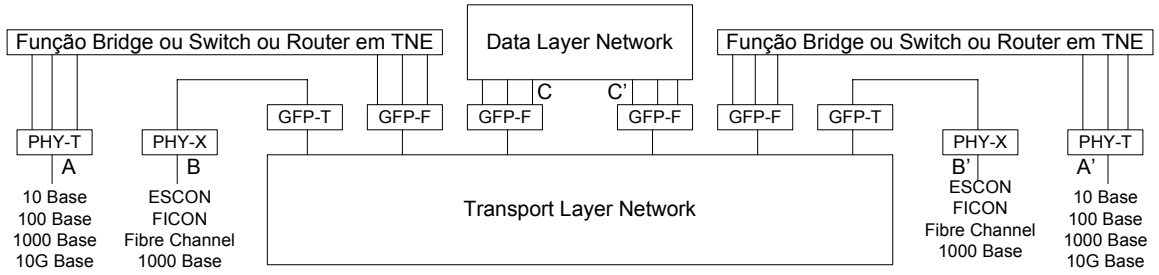


Figura 2.24 – Modelo funcional do GFP (Cliente Simples)

Para o modo de adaptação transparente (GFP-T), a função de adaptação Cliente/GFP opera sobre o fluxo de caracteres codificados da trama, em vez dos PDUs clientes de entrada. Portanto, é necessário o processamento do espaço codificado de palavra de entrada, (B/B' na figura 2.24).

Tipicamente, as interligações podem ser feitas entre os portos A e A', B e B', C e C', A e C', e C e A'.

2.4.2 Estrutura básica de tramas GFP de clientes

As tramas GFP são alinhadas por octetos e consistem num GFP *Core Header* e, excepto para tramas GFP *Idle*, uma área GFP *Payload*.

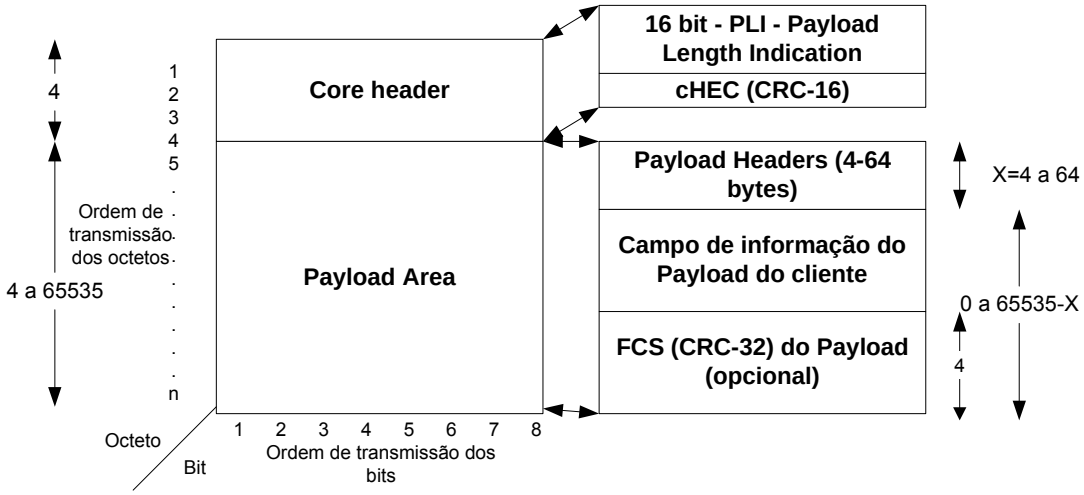


Figura 2.25 – Formato da trama GFP para tramas de cliente

Os quatro octetos do Core Header do GFP consistem num campo de 16 bits PLI e num campo de 16 bits cHEC. Este cabeçalho permite o delineamento da trama GFP, independentemente do conteúdo dos PDUs de camadas superiores.

Os dois octetos do campo PLI contêm um número binário que representa o número de octetos na GFP *Payload Area*. O valor mínimo absoluto do campo PLI numa trama GFP cliente é de quatro octetos. Os valores de 0-3 são reservados para uso de controlo da trama GFP.

Os dois octetos do campo cHEC contêm um código de controlo de erro CRC-16 que protege a integridade dos conteúdos do *Core Header* ao activarem ambas a correcção de erro *single-bit* e a detecção de erro *multi-bit*. A sequência cHEC (Core HEC) é calculada sobre os octetos do *Core Header*.

A área de *Payload* do GFP, que consiste em todos os octetos na trama de GFP após o GFP *Core Header*, é usada para fornecer informação específica do protocolo da camada mais elevada. Esta área de comprimento variável pode incluir 4 a 65535 octetos. Consiste em dois componentes: um cabeçalho do *payload* e um campo de informação do *payload*. É também suportado um campo opcional de FCS do *payload* (pFCS).

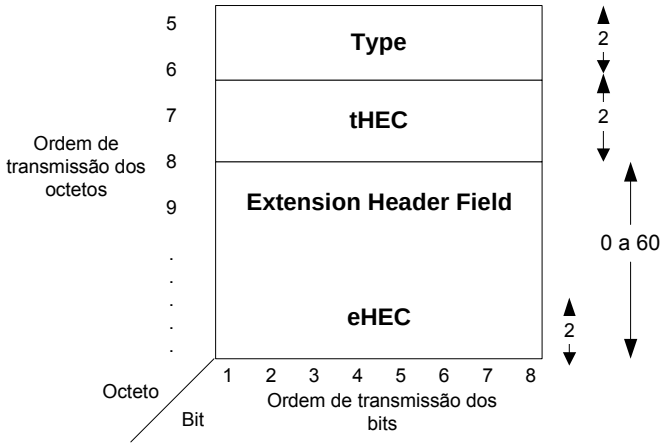


Figura 2.26 – Formato do cabeçalho do *payload* do GFP

Os tamanhos práticos de MTU para a área de payload do GFP são específicos da aplicação. Uma implementação deve suportar a transmissão e a recepção de tramas GFP com áreas de *payload* de pelo menos 1600 octetos. No entanto pode haver implementações GFP cujas especificações podem usar outros valores de MTU.

O cabeçalho do *payload* (*Payload Header*) é uma área de comprimento variável, 4 a 64 octetos, usado para suportar procedimentos de gestão da ligação de dados, específicos ao sinal do cliente da camada superior. Esta área contém os campos *Type* e tHEC (*Type* HEC), e um número variável de campos adicionais. Este grupo de campos adicionais do cabeçalho do *payload* é referido como *Extension Header*. A presença do *Extension Header*, o seu formato e a presença opcional do FCS do *payload,* são especificados pelo campo de *Type*. O tHEC protege a integridade do campo de *Type*.

Qualquer implementação terá de suportar a recepção de uma trama de GFP com um *Payload Header* de qualquer comprimento entre 4 a 64 octetos.

O campo de dois octetos GFP *Type* é um campo imperativo do *Payload Header* que indica o conteúdo e o formato do GFP. O campo de *Type* faz a distinção entre tipos de tramas de GFP e entre diferentes serviços num ambiente multi-serviço. Este campo consiste num PTI (Payload Type Identifier), num PFI (Payload FCS Indicator), num EXI (Extension Header Identifier) e num UPI (User Payload Identifier).

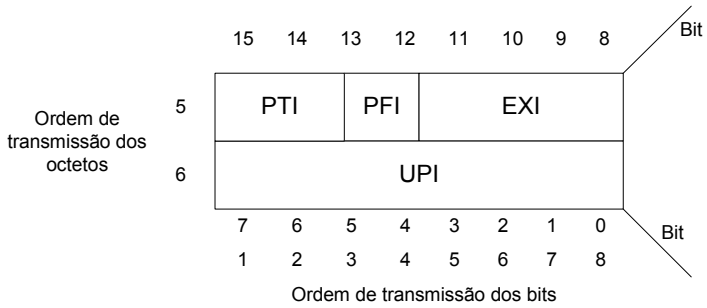


Figura 2.27 – Formato do campo GFP Type

O PTI é um sub campo de 3 bits do campo de *Type* que identifica o tipo de trama de cliente do GFP. São definidos dois tipos de trama de cliente, tramas de dados do utilizador (PTI = 000) e tramas de gestão do cliente (PTI = 100).

O PFI é um sub campo de um bit do campo de Type que indica a presença (PFI = 1) ou ausência (PFI = 0) do campo do FCS do *payload*.

O EXI é um sub campo de 4 bits do campo de *Type* que identifica o tipo de extensões do cabeçalho do GFP. São definidos três tipos de extensões de cabeçalhos: um cabeçalho de extensão nula, um cabeçalho de extensão linear e um cabeçalho de extensão do anel.

O UPI é um campo de 8 bits que identifica o tipo de *payload* fornecido no campo de informação do *payload* do GFP. A interpretação do campo do UPI é relativa ao tipo de trama do cliente do GFP como indicado pelo sub campo de PTI.

O campo tHEC é um campo de dois octetos de controlo de erro do tipo de cabeçalho, contém um código de controlo de erro CRC-16 que protege a integridade dos conteúdos do tipo de campo, permitindo a correcção de erro de *single-bit* e a detecção de erro de *multi-bit*.

O cabeçalho de extensão do *payload* é um campo de 0 a 60 octetos (incluindo o eHEC) que suporta cabeçalhos de ligação de dados específicos a determinada tecnologia tais como identificadores virtuais de ligação, endereços de origem/destino, números dos portos, classes de serviço, controlo de erro do cabeçalho da extensão, etc.. O tipo do cabeçalho de extensão é indicado pelo índice dos bits de EXI no campo do tipo de cabeçalho do *payload*.

São definidas três variantes do cabeçalho de extensão para suportar dados específicos do cliente sobre um anel lógico ou sobre configurações lógicas (lineares) ponto a ponto.

O cabeçalho de extensão nula (Null Extension Header) aplica-se a uma configuração ponto a ponto lógica. É destinada a cenários onde o trajecto de transporte é dedicado a um sinal do cliente.

O cabeçalho do *payload* para uma trama (ponto a ponto) linear com um cabeçalho de extensão é destinado a cenários onde há diversas ligações independentes que requerem a agregação num único trajecto de transporte.

O CID (Channel ID) é um número binário de 8 bits, usado para indicar um de 256 canais de comunicações num ponto de terminação do GFP. O campo de 8 bits de *Spare* é reservado para uso futuro. O campo de dois octetos de controlo de erro do cabeçalho da extensão (eHEC) contém um código de controlo de erro CRC-16 que protege a integridade dos conteúdos dos cabeçalhos da extensão, permitindo a correcção de erro *single-bit* (opcional) e a detecção de erro *multi-bit*.

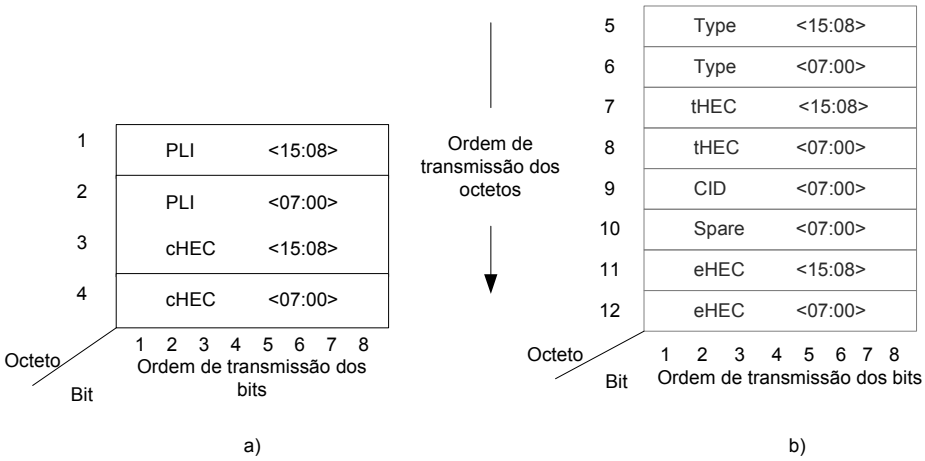


Figura 2.28 – Payload Header para uma trama GFP com um Null Extension Header (a) e para uma trama linear (ponto a ponto) incluindo o Extension Header (b)

O campo de informação do *payload* contém o PDU mapeado para a trama GFP ou, no exemplo de GFP transparente, um grupo de caracteres do sinal do cliente. Este campo de comprimento variável pode incluir de 0 a 65535-X octetos, onde X é o tamanho do cabeçalho do *payload*. Este campo pode incluir um campo opcional de FCS do *payload*. O sinal do cliente PDU é transferido sempre no campo de informação do *payload* do GFP, como um pacote alinhado por octeto.

O FCS do *payload* (pFCS) do GFP é uma sequência opcional, de quatro octetos, de verificação da trama. Contém uma sequência CRC-32 que protege os índices do campo de informação do *payload* do GFP. O valor de 1 no bit de PFI dentro do campo de *Type* identifica a presença do campo do FCS do *payload*.

2.4.3 Tramas GFP de cliente

No GFP são definidos dois tipos de tramas de cliente: tramas de dados que são usadas para transportar dados do sinal cliente; e tramas de gestão que são usadas para transportar a informação associada à gestão do sinal do cliente ou da ligação do GFP.

Os dados do cliente são transportados sobre GFP usando tramas de dados de cliente. As tramas de dados de cliente são tramas GFP que consistem num *Core Header* e numa *Payload Area*. O campo de *Type* das tramas de dados do cliente tem a estrutura ilustrada na figura 2.27, com o PTI = 000 e os PFI, EXI e UPI específicos do *payload.*

O PFI será ajustado como for necessário dependendo se o FCS está activo ou não. O EXI será ajustado consistentemente às exigências de *multiplexing* e aos requisitos da topologia para a ligação do GFP. O UPI será ajustado de acordo com o tipo de sinal do cliente transportado.

As tramas GFP de gestão de cliente fornecem um mecanismo genérico para o processo específico da adaptação da origem do cliente de GFP para, opcionalmente, emitir tramas de gestão de cliente ao processo de recepção específico da adaptação do cliente de GFP.

As tramas de gestão de cliente são tramas GFP que consistem num *Core Header* e numa *Payload Area*. O campo de *Type* das tramas de gestão do cliente tem também a estrutura ilustrada na figura 2.27, com o PTI = 100 e os PFI, EXI e UPI específicos do *payload.*

Para uso como uma trama GFP de gestão de cliente o PFI será ajustado como for necessário, dependendo se o FCS está activo ou não. O indicador do EXI será ajustado como necessário, dependendo se o cabeçalho da extensão é empregue ou não. O UPI define o uso do *payload* da trama GFP de gestão de cliente. Desta maneira a trama GFP de gestão de cliente pode ser usada para múltiplas finalidades.

2.4.4 Tramas de GFP Control

As tramas GFP de controlo são usadas na gestão da ligação do GFP. A única trama de controlo especificada é a trama *Idle* que é uma trama especial de controlo de quatro octetos que consiste somente num *Core Header* GFP com os campos de PLI e de cHCE ajustados a 0, e nenhuma *Payload Area*. A trama *Idle* é usada como uma trama de enchimento para facilitar o processo de adaptação da origem do GFP para a adaptação da trama do octeto de GFP a qualquer meio de transporte onde o canal de meio de transporte tem uma capacidade mais elevada do que a requerida pelo sinal do cliente.

2.4.5 Mapeamento de Ethernet e IP/PPP em tramas GFP

O formato das tramas do MAC *Ethernet* é definido no IEEE 802.3, (ver próxima secção). Há uma relação de paridade no mapeamento entre um PDU de uma camada superior e o PDU do GFP. Especificamente, os limites do PDU do GFP são alinhados com os limites dos PDUs da trama da camada mais elevada.

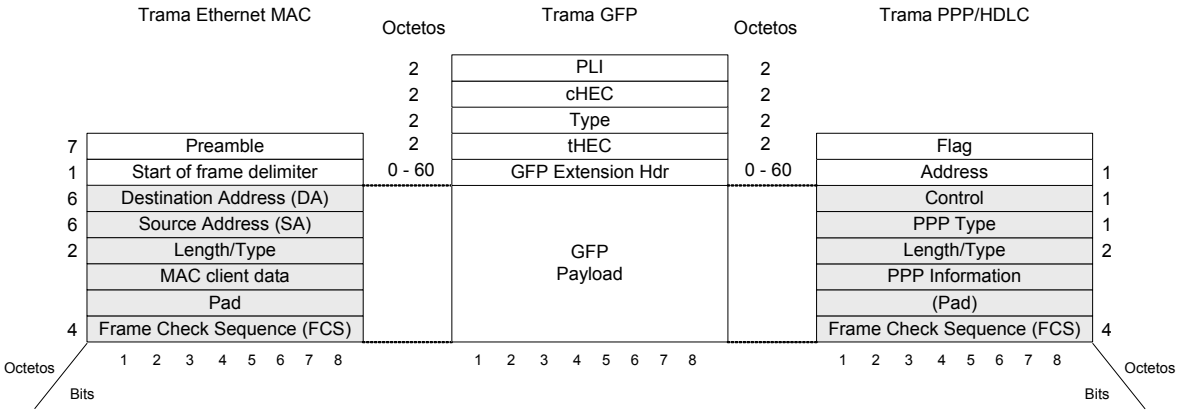


Figura 2.29 – Relações das tramas Ethernet, PPP/HDLC e GFP

Os octetos do MAC *Ethernet* desde o *Destination Address* até ao *Frame Check Sequence*, inclusive, são colocados no campo do GFP *Payload Information*. O alinhamento dos octetos é mantido e a identificação dos bits dentro dos octetos é mantida. Especificamente, numa base de octeto-por-octeto, os bits 0 e 7 na cláusula 3 do IEEE 802.3 correspondem aos bits 8 e 1, respectivamente, na recomendação do GFP.

Os *payloads* de IP/PPP são primeiramente encapsulados numa trama HDLC-*like*. O formato de uma trama PPP é definido na RFC 1661. O formato da trama HDLC-*like* é definido na RFC 1662. Ao contrário da RFC 1662, não é executado qualquer procedimento de enchimento de octeto em *flags* ou em caracteres de escape de controlo. Há um mapeamento directo entre um PDU PPP/HDLC de uma camada elevada e um PDU GFP. Especificamente, os limites do PDU do GFP são alinhados com os limites das tramas da camada mais elevada dos PDUs PPP/HDLC.

Todos os octetos da trama de PPP/HDLC, incluindo todo o *padding* opcional do campo de informação do PPP, são colocados no campo de informação do *payload* de uma trama GFP. O alinhamento do octeto é mantido e a identificação dos bits dentro dos octetos é também mantida.

2.5 Ethernet

A Ethernet é uma norma protocolar para uso nas LANs que emprega como método de acesso o CSMA/CD para um método de acesso nos dois sentidos. Esta norma foi desenvolvida para suportar vários tipos de media (ex. IP) e técnicas para sinais com taxas desde 1 Mb/s a 1 Gb/s. Adicionalmente, tem especificado um método para o incremento linear das taxas de dados de um

sistema, agregando múltiplas ligações físicas com a mesma taxa de dados, para dentro de uma ligação lógica. Esta norma providencia dois métodos distintos de comunicação, um em *half-duplex* outro em *full-duplex*. No modo *half-duplex*, o meio pelo qual duas ou mais estações partilham um meio comum de transmissão é o método CSMA/CD. A operação em *full duplex* permite a comunicação simultânea entre um par de estações usando um canal ponto-a-ponto dedicado e, neste caso, é desnecessário o uso do método CSMA/CD.

A figura 2.30 ilustra os serviços fornecidos pela sub-camada do MAC Ethernet e pela sub-camada opcional de MAC *Control,* ao cliente do MAC. Os clientes do MAC podem incluir a sub-camada de LLC, a entidade de *Bridge Relay*, ou outros utilizadores de serviços do MAC do ISO/IEC LAN International Standard.

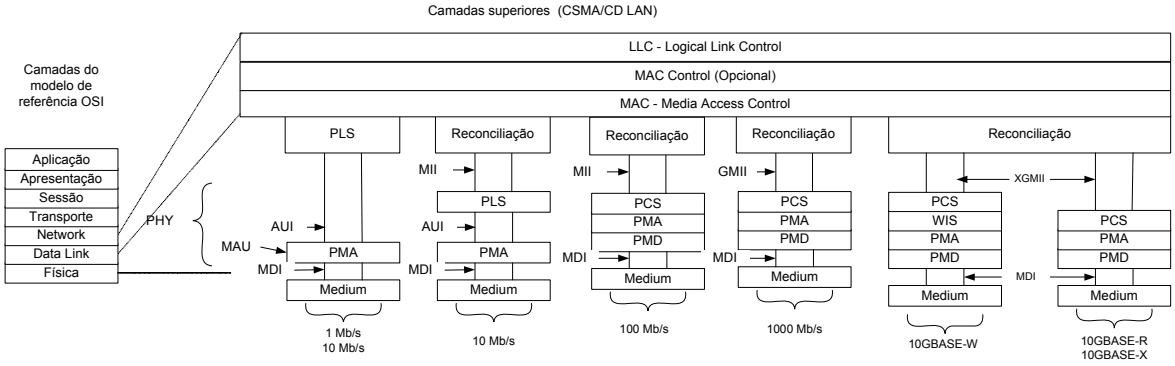


Figura 2.30 – Relações dos serviços no modelo da LAN

2.5.1 Controlo MAC

Os serviços fornecidos pela sub-camada do MAC permitem que a entidade local do cliente do MAC, troque unidades de dados do LLC com entidades pares da sub-camada do LLC.

A sub-camada opcional de controlo do MAC fornece um serviço adicional para o controlo da operação do MAC. Isto pode ser usado para fornecer o controlo de fluxo entre outras entidades clientes pares do cliente do MAC.

Há quatro primitivas entre o MAC Ethernet e o cliente do MAC (LLC) e, bem como, entre o MAC Ethernet e a camada física (PHY), como ilustrado na figura 2.31.

A função da primitiva MA_DATA.request é de definir a transferência de dados de uma entidade do cliente do MAC a uma única entidade par*,* ou a múltiplas entidades pares no caso de endereços de grupo.

- MA_DATA.request (destination_address, source_address, m_sdu, service_class)

O DA (destination_address) pode especificar uma estação individual ou o endereço duma entidade MAC de grupo. Deve conter a informação suficiente para criar o campo de DA que é adicionado à trama pela entidade local da sub-camada do MAC. O source_address, se presente, deve especificar um endereço de MAC individual. Se o parâmetro de source_address for omitido, a entidade local da sub-camada do MAC introduzirá um valor associado com essa entidade. O m_sdu especifica a unidade de dados de serviço do MAC a ser transmitida pela entidade da sub-camada do MAC. Há informação suficiente associada ao m_sdu para que a entidade da sub-camada do MAC determine o comprimento da unidade de dados. O service_class indica a qualidade de serviço requisitada pelo cliente do MAC.

Esta primitiva é gerada pela entidade de cliente do MAC sempre que os dados são transferidos a uma entidade par ou a várias entidades pares. Isto pode suceder em resposta a um pedido desde camadas protocolares mais elevadas, de dados gerados internamente ao cliente do MAC.

A recepção desta primitiva fará com que a entidade do MAC introduza todos os campos específicos do MAC, incluindo DA, SA e qualquer campo que seja especifico ao método de acesso

ao meio, e passe a trama correctamente formada às camadas mais baixas do protocolo para transferência à entidade ou às entidades pares da sub-camada do MAC.

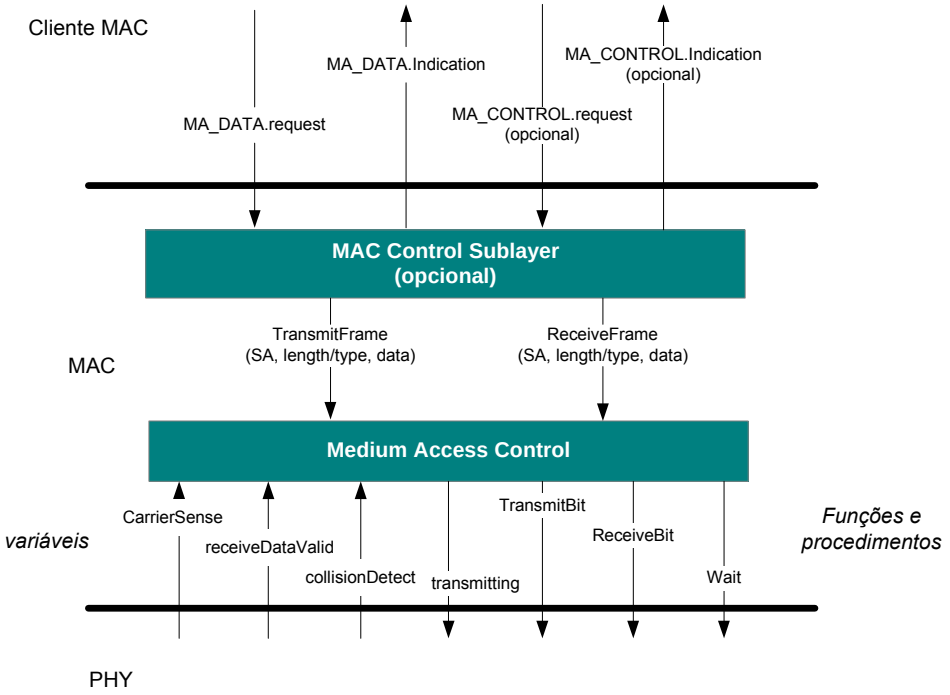


Figura 2.31 – Modelo usado para a especificação da relação entre as primitivas dos serviços

A função da primitiva MA_DATA.indication é de definir a transferência dos dados da entidade da sub-camada do MAC (através da sub-camada opcional de controlo do MAC, se implementada) à entidade ou às entidades de cliente do MAC no caso de endereços de grupo.

- MA_DATA.indication (destination_address, source_address, m_sdu, reception_status)

O destination_address pode ser uma estação individual ou um endereço de grupo como especificado pelo campo DA da trama recebida. O source_address é um endereço individual como especificado pelo campo de SA da trama recebida. O m_sdu especifica a unidade de dados do serviço do MAC tal como recebida pela entidade local do MAC. O reception_status é usado para passar a informação de estado à entidade de cliente do MAC.

O MA_DATA.indication é passado da entidade da sub-camada do MAC (através da sub-camada opcional de controlo do MAC, se implementada) à entidade ou às entidades de cliente do MAC, para indicar a chegada de uma trama à entidade local da sub-camada do MAC e que é destinada ao cliente do MAC. Tais tramas são relatadas somente se estiverem validamente formadas, recebidas sem erro e o seu endereço de destino designa a entidade local do MAC. As tramas destinadas à sub-camada opcional de controlo do MAC não são passadas ao cliente do MAC se a sub-camada de controlo do MAC for implementada.

Se a entidade local da sub-camada do MAC for designada pelo parâmetro de destination_address de um MA_DATA.request, a primitiva de indicação será também invocada pela entidade do MAC à entidade do cliente do MAC. Esta característica da sub-camada do MAC é devida à funcionalidade da mesma ou das características das camadas mais baixas (por exemplo, todas as tramas transmitidas ao endereço de difusão, invocam MA_DATA.indication em todas as estações na rede incluindo a estação que gerou o pedido).

A primitiva MA_CONTROL.request define a transferência de pedidos de controlo do cliente do MAC à sub-camada de controlo do MAC. A implementação da primitiva MA_CONTROL.request é imperativa se a sub-camada opcional de controlo do MAC for implementada. Portanto a função

desta primitiva é de definir a transferência de comandos de controlo de uma entidade do cliente do MAC à entidade local da sub-camada de controlo do MAC.

- MA_CONTROL.request (destination_address, opcode, request_operand_list)

O destination_address pode especificar o endereço de uma estação individual ou um endereço do grupo da entidade do MAC de destino. Deve conter a informação suficiente para criar o campo de DA que será adicionado à trama pela entidade local da sub-camada do MAC. O *opcode* especifica a operação de controlo pedida pela entidade do cliente do MAC. O *request_operand_list* é um conjunto de parâmetros específicos do *opcode*.

Esta primitiva é gerada por um cliente do MAC sempre que deseja usar os serviços da entidade opcional da sub-camada de controlo do MAC. O efeito de recepção desta primitiva pela sub-camada de controlo do MAC é específico do *opcode*.

A função da primitiva MA_CONTROL.indication é de definir a transferência de indicações de estado de controlo da sub-camada de controlo do MAC ao cliente do MAC. A execução da primitiva de MA_CONTROL.indication é imperativa se a sub-camada opcional de controlo do MAC for implementada.

- MA_CONTROL.indication (opcode, indication_operand_list)

A MA_CONTROL.indication é gerada pela sub-camada de controlo do MAC sob as condições específicas a cada operação de controlo do MAC.

2.5.2 Estrutura da trama MAC Ethernet

São especificados dois formatos de trama: um formato básico de trama do MAC; e uma extensão do formato básico de trama do MAC que transporta prefixos QTag.

A figura 2.32 ilustra os nove campos de uma trama Ethernet: o *preamble*, o SFD (Start Frame Delimiter), os endereços de origem e de destino da trama, um campo de comprimento ou campo de tipo para indicar o comprimento ou o tipo de protocolo do campo seguinte que contém os dados do cliente do MAC, um campo que contenha o *padding* se requerido, o campo de verificação de sequência da trama que contém o valor da verificação de redundância cíclica para detectar erros numa trama recebida, e um campo de extensão se requerido (somente para uma operação *half-duplex* de 1000 Mb/s). Destes nove campos, todos são de tamanho fixo à excepção dos dados, do *pad*, e dos campos de extensão, que podem conter um número inteiro de octetos entre os valores mínimo e máximo que são determinados pela implementação específica do MAC CSMA/CD.

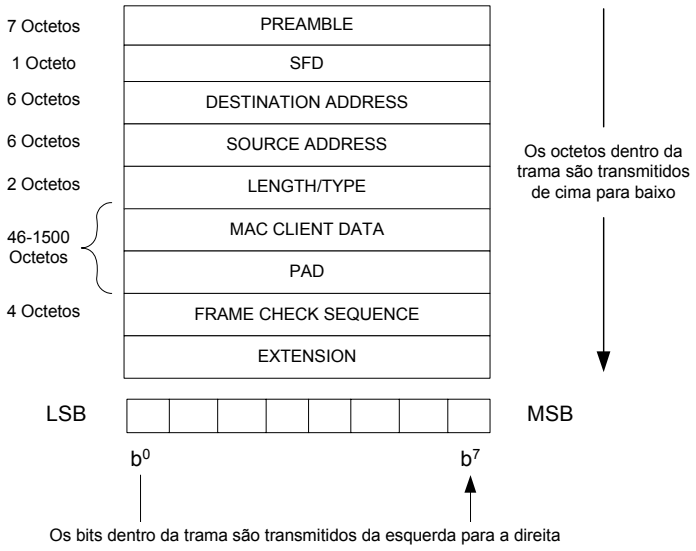


Figura 2.32 – Formato da trama MAC Ethernet

O *Preamble* é um campo de 7-octetos que é usado para permitir que os circuitos de sincronização de relógio alcancem um sincronismo de estado estacionário com o relógio da trama recebida. O campo de SFD contém a sequência 10101011. Este campo situa-se a seguir ao teste padrão do *preamble* e indica o começo de uma trama.

Cada trama do MAC contém dois campos de endereço: o endereço de destino e o endereço de origem. O campo de endereço de destino especifica o(s) endereço(s) de destino para o qual a trama é pretendida. O campo de endereço de origem identifica a estação da qual a trama foi iniciada.

O endereço da sub-camada do MAC pode ser um de dois tipos:

a) Endereço individual: endereço associado a uma estação particular na rede;

b) Endereço de grupo: endereço de multi-destino (*multicast*), associado a uma ou mais estações numa dada rede.

Por sua vez, há dois tipos de endereços *multicast*:

1) Endereço de *Multicast-Group*: endereço associado por convenção com um grupo de estações logicamente relacionadas;

2) Endereço de *Broadcast*: endereço *multicast* distinto, predefinido, que denota sempre o conjunto de todas as estações numa dada LAN.

O endereço de *Broadcast* é predefinido como sendo tudo a 1 no campo de endereço de destino. Este endereço é predefinido para que cada meio de comunicação consista em todas as estações ligadas a esse meio; é usado para transmitir a todas as estações activas nesse meio e todas reconhecem o endereço de *Broadcast*.

O *Destination Address* especifica a(s) estação(ões) para a(s) quais(al) a trama é pretendida. Pode ser um endereço de indivíduo ou de *multicast* (incluindo *broadcast*).

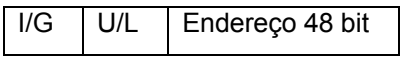
O *Source Address* especifica a estação que emite a trama.

a) Cada campo de endereço tem 48 bits de comprimento;

b) O primeiro bit (LSB) é usado no campo de endereço de destino como um bit de designação do tipo de endereço, para identificar o endereço de destino como sendo o endereço de um indivíduo ou de um grupo. Se este bit for 0, indica que o campo de endereço contém um endereço individual. Se este bit for 1, indica que o campo de endereço contém um endereço de grupo que identifica nenhuma, uma ou mais, ou todas as estações ligadas à LAN. No campo de endereço de origem, o primeiro bit é reservado e ajustado a 0;

c) O segundo bit será usado para distinguir entre endereços local ou globalmente administrados. Para endereços globalmente administrados, o bit é ajustado a 0. Se um endereço for localmente atribuído, este bit será ajustado a 1. De notar que para o endereço de transmissão, este bit é também 1;

d) Cada octeto de cada campo de endereço será transmitido primeiramente desde o bit menos significativo.



I/G = 0 endereço individual

I/G = 1 endereço de grupo

U/L = 0 endereço administrado globalmente

U/L = 1 endereço administrado localmente

Figura 2.33 – Formato do campo de endereço

O *Length/Type*, de dois octetos, tem um de dois significados, dependendo do seu valor numérico. Para avaliação numérica, o primeiro octeto é o octeto mais significativo deste campo.

a) Se o valor deste campo for menor do que ou igual ao valor de tamanho máximo válido da trama (maxValidFrame), então o campo de *Length/Type* indica o número de octetos dos dados do cliente do MAC contidos no campo de dados subsequente à trama (interpretação de comprimento);

b) Se o valor deste campo for maior do que ou igual ao valor decimal 1536 (igual a 0600 hexadecimal), então o campo de *Length/Type* indica a natureza do protocolo do cliente do MAC (interpretação de tipo). As interpretações de comprimento e de tipo deste campo são mutuamente exclusivas.

Quando usado como um campo de tipo é da responsabilidade do cliente do MAC assegurar que o cliente do MAC opera correctamente quando a sub-camada do MAC "adapta" (*pads*) os dados fornecidos.

Não obstante a interpretação do campo de *Length/Type*, se o comprimento do campo de dados for menor do que o mínimo requerido para a operação apropriada do protocolo, será adicionado um campo de PAD (uma sequência de octetos) na extremidade do campo de dados mas antes do campo de FCS. O campo de *Length/Type* é transmitido e recebido primeiramente com o octeto de ordem elevada.

Os campos de *Data* e PAD contêm uma sequência de n octetos. A transparência total dos dados é fornecida no sentido de que qualquer sequência arbitrária de valores do octeto pode aparecer no campo de dados até um número máximo especificado pela implementação que é usada da norma. É necessário um tamanho mínimo da trama para a operação correcta dos protocolos CSMA/CD. Se necessário, o campo de dados é estendido adicionando bits extra (isto é, um *pad*) em unidades de octetos, após o campo de dados mas antes de calcular e de adicionar o FCS. O tamanho do *pad*, se existir, é determinado pelo tamanho do campo de dados fornecido pelo cliente do MAC e pelos parâmetros mínimos do tamanho da trama e do tamanho do endereço da implementação específica. O tamanho máximo do campo de dados é determinado pelos parâmetros máximos do tamanho da trama e do tamanho do endereço da implementação específica.

No campo de FCS é usado um CRC pelos algoritmos de transmissão e recepção para gerar um valor de CRC para o campo do FCS. O campo de FCS contém um valor de 4 octetos (32 bit) de CRC. Este valor é calculado em função dos índices do endereço de origem, do endereço de destino, do comprimento, dos dados do LLC e do *pad* (isto é, todos os campos excepto o *preamble*, o SFD, o FCS, e a extensão).

O campo de Extension segue o campo de FCS, e é composto por uma sequência de bits de extensão, que são distintos dos bits de dados. Os índices do campo de extensão não são incluídos no cálculo do FCS.

2.5.3 Elementos da trama Ethernet com a opção de VLAN Tagging

Uma trama VLAN-*tagged* é simplesmente uma trama MAC Ethernet de dados básica, cujas extensões são as seguintes:

a) Um prefixo de 4 octetos (QTag) é introduzido entre a extremidade final do endereço de origem e do campo *Length/Type* do cliente do MAC. O prefixo de QTag compreende dois campos:

   1) um campo constante de 2 octetos Length/Type consistente com a interpretação de *Type* e igual ao valor do 802.1Q Tag Protocol Type;

   2) um campo 2 octetos que contém a informação de controlo do Tag;

b) Depois do prefixo QTag surge o campo Length/Type do cliente do MAC, os dados do cliente do MAC, o *pad* (se necessário), o FCS, e os campos de extensão (se necessário) da trama básica do MAC;

c) O comprimento da trama é estendido por 4 octetos pelo prefixo de QTag.

A opção VLAN *tagging* permite três capacidades importantes para os utilizadores e gestores das redes Ethernet:

- Providencia um meio para expedir da rede tráfego crítico em termos de tempo, ao atribuir prioridades para tramas de saída.
- Permite que estações estejam atribuídas a grupos lógicos, para comunicar através de múltiplas LANs, apesar daquelas estarem numa LAN individual, i. e, as *bridges* e os comutadores filtram os endereços de destino e direccionam as tramas da VLAN apenas para portos que servem as VLANs às quais pertence esse tráfego.
- Simplifica também a gestão da rede e facilita adições, transições e mudanças, para um administrador.

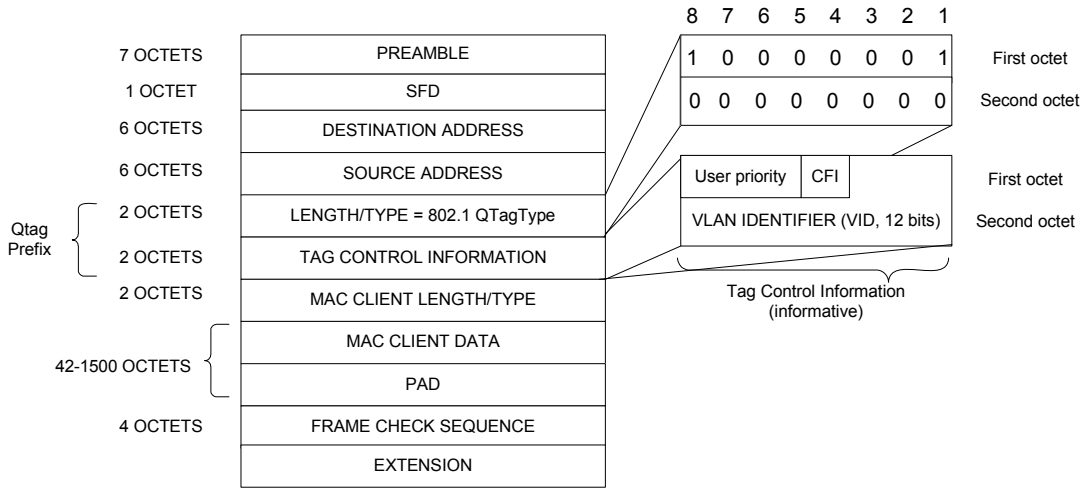


Figura 2.34 – Formato da trama MAC *Tagged*

O *Preamble* é idêntico na estrutura e na semântica ao campo do Preamble da trama básica do MAC. O SFD é idêntico na estrutura e na semântica ao campo de SFD da trama básica do MAC. Os campos de endereço de destino e endereço de origem são também idênticos na estrutura e na semântica aos campos de endereço da trama básica do MAC. O campo de *Length/Type* de uma trama etiquetada do MAC usa sempre a interpretação de tipo, e contém o tipo de protocolo do Tag 802.1Q.

O *Tag Control Information* (informativo) é subdividido por:

a) Um campo de 3 bits de User Priority;

b) Um CFI (Canonical Format Indicator), e;

c) Um VLAN Identifier de 12 bits.

A estrutura e a semântica dentro do campo de informação de controlo do Tag são definidas na norma IEEE P802.1Q.

O MAC *Client Length/Type* contém o campo original de Length/Type da trama do MAC antes da inserção do prefixo de QTag. O prefixo de QTag desloca este campo exactamente de 4 octetos da sua posição numa trama do MAC não etiquetada.

Os campos *Data* e PAD são idênticos na estrutura e na semântica aos campos de dados e de PAD da trama básica do MAC, a não ser no caso de tramas etiquetadas do MAC, em que o valor de n no cálculo do campo de PAD pode ser o comprimento dos dados do cliente do MAC ou o comprimento combinado dos dados do cliente do MAC e do prefixo de QTag.

O FCS é idêntico na estrutura e na semântica ao campo do FCS da trama básica do MAC.

O *Extension* é idêntico na estrutura e na semântica ao campo de extensão da trama básica do MAC.

3 Novas soluções para o transporte IP

3.1 Uma introdução à tecnologia Resilient Packet Ring

Como foi analisado no capítulo 1, uma tendência importante na área de comunicações é a migração de tecnologias baseadas em tráfego de pacotes das redes de área local para as redes da área metropolitana. O volume crescente de tráfego de dados nas redes metropolitanas está a desafiar os limites da capacidade das infra-estruturas de transporte existentes, baseados em tecnologias orientadas para a comutação de circuitos como SONET/SDH e ATM. As ineficiências associadas ao transporte de quantidades crescentes de tráfego de dados sobre redes de comutação de circuitos optimizadas para voz, tornam difícil a provisão de novos serviços e aumentam o custo para a adição de capacidade para além dos limites de orçamentos da maioria das operadoras. As tecnologias baseadas no transporte de pacotes são consideradas por muitos como sendo a única alternativa para "escalar" as redes da área metropolitana de forma a suportar o crescente aumento de tráfego de dados.

3.1.1 Limitações do SONET/SDH nos anéis das MANs

A maioria da fibra da área metropolitana está implementada em estruturas de anel. A topologia de anel aplica-se naturalmente às redes TDM baseadas em SONET/SDH que constituem o cerne da infra-estrutura existente das redes metropolitanas.

No entanto, há desvantagens de se usar SONET/SDH para transportar tráfego de dados (ou soluções de dados SONET/SDH ponto-a-ponto, como por exemplo o POS). O SONET/SDH foi projectado para aplicações ponto-a-ponto, de comutação de circuitos (por exemplo tráfego de voz), e a maioria das limitações advêm destas origens. Algumas das desvantagens de usar anéis SONET/SDH para o transporte de dados são:

• Circuitos Fixos: O SONET/SDH aprovisiona circuitos ponto-a-ponto entre nós do anel. Para cada circuito é reservada uma quantidade fixa de largura de banda que é desperdiçada quando não usada. Para a rede SONET/SDH que é usada para acesso (figura 3.1 - esquerda), para cada nó no anel é reservado somente um quarto da largura de banda total do anel (exemplo, quatro OC-3 num anel OC-12). Esta reserva fixa impõe um limite sobre o máximo de *burst* na taxa de transferência de dados de tráfego entre pontos extremos. Esta é uma desvantagem para o tráfego de dados, que é inerentemente *bursty*;

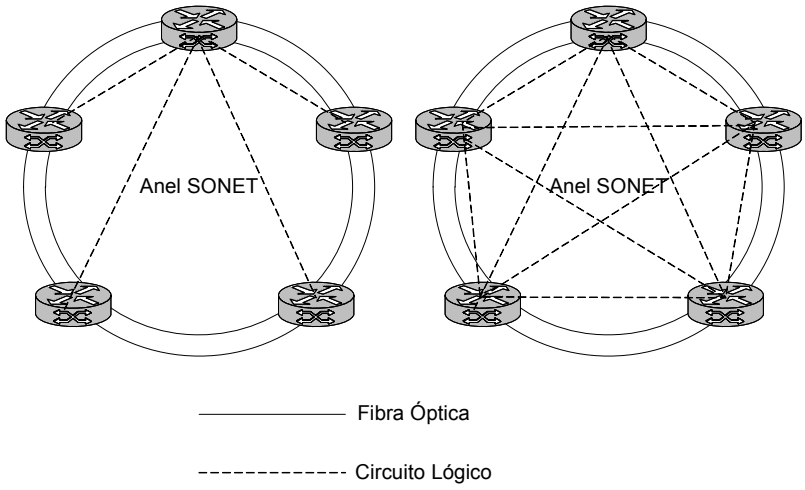


Figura 3.1 - Acesso SONET/SDH e Fully Meshed Networks

• Desperdício da largura de banda para rede em malha: Se o projecto de rede for do tipo malha lógica, (figura 3.1 - direita), o projectista da rede deve dividir o OC-12 de largura de banda do anel em dez circuitos aprovisionados. O aprovisionamento dos circuitos necessários para criar uma rede em malha lógica sobre um anel de SONET/SDH não é somente difícil, mas resulta também no uso extremamente ineficiente da largura de banda do anel, piorando à medida que a quantidade de tráfego de dados que permanece dentro das redes da área metropolitana, for aumentando. Uma rede inteiramente em malha que seja fácil de implementar, manter e melhorar está a tornar-se uma exigência importante;

• Tráfego *multicast:* Num anel SONET/SDH, o tráfego *multicast* requer que cada origem aprovisione um circuito separado para cada destino. É emitida separadamente a cada destino uma cópia do pacote, resultando em múltiplas cópias dos pacotes *multicast* que viajam em torno do anel, desperdiçando largura de banda;

• Largura de banda desperdiçada para protecção: Tipicamente, 50 por cento da largura de banda do anel é reservada para protecção. Quando a protecção for obviamente importante, o SONET/SDH não consegue este objectivo de um modo eficiente que dê ao fornecedor a escolha de quanta largura de banda deve reservar para protecção.

3.1.2 Ethernet na MAN

Para um operador, um serviço Ethernet é todo o serviço de dados feito através de uma interface Ethernet (10Mb/s, 100Mb/s, 1Gb/s). Uma diferença chave entre serviços Ethernet e serviços de dados tradicionais tais como linhas alugadas, Frame Relay ou ATM é a possibilidade de "escalar" a interface do serviço, por parte de serviços Ethernet.

Com os serviços de dados tradicionais, as exigências físicas da interface variam com a velocidade do serviço. Assim, o equipamento requerido para um serviço T1 é completamente diferente do requerido para serviços DS3 ou OC-3. Com serviços Ethernet, por outro lado, um fornecedor de serviços pode fornecer a um cliente um porto Fast Ethernet (100 Mb/s) ou um porto Gigabit Ethernet (1000 Mb/s) uma única vez, e actualizar o serviço fornecido quando necessário, sem constrangimentos adicionais em termos de equipamento, para além da instalação inicial. A largura de banda e outras mudanças de serviço podem ser administradas remotamente, simplificando e acelerando o aprovisionamento do serviço.

A tecnologia óptica Gigabit Ethernet, capaz de suportar extensões de fibra de mais de 50 Km, está a emergir como uma alternativa viável para o transporte de dados em redes públicas. Quase todos os pacotes de dados começam e terminam o seu trajecto como tramas Ethernet, transportando os dados num formato de pacote consistente, desde o começo até ao final durante todo o trajecto do transporte. Isto elimina a necessidade de camadas adicionais de protocolo e de sincronização que resulta actualmente em custos extras e em maior complexidade do sistema.

Além da manipulação eficiente de pacotes IP, a Ethernet tem as vantagens comerciais da familiaridade, da simplicidade e do baixo custo.

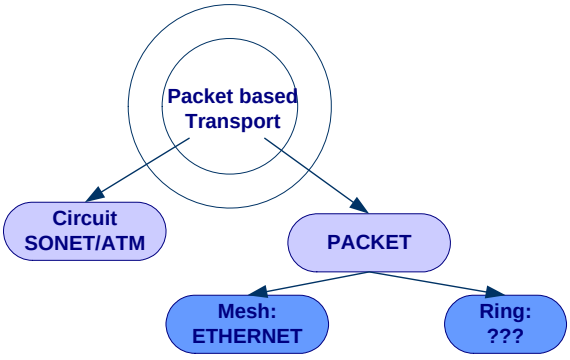


Figura 3.2 – Packet Rings: a próxima etapa de transporte baseado em pacotes

No entanto, a Gigabit Ethernet é apenas o primeiro passo na evolução do transporte baseado em pacotes na MAN.

Apesar de bem estruturada para topologias de rede ponto-a-ponto ou malha, é difícil implementar a Ethernet em configurações de anel (típico de redes de comunicação óptica) e em meios partilhados. Os anéis funcionam como meios partilhados e necessitam de mecanismos da camada MAC para gerir o acesso de múltiplos utilizadores. A Ethernet evoluiu para suportar infra-estruturas comutadas *full-duplex* e não tem MACs adequados a este cenário. Esta é uma desvantagem apreciável, a maioria da estrutura de fibra existente em áreas metropolitanas está sob a forma de anel, pois embora a tecnologia encarregada do transporte, SONET/SDH, seja implementada sobre anéis de fibra.

As topologias em anel permitem ao SONET/SDH executar um mecanismo de protecção rápido (abaixo de 50ms) que restaura a ligação usando um trajecto alternativo em torno do anel, caso a fibra quebre ou haja falha de equipamento. Ao contrário do SONET/SDH, a Ethernet não tem um mecanismo interno rápido de protecção, essencial para um operador. Há, consequentemente, grandes benefícios numa nova tecnologia que possa explorar inteiramente os anéis de fibra (em particular *ring resiliency*) e também possuir todas as vantagens inerentes de um mecanismo de transporte baseado em pacotes tipo Ethernet.

Nem os SONET/SDH ADMs nem a Ethernet obviam à necessidade de uma camada de MAC projectada para o ambiente da área metropolitana. O SONET/SDH emprega técnicas da camada física (ligações ponto-a-ponto) para controlar a capacidade de um anel. Os comutadores Ethernet baseiam-se em *Ethernet Bridging* ou em distribuição do IP para a gestão da largura de banda. Consequentemente, a rede é sub utilizada no caso de SONET/SDH, ou não determinística no caso de comutadores Ethernet.

A Ethernet também não é eficaz para implementar políticas globais de *fairness* para partilhar a largura de banda do anel. Os comutadores Ethernet podem fornecer o *fairness* ao *link-level*, mas isto não se traduz necessariamente ou facilmente em *fairness* global.

A Ethernet faz um uso eficiente da largura de banda disponível para o tráfego de dados, e oferece uma solução bastante mais simples e barata. No entanto, porque a Ethernet está optimizada para topologias ponto-a-ponto em malha, não aproveita ao máximo a topologia em anel.

Ao contrário do SONET/SDH, a Ethernet não aproveita a vantagem de uma topologia em anel para executar um mecanismo rápido de protecção. A Ethernet baseia-se geralmente no protocolo de *spanning tree* para eliminar todos os *loops* de uma rede comutada. Apesar de que o protocolo *spanning tree* possa ser utilizado para conseguir a redundância do trajecto, tem um tempo de recuperação elevado no caso de um corte numa fibra, dado que o mecanismo de recuperação requer que a condição de falha seja propagada em série a cada nó a montante. A agregação da ligação (802.1ad) pode fornecer uma solução de *resiliency* ao nível da ligação, mas é comparativamente lento com o SDH (~500ms contra ~50ms) e não é apropriado para fornecer a protecção ao nível do trajecto.

3.1.3 Resilient Packet Ring – Arquitectura da Rede Metropolitana

O RPR é uma arquitectura de rede em anel e uma tecnologia projectada para cumprir com as exigências de uma MAN baseada em tráfego de pacotes.

O problema de controlar eficazmente um recurso partilhado (neste caso o anel de fibra é o recurso que necessita ser partilhado através dos milhares de subscritores numa área metropolitana) é resolvido mais eficientemente na camada MAC do protocolo RPR.

Nos últimos anos houve desenvolvimentos na maioria das áreas metropolitanas. O desafio para os fornecedores de serviços é o de aproveitar a capacidade latente disponível nestes anéis de fibra e usá-la para rentabilizar com tantos serviços quanto possível.

Uma solução emergente para aplicações de transporte de dados na área metropolitana é a tecnologia RPR. Esta tecnologia oferece duas características chave que têm sido exclusivas do SONET/SDH: o suporte eficiente para a topologia de anel; e a recuperação rápida dos cortes da fibra e das falhas da ligação. Ao mesmo tempo, a tecnologia de anel em pacotes pode fornecer

eficiente transporte de dados, simplicidade e vantagens do custo que são típicas da Ethernet. Além disso como veremos, o RPR resolve problemas de justiça e controlo de congestionamento que não têm sido fornecidas pelas tecnologias actualmente empregues.

3.1.3.1 Características do RPR

O RPR tem diversas características originais que o tornam numa plataforma ideal para a entrega de serviços de dados em redes metropolitanas.

Uma rede metropolitana construída com comutadores Ethernet consiste em nós ligados ponto-a-ponto. O tráfego da rede é enfileirado e programado em cada nó intermédio entre a origem e o destino. Isto gera questões de escalabilidade. Cada nó tem de processar o tráfego que vem dentro da rede à taxa da linha. A tecnologia de processamento de pacotes em cada nó deve poder assegurar as taxas mais baixas de 1Gb/s ou de 2,5 Gb/s. Mas quando a taxa da rede sobe a 10 Gb/s e acima, esta abordagem falha, pois torna pouco exequível a decomposição e o processamento de pacotes a uma taxa tão elevada, mantendo o mesmo nível de equipamento e com os mesmos níveis de serviço que a taxas mais baixas.

A norma do RPR cria um novo protocolo de MAC projectado para topologias baseadas em anel, um anel de pacotes, sem depender da camada física. Nesta camada, as tecnologias de pacote em anel são compatíveis com normas da camada física como Ethernet, SONET/SDH e DWDM.

A vantagem básica dos anéis de pacotes é que cada nó pode assumir que um pacote emitido no anel alcançará eventualmente o seu nó de destino independentemente do trajecto que faça em torno do anel. Desde que os nós se identifiquem que estão num anel, são apenas necessárias três acções básicas para processamento de pacotes: *insertion* (adiciona um pacote no anel), *forwarding* (emite o pacote avante), e *stripping* (retira e decompõe o pacote para fora do anel). Isto reduz a quantidade de trabalho que os nós individuais têm que fazer para se comunicarem entre eles, especialmente em comparação à rede em malha onde cada nó tem que tomar uma decisão de *forwarding* sobre que porto de saída deve usar para cada pacote.

Os anéis de pacotes têm uma vantagem natural de restauração (*resiliency*). No caso da Ethernet, isto é suportado pelo protocolo *spanning tree*. Este mecanismo de restauração é relativamente lento. Na prática, os sistemas de transporte baseados em anel conseguem garantidamente menos de 50ms de período de falha. Um protocolo para anéis de pacotes pode iniciar o "*ring wrap*" nos nós que rodeiam o corte, ou o "*steering*" do pacote fazendo com que o nó de emissão redireccione os pacotes. Num ou noutro caso o tráfego pode alcançar o destino original circundando o anel no sentido oposto ao do local de um corte da fibra.

Os anéis de pacotes têm também uma vantagem inerente para executar algoritmos de *fairness* que servem para regular o uso da largura de banda. Um algoritmo de *fairness* é um mecanismo que atribui a cada cliente no anel uma parte "justa" da largura de banda do anel, idealmente sem a desvantagem de aprovisionar um circuito. A largura de banda do anel é um recurso partilhado, e é vulnerável à exploração por utilizadores individuais ou por nós. Um algoritmo de *fairness* ao nível do anel pode e deve tratar a largura de banda do anel como um recurso global. As políticas de largura de banda podem permitir que a largura de banda máxima do anel seja utilizada entre quaisquer dois nós quando não há qualquer congestionamento e podem ser executadas sem a inflexibilidade de um sistema baseado em comutação de circuito fixo como SONET/SDH, e com uma eficácia maior do que o ponto-a-ponto da Ethernet. De lembrar que o SONET/SDH é também implementado com circuitos ponto-a-ponto que reservam uma quantidade fixa de largura de banda para cada ligação, mas a falta de flexibilidade é o problema. Adicionar ou subtrair largura de banda requer a configuração manual de circuitos novos, e a reserva de tais circuitos desperdiça largura de banda. Por sua vez os comutadores Ethernet ou os SONET/SDH ADMs não têm qualquer potencialidade de gestão da largura de banda e assim não podem maximizar a utilização da rede.

Numa rede com padrões de tráfego em mudança constante (que é típica de todas as redes de pacotes), a única maneira de optimizar a utilização da mesma sem haver tráfego rejeitado é a de ter um mecanismo de *feedback* na própria rede. O mecanismo de *feedback* informa as fontes do tráfego da capacidade disponível na rede de modo a que essas fontes possam ajustar a taxa à qual injectam o tráfego na rede. A entidade MAC em cada nó monitoriza a utilização nas suas

ligações e torna essa informação disponível a todos os nós no anel. Cada nó pode então emitir mais dados ou estrangular essa emissão. O MAC é a entidade que executa estas funções de uma forma automática.

Os anéis de pacotes são uma solução natural para *broadcast* e para tráfego *multicast*. Como detalhado acima, para o tráfego *unicast*, os nós num anel de pacotes têm geralmente a escolha de fazer o *stripping* dos pacotes do anel ou o *forwarding* dos mesmos. No entanto, para *multicast*, os nós podem simplesmente receber o pacote e enviá-lo, até que o nó de fonte faça o *strip* do pacote. Isto torna possível ao *multicast* ou ao *broadcast* a emissão de somente uma cópia de um pacote em torno do anel.

Uma infra-estrutura de transporte baseada no RPR pode ser utilizada para transportar o leque total de serviços actuais baseados em SONET/SDH. Uma VPN, serviços ATM, Frame Relay, ou POS podem todos ser transportados transparentemente sobre RPR de uma maneira tão fiável quanto o SONET/SDH, e tão rentável quanto a tecnologia de transporte de pacotes.

3.2 Tendências

Como tem sido referido, as redes terão que lidar com as exigências crescentes de largura de banda. As taxas da transmissão de TDM estão a crescer para 40 Gbit/s e ao mesmo tempo o número de canais num sistema de DWDM está a crescer para 128 ou mais canais. Assim, as taxas totais de capacidade de 320 Gbit/s, 1 Tbit/s e acima serão possíveis.

Como a maioria dos novos serviços de telecomunicações são *data-oriented,* a transmissão nestas redes deve ser optimizada para dados. Isto é feito pelas MSPP, que permitem a agregação e a comutação de pacotes de dados por um elemento da rede.

As MSPPs comutam voz e vídeo assim como diferentes tipos de tráfego de dados. Há três características principais que são importantes para se poder executar o acima descrito:

1. Uso da tecnologia de DWDM para aumentar a capacidade das fibras;
2. Uso do SONET/SDH que assegura QoS para aplicações sensíveis no tempo;
3. Uso de mecanismos inteligentes (*switches* e *routers*) para os dados da camada 2 e da camada 3 que permite que os produtos comutem e distribuam o tráfego.

Combinando estas características num instrumento, permite que os fornecedores de serviços simplifiquem as suas redes. Em vez de instalar um *multiplexer* de DWDM, um ADM e um *router* de IP, podem usar apenas um dispositivo. Isto reduz o número dos dispositivos numa rede e resulta num número diminuto dos problemas na mesma.

3.2.1 Soluções propostas para a próxima geração de MANs

Muitos produtos são tidos actualmente como a solução certa para as necessidades da próxima geração de MANs. Estes produtos podem ser classificados em três categorias: (1) SONET/SDH MSPPs; (2) comutadores de pacotes da camada 2/3; e (3) plataformas DWDM da área metropolitana.

O legado da rede optimizada para voz e do equipamento baseado em SONET/SDH continua a existir inclusive em muitos produtos novos, supostamente optimizados para tráfego IP. Muitas destas novas tecnologias são baseadas em novos mapeamentos de *payloads* comutados por circuito em tramas SONET/SDH.

Tais produtos - SONET/SDH MSPPs – mapeiam tráfego dentro de circuitos (*pipes*) tais como SONET STS-Ns ou novos circuitos proprietários. Alguns destes produtos de circuitos SONET/SDH suportam funções de comutação de pacotes da camada 2 e da camada 3, mas executam-nos como camadas adicionais em cima das camadas de circuitos SONET/SDH. Neste sentido são comutadores e *routers* em cima de *multiplexers* SONET/SDH Add/Drop (ADMs) – mesmo que esses ADMs usem mapeamentos de *payload* SONET/SDH não normalizados.

Estas soluções não podem garantir a utilização óptima da largura de banda para serviços de voz e de dados, pois que a largura de banda não utilizada num dado circuito STS-N é desperdiçada. Acrescentando um *router* ou um comutador da camada 2 em cima de um SONET/SDH ADM, não se elimina o custo e a complexidade associados com estas múltiplas tecnologias. Também, a presença de múltiplas camadas tecnológicas, não elimina a necessidade de controlar estas mesmas camadas, tornando difícil para os portadores os escalonamentos das suas redes de multi-serviços.

Uma segunda classe de novas soluções de transporte na MAN usa os produtos *packet-switched* nativos construídos para a LAN. Estes são baseados em comutadores de camada 2 ou *routers* de camada 3, ligados ponto-a-ponto que transportam Ethernet, POS ou qualquer outro protocolo. Algumas destas soluções – particularmente os comutadores Gigabit Ethernet – são capazes de utilizar eficientemente a largura de banda e fornecem uma manipulação relativamente sofisticada de serviços de dados diferenciados. No entanto, estas abordagens não fornecem a QoS e a robustez requeridas para o transporte de serviços críticos de tempo real, dado que não garantem a entrega *extremo-a-extremo* do pacote com características bem controladas de latência, de *jitter*, de largura de banda e de perda de pacote. Em particular, soluções puras de Gigabit Ethernet não oferecem aos portadores a opção de fornecer dados e serviços de alta qualidade (T1/E1, DS3, etc.) numa única plataforma.

Um terceiro grupo de vendedores está a tentar adaptar as tecnologias ópticas de transporte no *backbone* da WAN para as redes MAN. As plataformas de transporte metropolitano DWDM fornecem um aumento significativo na capacidade do anel baseada na sua capacidade de multiplexar 32 ou mais sinais sobre uma única fibra ao mapear cada sinal a um comprimento de onda diferente. A maioria destas plataformas fornece alguma forma de comutação básica na protecção do anel, e podem transportar uma variedade de diferentes tipos de tráfego de uma maneira protocolar independente. Mas porque estas plataformas metropolitanas DWDM tipicamente mapeiam uma dada trama de entrada no seu próprio comprimento de onda, não fornecem meios para uma agregação eficiente e flexível de uma taxa mais baixa nos caros equipamentos DWDM.

Alguns vendedores de plataformas metropolitanas DWDM de segunda geração estão a aplicar as funcionalidades dos ADM ou *cross-connect* SONET/SDH sobre a camada WDM para conseguir alguma agregação e funcionalidade dinâmica no aprovisionamento na forma de anéis múltiplos SONET/SDH, com cada anel a usar um comprimento de onda. No entanto, ao fazer isto sacrificam a independência do protocolo e incorrem em todos os custos, ineficiências e inflexibilidade das plataformas SONET/SDH.

3.3 Soluções de algumas companhias de telecomunicações

Nesta secção são apresentadas sucintamente algumas soluções implementadas por alguns fabricantes de equipamento de transporte de tráfego IP/Ethernet que são usadas nas redes de comunicações actuais.

Existem algumas outras soluções de outras empresas que poderiam ser apresentadas nesta secção, similares quanto ao objectivo pretendido, apesar de usarem protocolos proprietários dessas mesmas empresas.

Estes exemplos são representativos de transporte de serviços Ethernet, e das soluções encontradas, algumas das quais com arquitecturas baseadas em conceitos semelhantes ao RPR.

3.3.1 Appian Communications

Esta empresa apresenta o OSAP™ (Optical Services Activation Platform™) que é uma solução optimizada para auxiliar os fornecedores de serviços a migrar dos serviços tradicionais de TDM para serviços de transmissão de pacotes. É um comutador de camada 2 que também percorre a camada 3, e que apresenta a Ethernet como uma interface universal de serviços de transmissão de pacotes. É remotamente ajustável em incrementos de 64 Kb/s, desde 64 Kb/s a 1 Gb/s. As suas potencialidades avançadas de QoS permitem que os clientes contratem uma GBR explícita

e/ou um MBR, permitindo acordos de serviço que combinam os existentes em ambientes Internet, Frame Relay e ATM.

Quando são adicionadas QoS e outras tecnologias protocolares, a Ethernet pode ser usada como um serviço universal que liga uma empresa ao seu ISP, a um ASP e a *sites* Intranet empresariais regionais. O mesmo cliente pode implementar uma Ethernet VPN que suporta as operações à velocidade da LAN que têm exigências na mudança da largura de banda durante todo o dia.

Com as suas potencialidades de mediação protocolar, o OSAP torna também a Ethernet numa solução de multi-serviços que fornece um único ponto de entrada de elevada velocidade a serviços tais como o acesso Internet, Frame Relay, Ethernet nativo e ATM. A complexidade e o custo de tratar o acesso às tecnologias WAN, separadamente para cada tipo de serviço, são eliminados. A sua orientação para serviços TDM permite que os fornecedores continuem a explorar as aplicações originais de voz e de linhas de dados privadas, enquanto vão migrando para os novos serviços baseados em transmissão de pacotes.

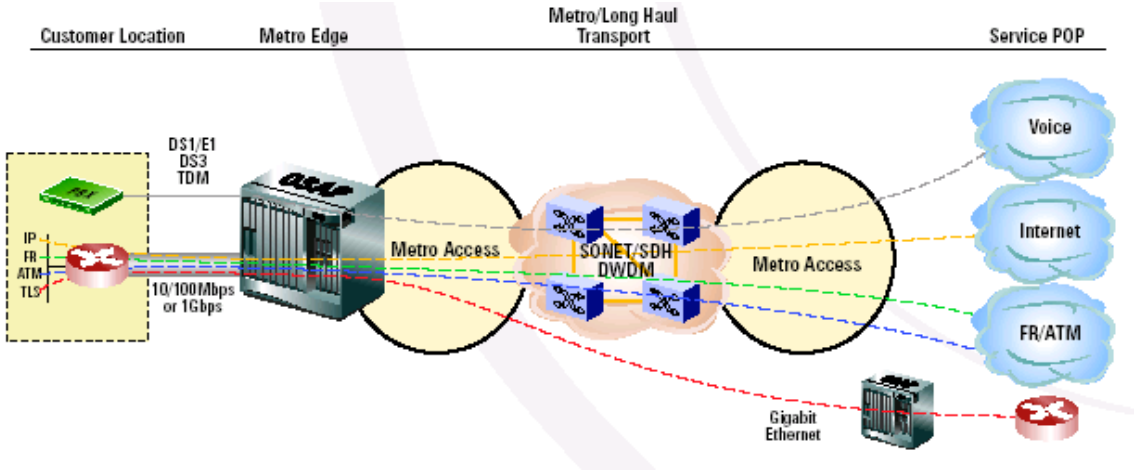


Figura 3.3 – Apian OSAP™: serviço Ethernet de interface de pacotes universal para o cliente

Há nesta solução, objectivamente, uma oportunidade para os serviços de LAN-a-LAN que combinam a largura de banda elevada da Ethernet com a qualidade de serviço e o escalar da capacidade das redes SONET/SDH para estender a colaboração da elevada largura de banda, da transmissão vídeo de qualidade e das aplicações de *recovery*/*backup,* através de uma rede global.

Os fornecedores de serviços podem também migrar os pequenos e médios negócios, de circuitos T1/E1 e T3/E3 tradicionais para as EPL. Estas EPLs são ligações seguras que podem suportar quaisquer serviços de voz, dados ou aplicações de vídeo que o cliente tem estado actualmente a utilizar sobre uma linha confidencial dedicada. As EPLs podem ser ajustadas por *software* em incrementos granulares da largura de banda com activação remota do serviço, o que elimina a deslocação/instalação caras e demoradas.

3.3.2 Corrigent Systems

O tráfego Ethernet, assim como outras formas de tráfego de dados, é actualmente transportado sobre redes tradicionais baseadas em SONET/SDH. Recentemente, a tecnologia Ethernet foi proposta para ser usada como uma tecnologia de transporte na rede pública, onde os serviços Ethernet são transportados de uma maneira mais rentável do que SONET/SDH. A solução apresentada pela Corrigent Systems é baseada nos conceitos de RPR.

3.3.2.1 Arquitectura Corrigent

A solução da Corrigent Systems transporta os serviços Ethernet SONET/SDH na sua forma nativa, sobre mecanismos RPR dentro de uma trama SONET/SDH, com um plano de controlo MPLS que fornece a gestão *extremo-a-extremo* do trajecto através dos múltiplos domínios da rede.

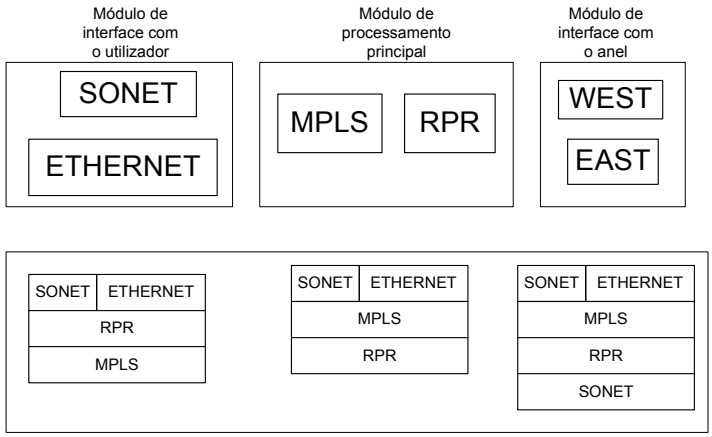


Figura 3.4 – Arquitectura Corrigent

3.3.2.2 Comutador distribuído baseado no RPR

Numa arquitectura de comutação distribuída baseada no RPR a agregação é executada localmente – tanto no tráfego do nó como no meio partilhado do anel.

É requerido um *hub* para fazer o *backhaul* de tráfego inter-metropolitano para a rede de *backbone*, mas não para a comutação de tráfego intra-metropolitano.

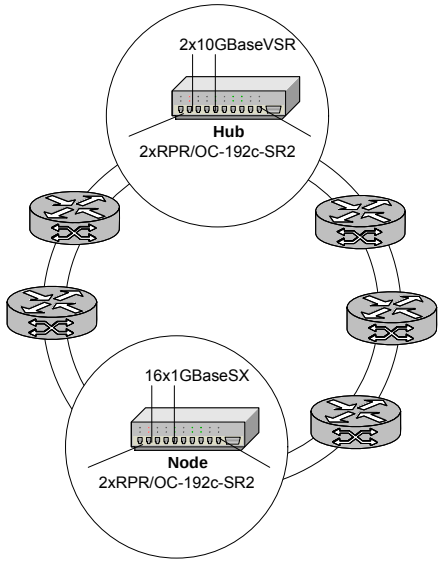


Figura 3.5 - Rede baseada no RPR

Neste cenário específico, cada ADM Corrigent baseado no RPR agrega interfaces de utilizador de 16x1GbE no meio partilhado do anel (transporta tramas RPR sobre um PHY OC192c), e o nó *hub* suporta o tráfego que parte usando os *uplinks* 2x10GbE.

3.3.2.3 Resultados

A tabela 3.1 ilustra as comparações da solução da Corrigent de 10Gb/s baseada em RPR com uma solução puramente baseada em Ethernet 10Gb/s. Uma arquitectura distribuída baseada em RPR tem um custo mais baixo comparada ao transporte puro de Ethernet, em aproximadamente *60%,* mesmo se o custo da fibra não é tido em conta. A mesma solução de RPR seria de mais baixo custo do que uma solução baseada em OC192 com Ethernet-sobre-SONET/SDH, em aproximadamente 75%.

	Ethernet sobre SONET	Ethernet	RPR
Pares de fibras	8	8	1
Portos ópticos *Trunk*	32x OC-192	32x 10 GbE	18x RPR/OC-192c
Trunk Reach	LR	LR	SR-2
Portos GbE	320	128	128

Tabela 3.1 –Principais factores de custo

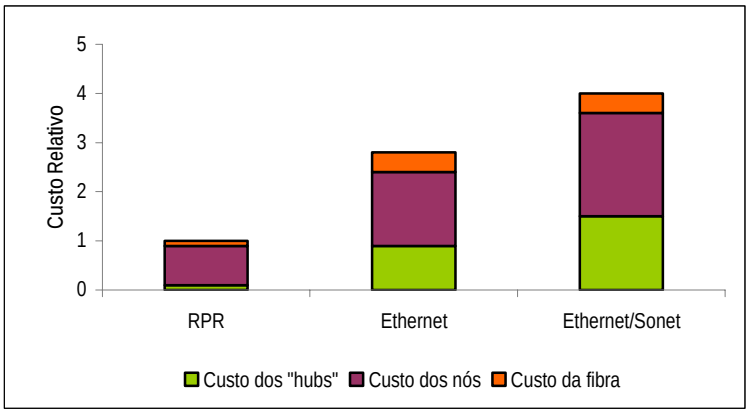


Figura 3.6 – Gráfico de custos

3.3.3 Luminous Networks

As MANs devem estar optimizadas para o transporte de tráfego IP, pois será o tipo de tráfego dominante transportado nestas redes. A optimização para o transporte IP permite a criação e a sustentação flexíveis de novos serviços, e minimiza o equipamento e custos operacionais. A maneira óptima de transportar tráfego IP é sobre uma rede *packet-switched* com um mínimo de circuitos de camadas intermédias. Isto significa um custo mais baixo, mais escalável, mais eficiente e mais flexível.

Embora as MANs devam ser optimizadas para o transporte IP, uma solução bem sucedida da MAN deve suportar os serviços tradicionais, isto é, dados *circuit-switched* de voz e vídeo. Os circuitos de voz de qualidade requerem uma sincronização *extremo-a-extremo*.

Uma rede *packet-switched* pode obter garantias aceitáveis de QoS somente através de um sistema operando na rede, em que o planeamento inteligente do tráfego é integrado firmemente com mecanismos subjacentes de gestão de recursos. A adição de camadas *circuit-switched* extras, adiciona complexidade e custos desnecessários ao sistema, diminui a eficiência da rede, e torna mais complexo e demorado o aprovisionamento da rede.

3.3.3.1 Luminous Networks PacketWave: solução óptica Ethernet MAN

A solução PacketWave é uma plataforma flexível de transporte de portadora de classe que suporta uma larga variedade de interfaces de tributários incluindo a emulação de circuitos T1, DS3 e SONET/SDH, Ethernet 10/100BaseT e Gigabit, e OC3 a OC48 POS.

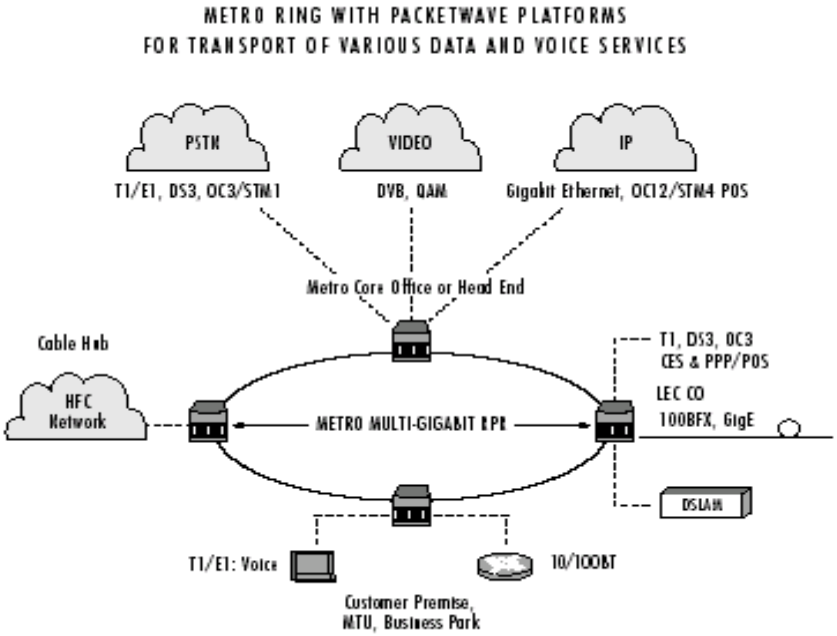


Figura 3.7 – Plataforma PacketWave

A PacketWave utiliza uma camada de transporte *packet-switched* óptica chamada RPT (Resilient Packet Transport™) que é optimizada para o transporte eficiente e robusto de IP. O RPT junta uma camada MAC de MPLS/Ethernet sobre uma camada NxGigabit Ethernet ou uma camada física de SONET/SDH, com a adição de mecanismos robustos de serviço tais como a monitorização de desempenho, a sincronização da rede e pacotes de controlo. O RPT alcança a robustez do transporte ao nível do SONET/SDH a custos de Ethernet Gigabit. Todo o tráfego é multiplexado estatisticamente em pacotes da camada 2 e ambos os sentidos do anel são utilizados inteiramente, fornecendo uma eficiência óptima no transporte de tráfego IP.

A camada de transporte óptica pode ser transportada num único ou em múltiplos comprimentos de onda. O uso de configurações flexíveis *add-drop* ópticas permite o desacoplamento da topologia física e lógica da rede. Isto fornece a habilidade de suportar redes em anel, malha e topologias lineares sobre um anel físico. Esta potencialidade de múltiplos comprimentos de onda não tem qualquer impacto no custo da instalação, e cada comprimento de onda pode ser adicionado incrementalmente.

A arquitectura *packet-switched* do PacketWave fornece um suporte extensivo de QoS, incluindo policiamento, *shaping*, enfileiramento *class-based* e *scheduling* flexível baseado em prioridade estrita, WRR (Weighted Round Robin) e outros algoritmos. A arquitectura de comutação suporta também trajectos de pacote de latência mínima para os pacotes que passam através de um nó.

O PacketWave permite a sincronização do *stratum* do tráfego TDM, equivalente a SONET/SDH sobre SONET/SDH ou sobre camadas físicas Gigabit Ethernet (8B/10B).

Isto permite que serviços e circuitos *extremo-a-extremo* T1/E1 ou SONET/SDH exibam as mesmas estabilidade e qualidade como em redes SONET/SDH. Suporta também a protecção e a restauração em menos de 50ms através dos mecanismos de protecção da camada 2 que permitem que 100% da largura de banda da rede seja utilizada quando não há qualquer ruptura da extensão. Pelo contrário, o SONET/SDH tradicional requer sempre a reserva de uma quantidade grande de largura de banda da rede para protecção.

4 Resilient Packet Ring – IEEE 802.17

O desenvolvimento de um MAC novo é um empreendimento enorme, e o grupo de trabalho do IEEE 802.17 tem feito um progresso significativo no desenvolvimento da norma do RPR IEEE 802.17 com as diversas aprovações dos sucessivos esboços completos. A versão 3.3 do esboço do RPR IEEE 802.17, é a última disponível aos membros do grupo de trabalho no *website* do IEEE 802.17.

Houve vários critérios para o estudo e implementação deste protocolo, entre os quais o potencial de mercado existente, a compatibilidade e uma identidade distinta de outros protocolos do IEEE 802, e também as exequibilidade técnica e viabilidade económica.

O potencial de mercado surgiu com o enorme conjunto de aplicações existente, com os inúmeros vendedores e utilizadores, e com os custos equilibrados entre o número de LANs e as estações ligadas. Foi também um factor determinante a completa compatibilidade com a arquitectura 802, com os 802.1D, 802.1Q, 802.1f, e com outros protocolos de gestão de sistemas. No entanto o 802.17 é um protocolo distinto de outros protocolos do IEEE 802 pela forma como implementa um MAC para a resolução de um problema específico. A exequibilidade técnica e a viabilidade económica foram demonstradas por diversas implementações de produtos comerciais por parte de alguns fabricantes, operadores e grupos de investigação.

4.1 Sumário

O RPR (Resilient Packet Ring) é uma tecnologia da MAN que:

- Suporta a transferência de dados entre estações interligadas numa configuração em anel-duplo tipo BLSR/2;
- Suporta transferências de dados *Unicast*, *Multicast* e *Broadcast;*
- Permite várias qualidades de serviço, i. e, contém protocolos de controlo de fluxo por qualidade de serviço que regulam o tráfego inserido pelo cliente;
- Suporta estratégias para aumentar as larguras de banda efectivas para além das de um anel de *broadcast*;
- Contém algoritmos de justiça (*fairness*) que garantem a distribuição apropriada do tráfego oportunístico;
- Suporta a descoberta automática da topologia, a inicialização dos parâmetros opcionais e a divulgação das capacidades da estação, o que permite que os sistemas se tornem operacionais sem intervenção manual.

Todas estas características permitem transmissões robustas da trama pelo anel, de e para os clientes das estações RPR.

4.2 Estrutura em anel

O RPR emprega uma estrutura em anel usando *ringlets* contra-rotacionais e unidireccionais. Cada *ringlet* é composto por ligações com o fluxo de dados no mesmo sentido. Os *ringlets* são identificados como *ringlet0* e *ringlet1*.

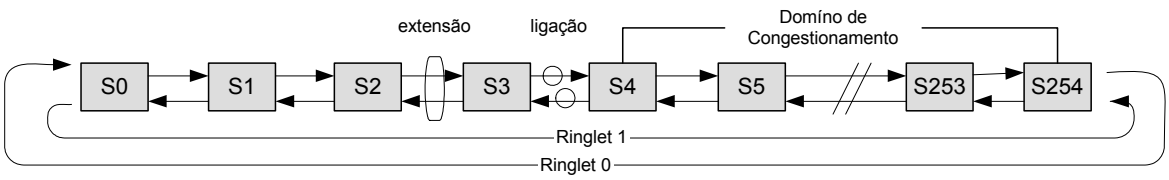


Figura 4.1 – Estrutura em anel duplo

As estações no anel são identificadas por um endereço MAC de 48-bit, especificado no IEEE Std 802-2002 [2].

O troço de um anel limitado por estações adjacentes é definido como uma extensão, e é composto por ligações unidireccionais que transmitem dados em sentidos opostos. Um grupo de estações contíguas afectadas por um ponto comum de congestionamento é chamado um domínio de congestionamento.

Os dispositivos RPR implementam a noção de um trajecto de tráfego. A entidade do MAC em cada nó executa três funções, "*add*" para a inserção de tráfego do subscritor do nó, "*drop*" ou remoção de tráfego destinado a um subscritor no nó, e "*pass*" ou transferência directa do tráfego em trânsito de uma ligação da rede a outra. Em cada nó, o tráfego que não é destinado a esse nó simplesmente atravessa o nó; não é enfileirado nem processado. O trajecto do trânsito transforma-se efectivamente numa parte do meio de transmissão e faz o anel RPR comportar-se como um meio contínuo partilhado por todos os nós RPR. Dado que um nó ADM (Add Drop Multiplexer), não processa o tráfego em trânsito, a arquitectura de pacote ADM pode escalar mais facilmente para taxas de dados mais elevadas.

4.3 Tramas RPR

Para sistemas de comunicação de dados que usam o MAC do RPR, são especificados os seguintes formatos de trama (ver anexo A – Formato das tramas RPR):

- trama de dados;
- trama de controlo;
- trama de fairness;
- trama de idle.

As tramas de dados são identificadas por um campo de 2 bits, denominado *ft* (*frame type*) e incluído no byte *baseControl*, com o valor FT_DATA.

Quando os campos *daExtended* e *saExtended* (fig. 4.2) são incluídos, a trama de dados é classificada de trama de dados estendida, quando os mesmos não estão incluídos, este tipo de trama é classificada de trama de dados básica.

O número de bytes reservados é de 92 para a trama básica e de 80 para a trama estendida. Os bytes reservados para tramas regulares com tamanho máximo e para tramas jumbo, não estão incluídos na trama transmitida ou recebida. Existem apenas na definição do comprimento e estão projectados para permitirem evoluções futuras nos cabeçalhos. O campo *serviceDataUnit* (fig. 4.2) não pode usar os bytes reservados, e não pode crescer para além dos 1500 bytes para uma trama regular ou 9100 bytes para uma trama jumbo.

a) formato da trama básica

1 byte	ttl	cabeçalho
1 byte	baseControl	
6 bytes	da	
6 bytes	sa	
1 byte	ttlBase	
1 byte	extendedControl	
2 bytes	hec	
2 bytes	protocolType	payload
N bytes	serviceDataUnit	
4 bytes	fcs	trailer

b) formato da trama estendida

1 byte	ttl	cabeçalho
1 byte	baseControl	
6 bytes	da	
6 bytes	sa	
1 byte	ttlBase	
1 byte	extendedControl	
2 bytes	hec	
6 bytes	daExtended	payload
6 bytes	saExtended	
2 bytes	protocolType	
N bytes	serviceDataUnit	
4 bytes	fcs	trailer

Figura 4.2 – Formatos da trama de dados

As definições e informação contidos nos diversos campos são sumariamente as seguintes: *ttl* – número de estações percorridas até ao destino; *baseControl* – tipo de trama, classe de serviço e controlo base do protocolo; *da* – endereço da estação destino; *sa* – endereço da estação de origem; *ttlBase* – cópia do *ttl*, para o cálculo do número de estações até à origem; *hec* – CRC de 16 bit do cabeçalho; *protocolType* – função e forma do campo *serviceDataUnit*; *serviceDataUnit* – dados providenciados pelo cliente; *fcs* – CRC de 32 bit para os campos *protocolType* e *serviceDataUnit*.

As tramas de controlo são identificadas pelo campo de 2 bits, *ft*, com o valor FT_CONTROL. Uma trama de controlo pode ser uma trama *broadcast* ou *unicast*.

Os 92 bytes reservados para uma trama de controlo com tamanho máximo, não estão incluídos na trama transmitida ou recebida. Existem apenas na definição do comprimento, e estão projectados para permitirem evoluções futuras nos cabeçalhos. O campo *controlDataUnit* (fig. 4.3) não pode usar os bytes reservados, e não pode crescer para além dos 1500 bytes.

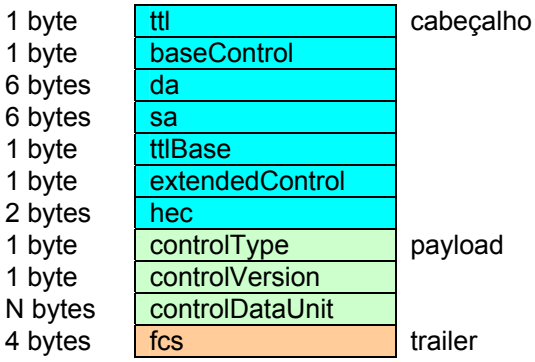


Figura 4.3 – Formato da trama de controlo

As tramas de *fairness* são identificadas pelo campo de 2 bits, *ft*, igual a FT_FAIRNESS.

A trama de *fairness* é enviada para os MACs vizinhos para fornecer dados ao algoritmo de *fairness* desses MACs.

O formato da trama de *fairness* é diferente dos formatos das tramas de dados e de controlo. As tramas de *fairness* não são enviadas para nós de destino específicos, mas sim para a estação vizinha mais próxima ou difundidas (*broadcast*) para todo o anel. Portanto, o endereço de destino não contém qualquer informação útil e é omitido. As tramas de *fairness* são mantidas o mais pequenas possível para reduzir o *jitter* noutras tramas, para reduzir o seu consumo efectivo de largura de banda e para minimizar os requisitos de memória para armazenar múltiplas tramas de *fairness* (especialmente quando são usados os algoritmos de *fairness multi choke*).

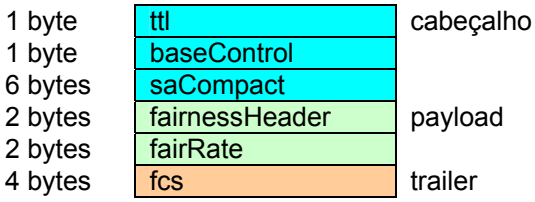


Figura 4.4 – Formato da trama de fairness

As tramas de *idle* são identificadas pelo campo de 2 bits, *ft*, igual a FT_IDLE.

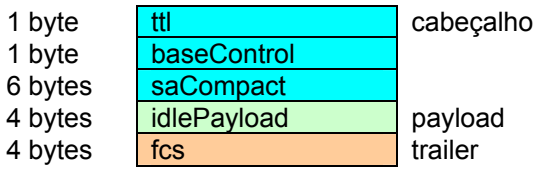


Figura 4.5 – Idle frame format

A trama de *idle* é enviada para os MACs vizinhos para ajustar o ritmo de sincronização entre a estação e as estações vizinhas.

O formato de trama para tramas *idle* é diferente do formato de trama para tramas de dados e de outras tramas de controlo. As tramas *idle* não são enviadas para nós específicos de destino, mas são enviadas para a estação vizinha mais próxima. Portanto, o endereço de destino não contém qualquer informação útil e é omitido. As tramas de *idle* são mantidas a um tamanho fixo pequeno, para reduzir o *jitter* noutras tramas e para reduzir o seu consumo efectivo de largura de banda.

4.3.1 Formatos das tramas do algoritmo de Fairness, Topologia e Protecção

O *payload* do formato da trama de *fairness* contém os valores de *fairnessHeader* e de *fairRate*.

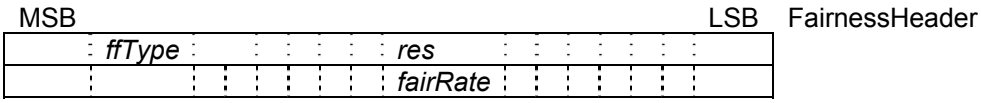


Figura 4.6 — *Payload* da trama de *fairness*

A trama TP (*Topology & Protection*) de comprimento fixo é identificada pelo campo de *controlType*=CT_TOPO_PROT. Os índices da trama TP incluem a informação para sinalizar o estado de protecção da ligação, para a descoberta da topologia física do anel e para relatar a informação das preferências da estação.

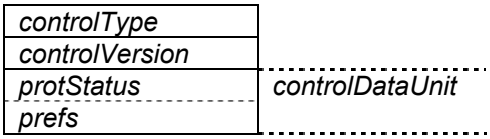


Figura 4.7 — *Payload* TP

As tramas TC (*Topology Checksum*), de comprimento fixo, são identificadas pelo campo *controlType*=CT_TOPO_CHKSUM. Estas tramas comunicam o *checksum* da topologia às estações vizinhas de modo a que as estações possam determinar se as parcelas relevantes das suas bases de dados da topologia coincidem, e consequentemente, se se podem retirar do conteúdo de contexto.

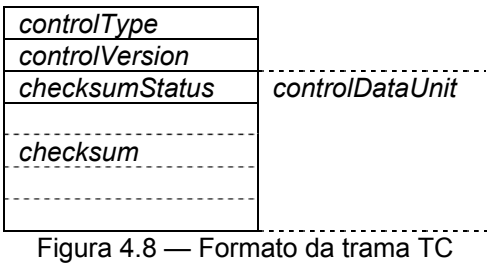


Figura 4.8 — Formato da trama TC

As tramas de LRTT (*Loop Round Trip Time*) com comprimento fixo são identificadas pelo campo *controlType*=CT_LRTT_REQ. As estações que usam o ajuste de taxa conservador gerarão tramas de LRTT. Todas as estações executarão o processamento de tramas de pedido de LRTT.

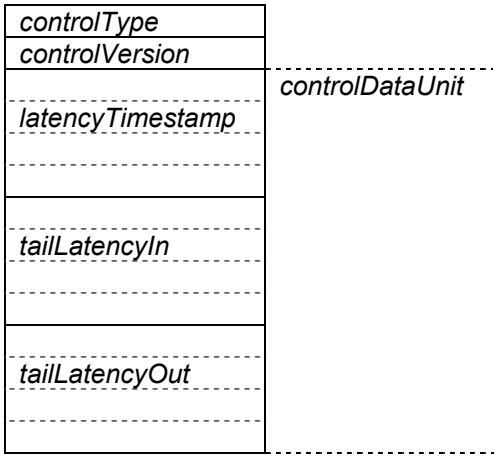


Figura 4.9 — *Payload* da trama de pedido de LRTT

As tramas de LRTT (*Loop Round Trip Time*) com comprimento fixo são identificadas pelo campo *controlType*=CT_LRTT_RSP. Todas as estações executarão a geração da trama de resposta LRTT.

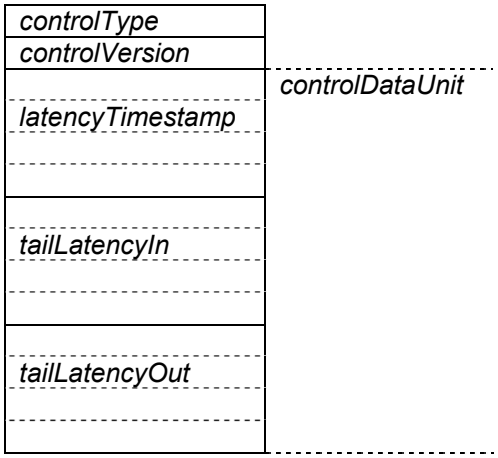


Figura 4.10 — *Payload* da trama de resposta à LRTT

As tramas de ATD (*Attribute Discovery*) com comprimento fixo são identificadas pelo campo *controlType*=CT_STATION_ATD.

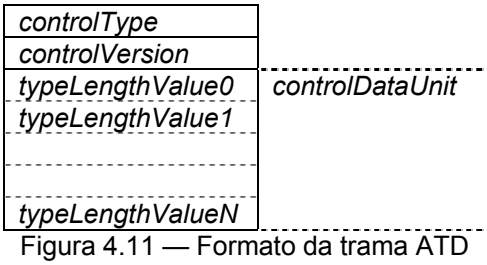


Figura 4.11 — Formato da trama ATD

4.4 Estrutura da estação

Uma estação é composta por uma entidade cliente, uma entidade MAC e duas entidades PHY. A entidade MAC contém uma entidade de controlo e duas entidades de *datapath*.

O MAC suporta somente uma única interface de serviço e assume que há uma única entidade cliente unida a essa interface. No entanto, cada cliente pode conter múltiplas entidades de clientes secundários, transformando-se efectivamente numa estação multi-função.

As funções da interface cliente do MAC são usadas apenas pelas estações nas quais são adicionadas tramas ou através das quais as tramas são transferidas desde o anel, correspondem portanto às estações de origem e de destino. Esta interface faz a adaptação de um cliente Ethernet ao ambiente do MAC RPR.

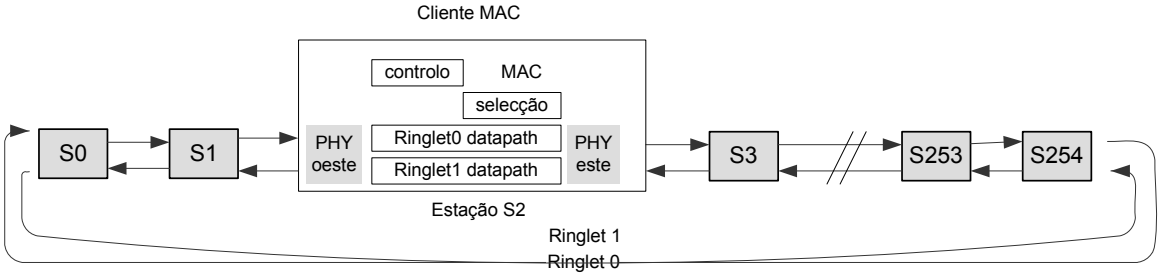


Figura 4.12 – Estrutura da estação

4.5 Arquitectura do MAC

Cada unidade de *datapath* do MAC serve um dos dois *ringlets*. A entidade de controlo do MAC emite tramas para a unidade de selecção do *ringlet*, e recebe tramas de cada uma das duas unidades de *datapath* do MAC.

A unidade de selecção do *ringlet*, verifica os endereços de origem e de destino (e outros parâmetros), para seleccionar que *ringlet(s)* transmite(m) a trama, se e como fazer o *flood* da trama, como proteger a trama e que formato da trama (básico ou estendido) é usado.

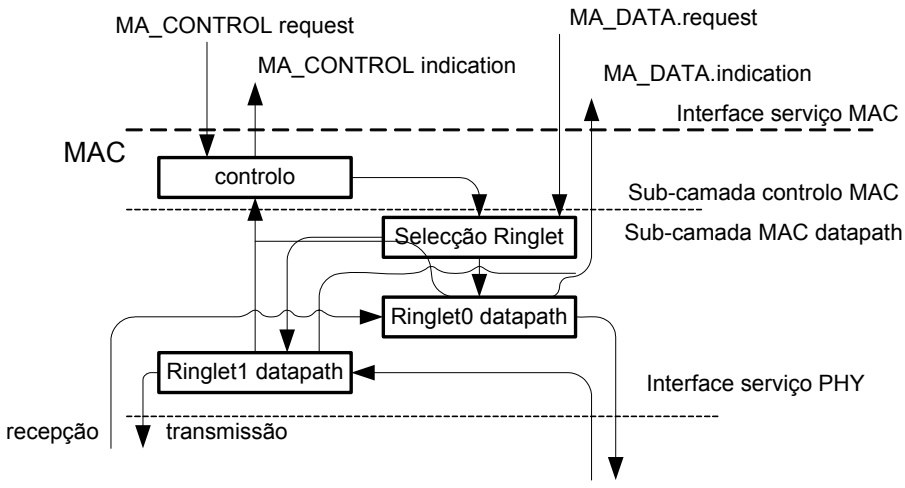


Figura 4.13 – Arquitectura de uma estação simples

O *datapath* do MAC fornece a lógica de verificação que determina que tramas são decompostas, e que filas de trânsito encaminham as tramas recebidas que aguardam para ser transmitidas. As

estações têm uma PTQ (Primary Transit Queue), tipicamente com somente alguns MTUs de tamanho que encaminha tráfego prioritário, e podem, se for uma implementação de fila-dupla, ter também uma STQ (Secondary Transit Queue) tipicamente muito maior que encaminha tráfego com prioridade mais baixa.

As tramas são transmitidas desde o cliente para uma fila de estágio (*Stage Queue*) residente no MAC, ao invés de serem transmitidas directamente do cliente, isto para desacoplar os sincronismos da interface MAC-para-cliente dos sincronismos da interface da camada física.

A funcionalidade da recepção inclui as funções de verificação, ajuste e filtragem. As regras de verificação são responsáveis por rejeitar tramas com erro e expiradas, assim como fazer a contagem das estatísticas de fluxo de recepção. Seguidamente, as regras de ajuste são responsáveis por decompor as tramas no seu destino pretendido, ajustar os campos da trama e colocar as tramas na fila de trânsito correcta. Por fim, as regras de filtragem são responsáveis por decidir se as tramas são copiadas para o cliente, ou para a sub-camada de controlo do MAC, ou se são apagadas do MAC.

As interfaces de serviço do MAC fornecem as primitivas de serviço, usadas pelos clientes do MAC para trocar dados com um ou mais clientes no anel, ou para transferir informação de controlo local entre o cliente do MAC e o MAC. A sub-camada de controlo do MAC, controla a sub-camada de *datapath*, mantém o estado do MAC e a coordenação com a sub-camada de controlo do MAC de outros MACs, e também controla a transferência de dados entre o MAC e o seu cliente. A sub-camada de *datapath* do MAC fornece funções de transferência de dados para cada *ringlet*.

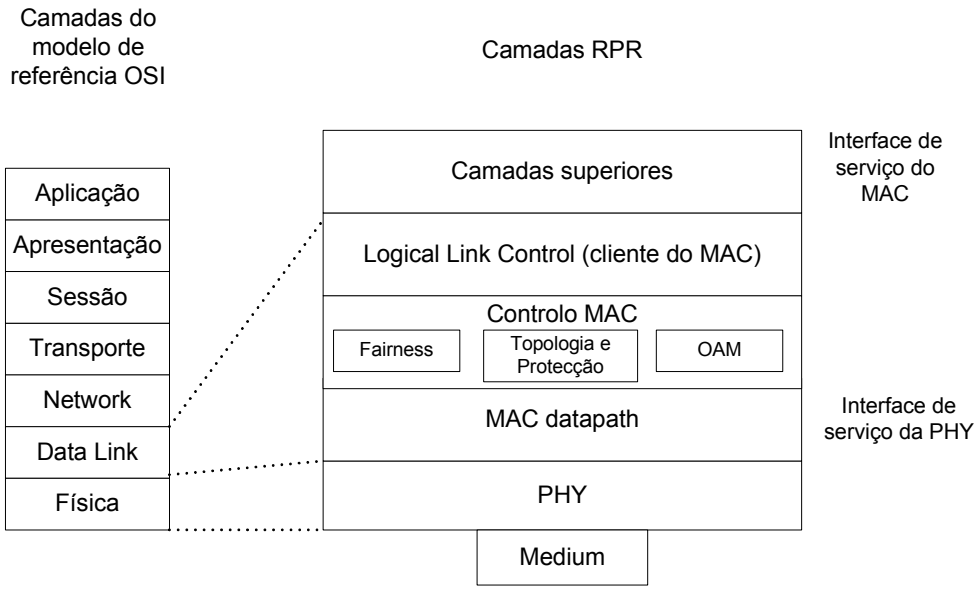


Figura 4.14 – Relação entre os serviços do RPR com o modelo de referência OSI do ISO/IEC

A interface de serviço da PHY é usada pelo MAC para transmitir e receber tramas nos meios físicos. No IEEE 802.17, há distintas sub-camadas de reconciliação e, opcionalmente, interfaces independentes do meio (MIIs) que especificam o mapeamento entre PHYs específicos.

4.6 Serviços do MAC

Os serviços fornecidos pela sub-camada do MAC permitem:

a) À camada local do cliente numa estação de extremidade, trocar dados com outras entidades da camada cliente;

b) À camada local do cliente numa estação de extremidade, trocar parâmetros com a entidade local do MAC;

c) À entidade de *relay* numa *bridge*, trocar dados com as entidades locais do MAC na *bridge*.

O MAC do RPR fornece dois tipos de serviço de transmissão da trama:

a) Estrito – Adere ao ordenamento, duplicação e exigências de perda de trama do 802.1, isto é:

1. Não há garantia de que todas as SDUs sejam entregues;
2. Não é permitido o reordenamento das tramas do utilizador dentro de uma dada prioridade, para uma dada combinação de endereços de destino e de origem;
3. Não é permitida a duplicação de tramas de dados do utilizador.

Geralmente, a transmissão estrita requer processamento especial somente durante eventos de falha (da estação ou da ligação).

b) Relaxado – Adere ao ordenamento, duplicação e exigências de perda de trama do 802.1, excepto ao recuperar de falhas no anel. Durante eventos de protecção do anel, pode haver uma quantidade mínima de reordenamento e/ou de duplicação, porém a possibilidade de entrega é aumentada. Cenários onde pode ocorrer uma quantidade insignificante de reordenamentos ou duplicação de tramas, incluem os seguintes:

1. Após eventos de restauração do anel (ligação ou estação);
2. A base de dados da topologia e do estado das estações no anel não está sincronizada;
3. Falha da estação tendo por resultado o comportamento de *passthrough*. Isto é, as tramas são emitidas através do trajecto de trânsito sem decréscimo do *ttl* ou não são aplicadas quaisquer outras regras de processamento/decomposição de tramas;
4. Falhas compostas do anel (ligação ou estação), tendo por resultado anéis abertos segmentados;
5. Falhas rápidas em cascata, no anel.

4.6.1 Classes de serviço

A sub-camada do MAC apresenta uma interface de serviço para a troca de SDUs do MAC entre entidades clientes do MAC. O serviço de interface suporta classes de serviço do MAC classificadas em classe A, classe B e classe C.

O serviço de Classe-A fornece: uma taxa de dados reservada e garantida; baixo atraso extremo-a-extremo; e *jitter* limitado por uma ordem de grandeza de (*numStations*mtuSize* – número máximo de estações * tamanho máximo da trama). O tráfego de Classe-A tem precedência sobre o tráfego de Classe-B e de Classe-C, no ingresso ao anel e durante o trânsito através do mesmo (para estações de fila-dupla). Este tráfego não é sujeito ao algoritmo de *fairness* no ingresso ao anel ou ao transitar através do mesmo. Consequentemente, neste tipo de tráfego, o bit *fe* dentro do byte *baseControl* do cabeçalho da trama RPR é sempre ajustado a 0.

Internamente ao MAC, o tráfego de Classe-A é dividido em duas subclasses: subclasse-A0 e subclasse-A1. Esta divisão é feita com a finalidade de aumentar a capacidade do anel em reclamar tráfego de Classe-A não utilizado. O cliente do MAC requisita tráfego de Classe-A e não uma das subclasses internas, pois não lhe são visíveis. O MAC é configurado para uma quantidade total de Classe-A, da qual determina quanto é subclasse-A0 e quanto é subclasse-A1. A divisão da Classe-A é baseada na extensão da circunferência do anel e no tamanho da STQ dessa estação. As implementações de fila-única reservam sempre 100% do tráfego de Classe-A para subclasse-A0, e 0% para subclasse-A1. O MAC anuncia, através da trama ATD da estação, uma largura de banda reservada igual à quantidade de subclasse-A0 interna. A largura de banda reservada à subclasse-A1 pode facilmente ser recuperada pelo tráfego de Classe-B-EIR e de Classe-C ao não ser usada pela estação que origina o tráfego de Classe-A que está a ser recuperado.

A quantidade de tráfego de Classe-A de uma estação, que pode ser emitida como subclasse-A1, deve ser determinada tendo em consideração quanto tráfego em trânsito de Classe-B e de Classe-C pode ser enfileirado pela estação local enquanto está a sinalizar para estações a montante para diminuírem o seu tráfego em excesso.

Baseado no tamanho da implementação de uma STQ, a quantidade por defeito de tráfego adicionado de subclasse-A1 e Classe-B-CIR que pode ser suportada, pode ser estimada por:

$$addRateA1 \leq ((sizeSTQ - stqHighThreshold)/responseTime) - addRateB \qquad \text{(Eq. 4.1)}$$

Subtraindo o *addRateA1* da reserva total para Classe-A, resulta na quantidade de tráfego de Classe-A reservado como *addRateA0* e que o MAC anuncia para cada *ringlet* através de ATT_STATION_BW. As propagandas de subclasse-A0 são usadas para determinar o *reservedRate* num *ringlet*. Subtraindo o *reservedRate* do LINK_RATE resulta no *unreservedRate.*

O serviço de Classe-B fornece: uma taxa de dados reservada e garantida; e um atraso extremo-a-extremo e *jitter* limitados para o tráfego dentro da taxa reservada, isto é, limitados na ordem de grandeza de um RRTT. Dentro desta classe, o MAC usa marcas de elegibilidade para *fairness,* para diferenciar a parcela da taxa de informação cometida de Classe-B (Classe-B-CIR) e a parcela da taxa da informação em excesso de Classe-B (Classe-B-EIR). Fornece acesso à transmissão adicional de dados *best-effort* que não é reservada, garantida, ou limitada, e é sujeita ao algoritmo de *fairness*. O tráfego de Classe-B (incluindo Classe-B-EIR) tem precedência sobre o tráfego de Classe-C no ingresso ao anel.

Este serviço tem similaridades com o serviço de Classe-A, descrito acima, em que as taxas de transmissão da trama dentro do perfil de taxa reservada (conhecido como Classe-B-CIR), tem limites de atraso e de *jitter* garantidos, embora com limites menos apertados do que para tramas de Classe-A. O tráfego dentro do perfil de taxa reservada não é sujeito ao algoritmo de *fairness* no ingresso ao anel ou ao transitar através das suas estações.

O tráfego desta classe tem também similaridades com o tráfego de serviço Classe-C, descrito abaixo, em que o tráfego reservado para além do perfil da taxa (Classe-B-EIR) é sujeito ao algoritmo de *fairness*, e é marcado pelo MAC no bit *fe* do cabeçalho do RPR antes da transmissão no anel. As tramas elegíveis para *fairness* são contadas no algoritmo de *fairness* do RPR tanto no ingresso ao anel, como ao transitar pelas estações do mesmo.

Internamente ao MAC, o tráfego de Classe-B é dividido em Classe-B-CIR e em Classe-B-EIR no ingresso ao anel com o uso do bit *fe*. O cliente pode deixar o MAC escolher baseado na presença ou ausência de *sendB*, ou pode forçar uma trama de Classe-B a ser considerada somente para Classe-B-EIR ajustando o parâmetro de *mark_fe*. O tráfego de Classe-B-EIR recebe uma maior qualidade de serviço do que o de Classe-C porque todo o tráfego de Classe-B, incluindo o tráfego de Classe-B-EIR, recebe precedência no ingresso sobre o tráfego de Classe-C.

Dado que a Classe-B-EIR tem uma precedência mais elevada do que a Classe-C, aquela pode estrangular esta. No entanto, como a submissão de tráfego Classe-B-EIR e de Classe-C é feita pelo cliente, este tem o controlo sobre as quantidades relativas destas classes de tráfego.

Numa implementação de fila-única, todo o tráfego de Classe-B move-se através da PTQ. Numa implementação de fila-dupla, esta classe de tráfego move-se através da STQ, não obstante a trama esteja marcada elegível para *fairness* ou não.

O serviço de Classe-C fornece um serviço de tráfego de melhor esforço com nenhuma taxa de dados reservada ou garantida, e nenhum limite de atraso extremo-a-extremo ou *jitter*.

Este tipo de tráfego tem a precedência mais baixa no ingresso ao anel. É sempre sujeito ao algoritmo de *fairness*, e é marcado pelo MAC no bit *fe* do cabeçalho RPR, antes da transmissão no anel. As tramas de Classe-C são contadas no algoritmo de *fairness* do RPR tanto no ingresso ao anel como ao transitar pelas estações no mesmo.

Numa implementação de fila-única, este tráfego move-se através da PTQ, e numa implementação de fila-dupla, move-se através da STQ.

Para todas as classes de serviço, a interface de serviço do MAC fornece indicações de cada *ringlet* ao cliente do MAC que indicam se o tráfego pode ou não ser aceite. Para o serviço de Classe-C, a interface de serviço do MAC fornece também o número de extensões atravessadas (de *hops*) até à estação congestionada mais próxima.

A taxa de tráfego de cada classe de serviço é controlada. O MAC força as restrições da largura de banda, recusando permitir que o seu cliente transmita mais tráfego do que a taxa reservada pela gestão da estação ou permitida pelo *fairness*, ou se não houver qualquer limite de taxa reservada, até à largura de banda total do anel.

Para serviços reservados, é emitida do MAC ao cliente, uma indicação de estado que indica se o cliente tem ou não permissão para transferir dados. Para o serviço oportunístico, a distância da estação local ao destino permitido mais distante, se existir, é também fornecida. A distância é representada como o número das extensões atravessadas, conhecido também como *hop-count*. Esta informação permite que o cliente enfileire o tráfego distintamente para cada destino, ou quando um cliente não obedece à indicação do MAC no que diz respeito ao controlo de fluxo numa dada classe de serviço, o MAC deve assegurar a operação correcta do anel recusando os pedidos "ofensivos" do cliente, ou bloqueando o topo da linha (exactamente na interface de entrada dos dados do cliente para o MAC) até que o pedido possa ser correctamente assegurado, evitando assim obstrução nesse local.

<table>
<tr><th colspan="3">Classe de Serviço</th><th colspan="4">Qualidade de Serviço</th></tr>
<tr><td>Tipo</td><td>Uso</td><td>Subclasse</td><td>Largura de banda garantida</td><td>Atraso/Jitter</td><td>Tipo de largura de banda</td><td>Subtipo de largura de banda</td></tr>
<tr><td rowspan="2">Classe A</td><td rowspan="2">Tempo real</td><td>Subclasse A0</td><td rowspan="2">Sim</td><td rowspan="2">baixo</td><td rowspan="2">Atribuída (reservada)</td><td>reservada</td></tr>
<tr><td>Subclasse A1</td><td rowspan="3">reclamada</td></tr>
<tr><td rowspan="2">Classe B</td><td rowspan="2">Perto de tempo real</td><td>Classe B –CIR</td><td>limitado</td><td>Atribuída (reservada)</td></tr>
<tr><td>Classe B –EIR</td><td>Não</td><td>Não limitado</td><td>oportunística</td></tr>
<tr><td>Classe C</td><td>Best-effort</td><td>-</td><td colspan="3"></td></tr>
</table>

Tabela 4.1 – Classes de serviço e as suas relações de qualidade de serviço

A capacidade do anel requerida para suportar o serviço de Classe-A e o serviço de Classe-B-CIR, é reservada através de aprovisionamento, e estes serviços podem ser caracterizados como serviços reservados. A actividade de aprovisionamento deve garantir que o compromisso do serviço agregado em cada ligação não exceda a capacidade dessa ligação. As taxas de reserva distribuídas pelo aprovisionamento regulam o acesso a estes serviços garantidos.

É de referir que o MAC não aprovisiona largura de banda. Esta é uma função de gestão a um nível mais elevado. A capacidade do anel tem que ser assegurada para suportar as garantias de serviço das Classe-A e Classe-B-CIR.

O tráfego elegível para *fairness* (Classe-B-EIR e Classe-C) é oportunístico em vez de reservado, pois usa a largura de banda disponível da largura de banda não reservada e da largura de banda reclamável não utilizada, como descrito abaixo. É usado um algoritmo de *fairness* ponderado para dividir a largura de banda elegível para *fairness* entre estações concorrentes.

4.6.1.1 Reclamação da largura de banda

A largura de banda reservada pode ser reutilizada ou reclamada, por um serviço de classe mais baixa sempre que a reclamação não afecte as garantias de serviço de qualquer classe(s) de prioridade igual ou superior à da estação local ou à de qualquer outra estação no anel.

A reclamação da largura de banda não utilizada, de todo o tráfego excepto o tráfego de subclasse-A0, pode ser feita por qualquer estação que determine através do algoritmo de *fairness* que tem permissão para adicionar tráfego elegível para *fairness*.

O algoritmo de *fairness* e os *shapers* do *datapath* garantem que o tráfego elegível para *fairness* possa ser adicionado numa determinada quantidade com qualquer classe de serviço igual ou superior que não viole as garantias de serviço dessa classe.

Da mesma forma, o tráfego adicional pode ser reclamado, para além daquele reclamado pelo algoritmo de *fairness*, quando uma estação o puder fazer sem entrar em conflito com as garantias de serviço das classes de serviço cujo tráfego está a ser reclamado.

4.6.2 Primitivas de controlo de fluxo do MAC

As primitivas de controlo do MAC são usadas pelo cliente do MAC para requerer e para receber informação ou acção de controlo do MAC local. Exemplos de informação de controlo incluem a configuração da estação, o estado de congestionamento, a emissão do estado e a topologia do anel.

Um dos usos das primitivas de controlo do MAC é o de fornecer indicações de controlo de fluxo para regular o acesso do cliente ao anel. Todo o tráfego que entra num *ringlet* originado de um cliente é sujeito ao controlo da taxa por *shaping*. No exemplo de serviços reservados, os parâmetros de *shaping* são mudados somente quando as reservas mudam. No exemplo de serviços oportunísticos, os *shapings* dos parâmetros são dinâmicos e são ajustados quando os valores da taxa são recalculados pelo algoritmo de *fairness*.

4.6.3 Serviços do MAC para a camada cliente

O serviço do MAC é definido pela semântica do modelo de interfaces ilustrado na figura 4.15. Um pedido está associado ao sentido do cliente do MAC para o MAC, uma indicação está associada ao sentido do MAC para o cliente do MAC. As primitivas de controlo são usadas pelo cliente local do MAC para solicitar e para receber a informação de controlo das primitivas locais do MAC. São usadas primitivas de dados pelo cliente local do MAC, para trocar unidades de dados do protocolo da camada cliente (PDUs) com os clientes remotos do MAC.

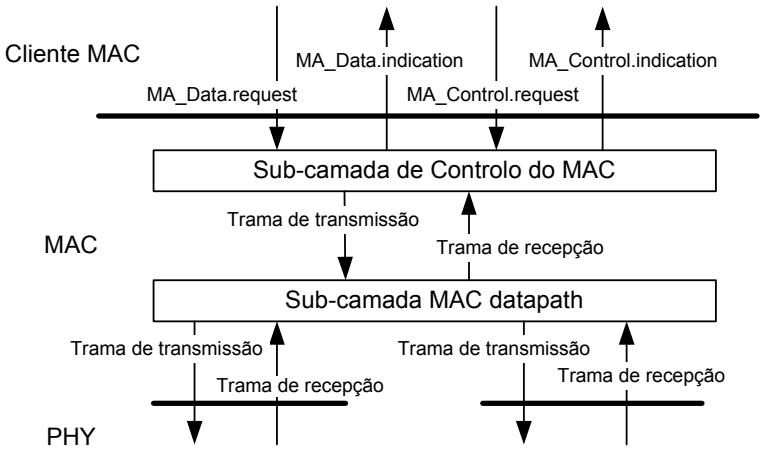


Figura 4.15 — Modelo de serviço do MAC

A primitiva MA_DATA.request define a transferência de dados de uma entidade cliente do MAC a uma única entidade par, ou a múltiplas entidades pares no caso de endereços de grupo.

A primitiva MA_DATA.request é invocada pela entidade cliente sempre que os dados devam ser transferidos a uma entidade ou a entidades pares. A recepção desta primitiva faz com que a entidade MAC crie uma trama de dados ou uma trama de dados estendida; preencha os campos cujos valores são dados ou determinados pelos parâmetros deste pedido; e passe a trama correctamente formada à unidade de transmissão, para transferência à entidade ou às entidades pares da sub-camada do MAC.

O MAC não reflecte tramas de volta ao cliente. Se um cliente emitir uma primitiva MA_DATA.request com um valor de destination_address igual ao seu endereço MAC local, o pedido será rejeitado.

O MAC também não aceita pedidos do cliente quando a indicação apropriada de *send* não estiver presente.

A primitiva MA_DATA.indication define a transferência de dados da entidade da sub-camada do MAC à entidade cliente do MAC.

O MA_DATA.indication é passado desde a entidade da sub-camada do MAC (através da sub-camada de controlo do MAC) à entidade ou às entidades clientes do MAC, para indicar a chegada de uma trama à entidade local da sub-camada do MAC que é destinada ao cliente do MAC. Tais tramas são relatadas apenas se são validamente formadas e o seu endereço de destino designa a entidade local do MAC (endereço de estação local, endereço de grupo ou *flooded*).

A primitiva MA_CONTROL.request define a transferência de pedidos de controlo do cliente do MAC à sub-camada de controlo do MAC. Não fornece meios directos para um cliente transmitir uma trama de controlo do MAC local em qualquer *ringlet*; embora as tramas de controlo (por exemplo eco ou *flush*) possam indirectamente ser geradas em consequência deste pedido.

Se uma entidade ao nível do cliente requerer o ajuste ou a informação de *status,* ou a configuração de parâmetros de *status* do MAC, direcciona ou questiona o MLME para ajustar ou obter esta informação. Exemplos incluem as bases de dados da topologia e do *status*, comutação manual ou forçada de fila-única versus fila-dupla

Esta primitiva define também a transferência de comandos de controlo de uma entidade cliente do MAC à entidade local da sub-camada de controlo do MAC.

Esta primitiva é gerada por um cliente do MAC sempre que deseja usar os serviços da entidade da sub-camada de controlo do MAC.

A primitiva MA_CONTROL.indication define a transferência de indicações de *status* de controlo da sub-camada de controlo do MAC ao cliente do MAC.

Esta primitiva é gerada pela sub-camada de controlo do MAC sob circunstâncias específicas a cada operação de controlo do MAC.

O uso pelo cliente das indicações fornecidas pelo MA_CONTROL.indication é disponibilizado para permitir que um cliente execute acções mais complexas para além da potencialidade do MAC, por exemplo, implementando um algoritmo mais eficiente de programação de uma trama, baseado no conhecimento dos pontos de bloqueio relatados através do MULTI_CHOKE_IND.

As indicações e os operandos das indicações não são necessariamente passados fisicamente ao cliente. Podem meramente ser disponibilizados ao cliente. Os meios de indicação ou de disponibilização são um detalhe da implementação local.

MA_DATA.request [destination_address, source_address (opcional), mac_service_data_unit, frame_check_sequence (opcional), service_class, ringlet_id (opcional), mac_protection (opcional), mark_fe (opcional), strict_order (opcional), destination_address_extended (opcional), source_address_extended (opcional), flooding_form (opcional)]

- O destination_address especifica um indivíduo ou um endereço MAC de grupo, diferente do endereço MAC local, a ser usado para criar o campo *da* (endereço MAC de destino) da trama

transmitida. O source_address, se presente, e diferente do endereço MAC da estação, especifica um endereço MAC individual, a ser usado para criar o campo de *saExtended* (endereço MAC de origem, estendido) da trama transmitida e para determinar o formato da trama de dados básica ou estendida.

- O mac_service_data_unit fornece o *payload* a ser entregue. Especifica a unidade de dados de serviço do MAC a ser transmitida pela entidade da sub-camada do MAC no campo de serviceDataUnit da trama transmitida.
- O frame_check_sequence fornece o valor do campo do *fcs* da trama transmitida.
- O service_class indica a classe de serviço requisitada pelo cliente do MAC, e é usado pela entidade do MAC para seleccionar o valor do campo *sc* (*service class*) e para indicar o tratamento requerido pelo MAC para a trama transmitida.
- O ringlet_id indica a escolha do *ringlet* do cliente, e é usado pela entidade MAC para seleccionar o valor do campo *ri* (*ringlet identifier*) da trama transmitida, isto antes de alguma mudança baseada na protecção.
- O mac_protection indica a escolha de se o MAC fornece protecção para a trama.
- O mark_fe indica um pedido para marcar e tratar uma trama como elegível para *fairness*, não obstante tenha sido marcada ou tratada de outra maneira, guiando a entidade MAC em como ajustar o campo *fe* (*fairness eligible*). Este parâmetro é válido apenas para os pedidos que incluem um pedido de Classe-B para o service_class, e é ignorado para todos os pedidos restantes.
- O strict_order indica um pedido para marcar e tratar uma trama de dados como estrita ou relaxada, o que é usado pela entidade MAC para seleccionar o valor *so* (*strict order*). O parâmetro de strict_order é ignorado e todas as tramas têm o campo *so* ajustado a 1, se a variável de *forceStrict* for ajustada a TRUE.
- O destination_address_extended especifica um indivíduo ou o endereço MAC de grupo, diferente do endereço de MAC local, a ser usado para criar *o* campo de *daExtended* da trama transmitida. Se o parâmetro destination_address_extended é fornecido, o MAC usa o formato estendido da trama de dados. Sempre que o parâmetro destination_address_extended é fornecido, o parâmetro source_address_extended é também requerido para ser fornecido.
- O source_address_extended especifica um endereço MAC individual, a ser usado para criar o campo de *saExtended* da trama transmitida. Se o parâmetro source_address_extended é fornecido, o MAC usa o formato estendido da trama de dados. Sempre que o parâmetro source_address_extended é fornecido, o parâmetro destination_address_extended é requerido também para ser fornecido.
- O flooding_form indica um pedido para usar um *flooding form* particular, e ajustar o campo *fi* (*flooding indication*) de acordo. Um valor omitido ou um valor de FI_NONE dirigem o MAC a não fazer o *flooding* da trama, a menos que a selecção do *ringlet* determinar que o necessita. Um valor à excepção de FI_NONE dirige o MAC para fazer o *flooding* da trama não obstante a determinação que seria feita de outra maneira pela selecção do *ringlet*, e para usar o *flooding form* seleccionado para a trama.

MA_DATA.indication (destination_address, source_address, mac_service_data_unit, frame_check_sequence, reception_status, service_class, ringlet_id, fairness_eligible, strict_order, extended_frame, destination_address_extended, source_address_extended)

- O destination_address indica o valor do campo *da* da trama que entra. O source_address indica o valor do campo *sa* da trama que entra.
- O mac_service_data_unit fornece a unidade de dados de serviço do MAC, como especificado pelo campo de *serviceDataUnit* da trama que entra.
- O frame_check_sequence indica o valor do campo *fcs* no cabeçalho da trama.

- O reception_status indica o *status* da trama recebida à entidade cliente do MAC. Este campo pode ter o valor de RECEIVE_FCS_ERROR somente se a variável *copyBadFcs* é ajustada a TRUE.
- O service_class indica a classe de serviço com a qual a trama foi emitida.
- O ringlet_id indica o ajuste do bit *ri* no cabeçalho da trama.
- O fairness_eligible indica o ajuste do bit *fe* no cabeçalho da trama.
- O strict_order indica o ajuste do bit *so* no cabeçalho da trama.
- O extended_frame indica o ajuste do bit *ef* no cabeçalho da trama.

No destination_address_extended, se o extended_frame é TRUE indica o valor do campo daExtended da trama que entra.

No source_address_extended, se o extended_frame é TRUE indica o valor do campo saExtended da trama que entra.

O MAC não reflecte tramas de volta ao cliente. Se um MAC receber uma trama com o valor do *sa* igual ao endereço do MAC local, não faz com que uma primitiva de MA_DATA.indication seja emitida ao cliente de origem.

MA_CONTROL.request (opcode, request_operand_list)

O parâmetro *opcode* indica a operação de controlo requerida pela entidade cliente do MAC.

opcode	**significado**	**operandos**
OAM_ECHO_REQ	Pedido para transmitir uma trama de pedido de eco	*Payload* e parâmetros de pedido de eco
OAM_FLUSH_REQ	Pedido para transmitir uma trama de *flush*	*Payload* e parâmetros de *flush*
OAM_ORG_REQ	Pedido para transmitir uma trama OAM especifica de uma organização	*Payload* e parâmetros específicos de uma organização

Tabela 4.2 — *Opcodes* de *control request*

O efeito da recepção desta primitiva pela sub-camada de controlo do MAC é específico do *opcode*.

MA_CONTROL.indication (opcode, indication_operand_list)

Os elementos do parâmetro indication_operand_list são específicos a cada parâmetro do *opcode*.

4.6.4 Modelo de referência do MAC

Na figura 4.16 está ilustrado o modelo de referência do MAC, suas funções e *datapaths* internos do mesmo.

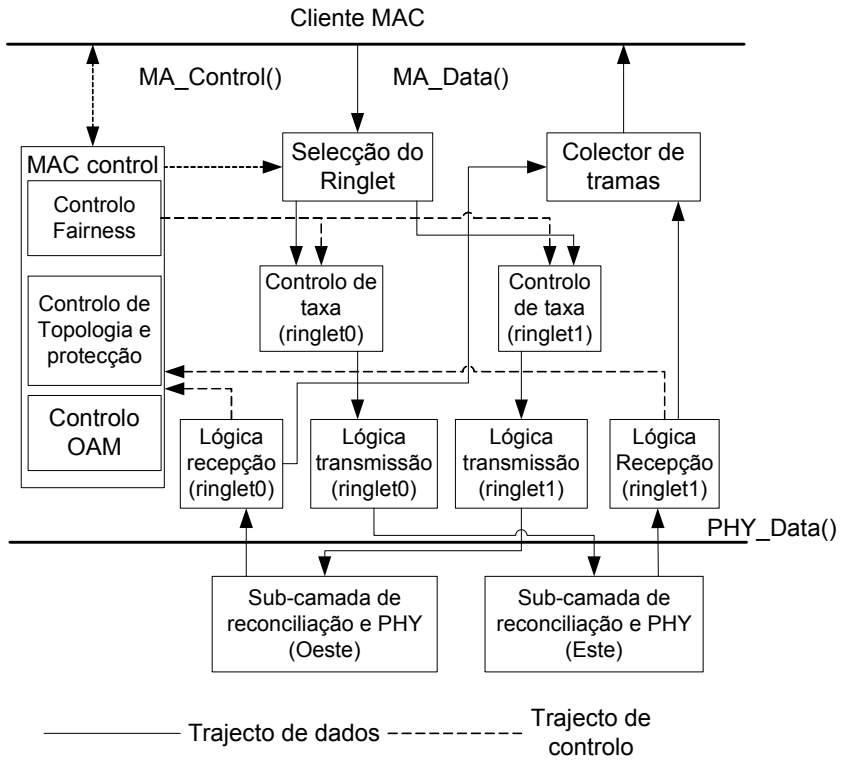


Figura 4.16 – Modelo de referência do MAC, funções e datapaths internos do MAC

A sub-camada de reconciliação faz parte da camada física, e fornece uma interface de serviço uniforme à camada do MAC. Há uma entidade de sub-camada de reconciliação para cada interface da camada física.

A sub-camada de controlo de acesso ao meio fornece o controlo de acesso para o meio da camada física. Controla também o(s) trajecto(s) de trânsito através do MAC. As suas funções incluem recepção de tramas, transmissões de tramas, selecção do *ringlet*, colocação das tramas na fila de trânsito correcta, controlo da taxa, *fairness*, protecção e descoberta da topologia.

A função de recepção é de receber tramas da sub-camada de reconciliação e copiá-las para um ou mais clientes do MAC, para a sub-camada de controlo do MAC, e desde a fila de trânsito designada (PTQ ou STQ), como apropriado para cada trama.

A função de transmissão é de transmitir tramas à sub-camada de reconciliação da(s) fila(s) de trânsito (PTQ ou STQ), à sub-camada de controlo do MAC e ao cliente do MAC.

A selecção do *Ringlet* pode ser especificada inteiramente pelo cliente, especificada pelo cliente mas com a opção do MAC poder cancelá-la inteiramente para protecção, ou totalmente responsável ao MAC.

A função de controlo da taxa é de gerir a taxa à qual as tramas são transmitidas desde o cliente e desde as filas de trânsito, e coordenar esse controlo com as outras sub-camadas MAC no anel. Controla o acesso pelo cliente do MAC a cada serviço com a finalidade de assegurar que as regras para o acesso ao meio e para a reserva de largura de banda são obedecidas. O tráfego adicionado é *shaped* usando *token buckets* por classe de serviço. Estes *shapers* são usados para indicar ao cliente do MAC para cessar de fazer MA_DATA.requests para uma classe de serviço em particular. As funções dos *shapers* são as indicações resultantes de *sendA*, *sendB* e *sendC*. O controlo dinâmico da largura de banda para acomodar o tráfego *bursty* é implementado através do mecanismo de controlo da largura de banda do algoritmo de *fairness*. Este mecanismo fornece resultados ao *shaper fairness eligible* para permitir o ajuste correcto dos valores para a indicação de *sendC*. Além disso, o mecanismo opcional de *multi-choke* permite que o cliente forneça

unidades de dados de serviço ao MAC a uma taxa óptima mesmo quando mais de uma ligação está congestionada.

A gestão da largura de banda é feita para manter a justiça para as tramas *fairness eligible*, usando mecanismos para assegurar que todas as estações recebem a sua parte justa da capacidade do anel através das ligações que estão a ser usadas pelas estações, em que a parte justa não é necessariamente a mesma para todas as estações. O algoritmo de *fairness* assegura dinamicamente a distribuição ponderada de larguras de banda de ligações disponíveis, às estações de origem usando aquelas ligações.

Por defeito, o tráfego é protegido de falhas no anel e no equipamento do anel. O cliente do MAC pode escolher não proteger uma dada trama. A função da protecção fornece a máquina de estados de protecção e controla a base de dados de protecção para o MAC local, e a coordenação deste controlo com as outras sub-camadas MAC no anel.

A função da topologia da sub-camada do MAC é de controlar a base de dados da topologia. Fornece também a máquina de estados da topologia para o MAC local, e a coordenação deste controlo com as outras sub-camadas de controlo MAC no anel. Uma rede RPR consiste em *ringlets* duplos contra-rotacionais. O MAC pode apresentar duas vistas da rede ao cliente do MAC: uma vista "plana" da rede, em que a sub-camada do MAC esconde ao cliente a topologia baseada em anel duplo; ou uma vista ciente da topologia que permita que o cliente do MAC faça pedidos de dados e controlo para *ringlets* específicos. A informação topológica é colectada através de um processo da entidade da sub-camada do MAC, conhecido como "descoberta da topologia", e disponibilizada ao cliente através de um MA_CONTROL.indication.

A operação, administração e manutenção (OAM) fornece um conjunto de tramas e de indicações de controlo para suportar a configuração, a localização de falha e a manutenção do anel.

A entidade da gestão da camada do MAC (MLME) é uma entidade independente que reside fora da camada do MAC, num plano separado de gestão. O MLME contém o MIB para a camada do MAC, e fornece operações de *get* e *set* no MIB às entidades do utilizador do MLME. O MLME lê de e escreve para a camada MAC em consequência da invocação de primitivas do MLME.

4.6.4.1 Sub-camada de controlo do MAC

A sub-camada de controlo do MAC suporta as actividades necessárias para manter o estado do MAC e do *datapath,* identificado com um *ringlet* em particular (fig. 4.17). As actividades de controlo são distribuídas entre estações no anel com a finalidade de sobreviver a qualquer ponto único de falha. As entidades de controlo numa estação comunicam com as entidades pares de controlo noutras estações, usando os serviços da sub-camada do *datapath* do MAC. As actividades da sub-camada de controlo do MAC incluem o seguinte:

a) Interface de serviço de controlo;
b) Algoritmo e protocolo de *fairness*;
c) Base de dados do protocolo de protecção;
d) Base de dados do protocolo da topologia;
e) Funcionalidades da operação, administração e manutenção (OAM);
f) Emissão e recepção da trama de controlo.

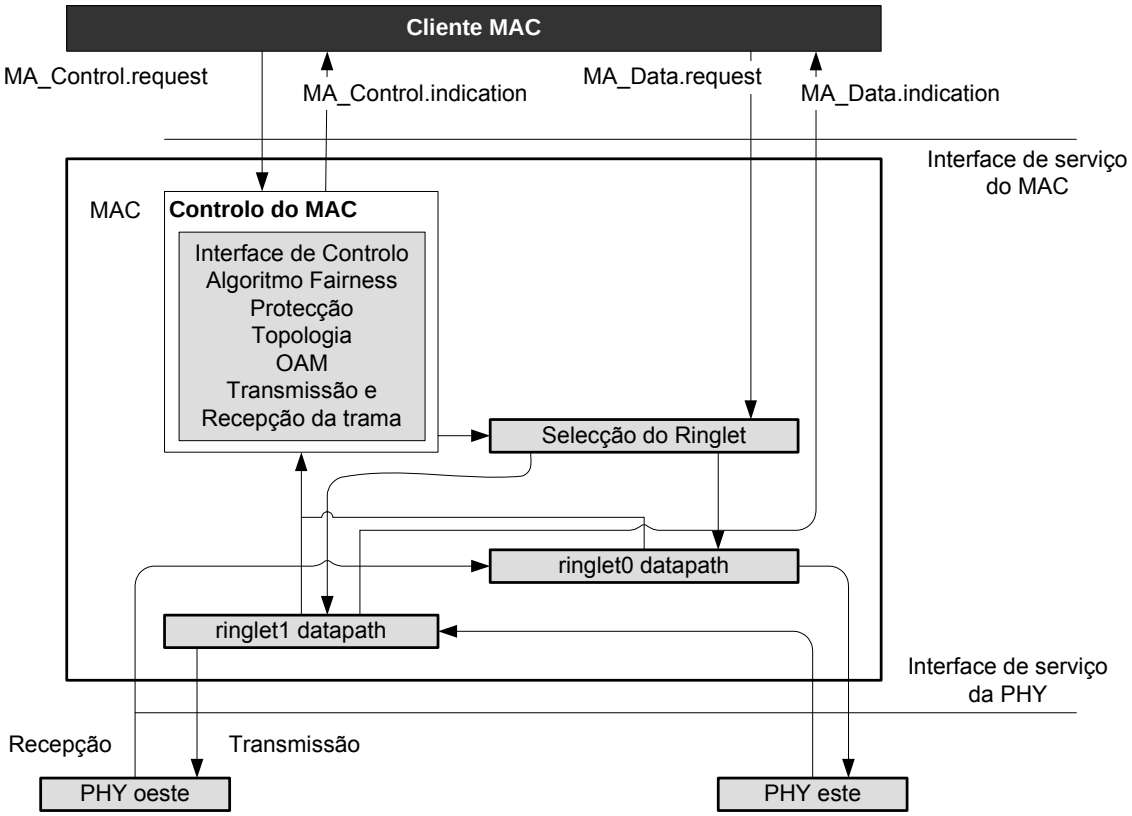


Figura 4.17 — Arquitectura da unidade de controlo do MAC

4.6.4.2 Sub-camada de *datapath* do MAC

A sub-camada de *datapath* do MAC inclui um único componente de selecção do *ringlet* e dois módulos distintos de *datapath* específicos do *ringlet* (fig. 4.18).

A unidade de selecção do *ringlet* usa os destination_address, source_address e valores da base de dados de topologia e protecção para determinar o seguinte:

a) O ringlet a usar para transmitir a trama;

b) Se e como fazer o *flood* da trama;

c) Se usar o formato de trama básico ou estendido.

Os componentes específicos do *datapath* do *ringlet* fornecem as seguintes funções:

a) O encapsulamento e o desencapsulamento de tramas de dados do cliente (incluindo inserção e extracção de campos do cabeçalho e do *trailer*) na transmissão e recepção;

b) *Shaping* do tráfego por classe de serviço para regular o acesso ao meio partilhado do anel;

c) *Staging* e *queueing* das tramas na sua origem e enfileiramento de tramas em trânsito;

d) Cópia e *routing* de tramas recebidas para o cliente do MAC e para a sub-camada de controlo do MAC;

e) Decomposição (*strip*) do anel de tramas erradas ou expiradas;

f) Transmissão e recepção de tramas.

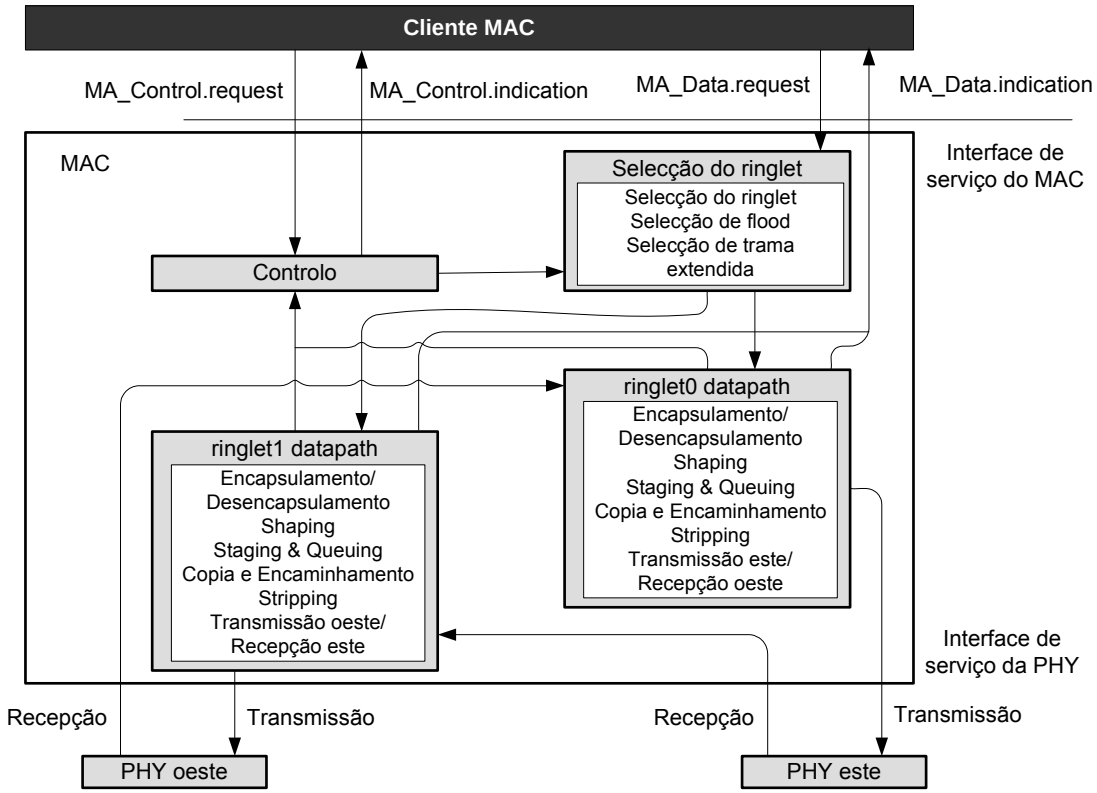


Figura 4.18 – Arquitectura do *datapath* do MAC

4.6.4.3 Fluxo de dados dentro do MAC

As estações podem suportar *steering*, ou esquemas de protecção *central-wrapping* ou *edge-wrapping* (ver 5.2.4).

Para trajectos protegidos tipo *steering*, a protecção é fornecida na origem, não na estação fronteira, como no caso da protecção tipo *wrapping*. Esta protecção é fornecida dirigindo as tramas para fora de todas as estações fronteiras que possam encontrar. A decisão de fazer o *steering* é tomada como parte da selecção do *ringlet*.

Durante o trânsito das tramas não é feito qualquer ajuste específico de *steering* às mesmas. O bloco de controlo de recepção da fronteira de um trajecto *steering*, decompõe as tramas que entram somente quando o restante trajecto não está a ser usado.

O *steering* não tem qualquer efeito na transmissão local de tramas de topologia e protecção e de tramas de *fairness*, que continuam a ser emitidas para fora de ambos os PHYs mesmo quando um PHY está a transmitir para dentro de uma fronteira.

As estações que usam sistemas de protecção tipo *wrapping*, contêm caminhos de dados circundados, permitindo que as tramas retornem pelo *ringlet* oposto após ser detectada uma falha na ligação.

Para sistemas *central-wrapping*, o *datapath unwrapped* e o *datapath wrapped* não estão activos ao mesmo tempo. Durante condições normais, o *datapath unwrapped* é usado e o datapath *wrapped* é desactivado. Durante condições de falha locais, o datapath *wrapped* é usado e o datapath *unwrapped* é desactivado.

Uma estação *center wrapping* é uma estação que implementa *wrapping* no centro do trajecto. Para configurações *center wrap* as tramas são ligadas desde a cabeça de um *ringlet* para dentro da fronteira da sub-camada de reconciliação do PHY do *ringlet* oposto. Isto tem o efeito de isolar o trajecto do *ringlet* oposto em relação ao cliente.

Para estações circundadas *center-wrap*, o tráfego adicionado é redireccionado através da selecção do *ringlet*, como é feito para um nó de direccionamento. O bloco de controlo de recepção da fronteira de um trajecto *center-wrap,* decompõe as tramas que entram, somente quando o restante trajecto não está a ser usado.

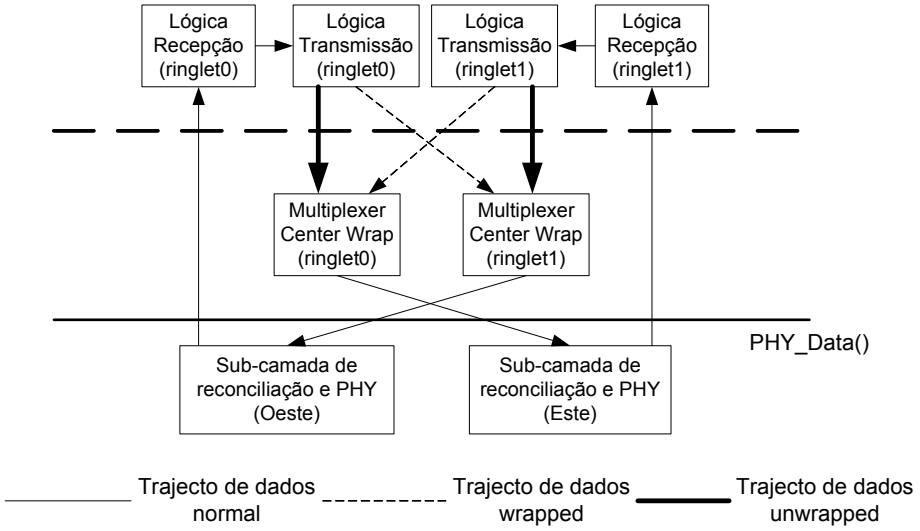


Figura 4.19 – Trajectos *center wrap*

O *center wrapping* não tem qualquer efeito na transmissão local de tramas de topologia e protecção e de tramas de *fairness*, que continuam a ser emitidas para fora de ambos os PHYs mesmo quando um PHY está a transmitir para dentro de uma fronteira.

Para sistemas *edge wrapping*, o *datapath unwrapped* e o *datapath wrapped* estão por vezes activos ao mesmo tempo. Durante condições normais, o *datapath wrapped* é usado e o *datapath unwrapped* é desactivado. Durante condições de falha locais, ambos os *datapaths wrapped* e *unwrapped* estão activos.

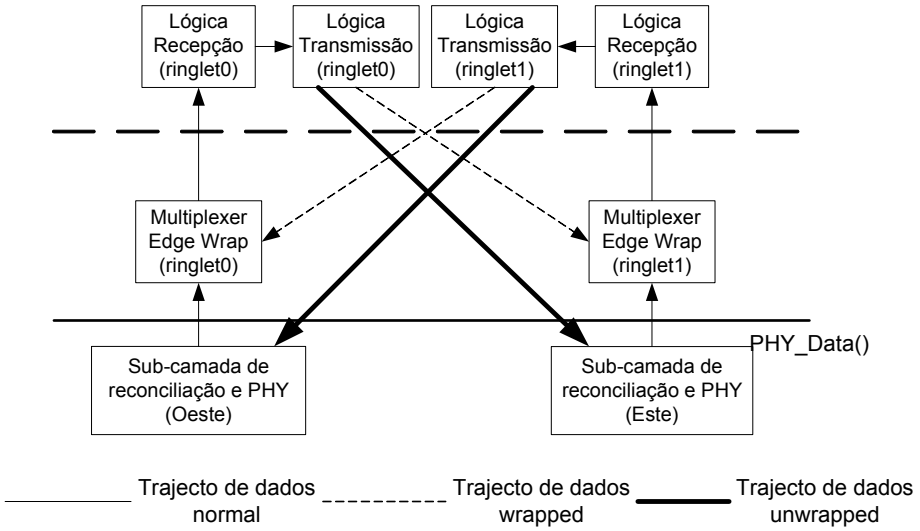


Figura 4.20 – Trajectos *edge wrap*

Uma estação *edge-wrapping* é uma estação que implementa o *wrapping* nas fronteiras do trajecto. Para configurações *edge-wrap* a corrente de dados é ligada ao *ringlet* oposto, e participa no processamento do trajecto do *ringlet* oposto. Isto tem o efeito de permitir que o cliente alcance ambos os trajectos sem quaisquer ajustes na selecção do *ringlet*.

O bloco de controlo de recepção da fronteira de um trajecto *edge-wrap,* decompõe as tramas que entram, somente quando o restante trajecto não está a ser usado. A sua função poderia ser executada pela implementação do trajecto no mesmo *ringlet*.

O *edge-wrapping* não tem qualquer efeito na transmissão local de tramas de topologia e de protecção, que continuam a ser emitidas para fora de ambos os PHYs mesmo quando um PHY está a transmitir numa fronteira. Ao transmitir para dentro de uma fronteira, as estações *edge-wrapped* emitem uma cópia de qualquer trama *single-choke fairness* (SCFF) para essa fronteira e transmitem uma outra cópia da mesma trama para o trajecto circundado.

4.7 Descoberta da Topologia e Protecção

As funções de descoberta da topologia e de protecção do MAC do RPR, incluem a descoberta da topologia para que a modificação da mesma permita: a configuração e funcionamento automáticos da estação; a comutação da protecção para permitir atrasos abaixo de 50ms, sem reordenamento ou duplicação de tramas de dados estritas; a descoberta de atributo (ATD) para, adicionalmente, relatar entre estações informação menos crítica em termos de tempo; o *Checksum* da topologia (TC) para minimizar a perda de tramas de dados estritas devido às mudanças da topologia; e a medição do *loop round trip time* (LRTT) para optimizar o desempenho do modo conservador de *fairness.*

As funções de topologia, protecção, ATD, TC e medição do LRTT residem na sub-camada de controlo do MAC.

A função de protecção fornece mecanismos fiáveis para a comutação da mesma, com tempos inferiores a 50 ms, para todo o tráfego protegido num anel RPR. Permite o mecanismo imperativo de protecção denominado *steering*, e o mecanismo opcional de protecção denominado *wrapping*.

Este protocolo assegura que cada estação receba informação da mudança do estado da ligação, por exemplo falha da ligação ou informação de restauração, requerida para tomar com fiabilidade as decisões de comutação de protecção, e no instante requerido. Assegura também que as estações façam o *steer* ou *wrap* de acordo com a hierarquia de protecção do RPR.

A função de descoberta da topologia fornece meios fiáveis e exactos para que todas as estações num anel descubram a topologia das estações e todas as mudanças dessa topologia, nesse anel. Fornece também um mecanismo para a detecção rápida de mudanças da topologia. As funções de descoberta e de protecção da topologia estão intimamente relacionadas, dado que partilham de um mecanismo de mensagens de controlo comum.

A informação recolhida pelas funções de descoberta da topologia, protecção, ATD, TC, e de medição do LRTT, é armazenada numa base de dados de topologia, partilhada. A informação nesta base de dados é usada pelos protocolos de selecção do *ringlet* e de *fairness,* e pelo OAM.

4.8 OAM - Operação, Administração e Manutenção

O OAM fornece as funções de operação, administração e manutenção suportadas por estações RPR. Os métodos definidos pela norma IEEE 802.17 pretendem-se escaláveis, e que causem um *overhead* insignificante para o tráfego do anel, para o *software* e para os dispositivos de *hardware*. Este protocolo reside na sub-camada de controlo do MAC.

Os serviços e as características fornecidas são:

a) Determinação e/ou validação das ligações entre quaisquer duas estações no anel;

b) Determinação e/ou validação das operações de trajecto do trânsito para qualquer classe de serviço;

c) Operar sem a necessidade de uma estação mestra;

d) Fornecer um mecanismo para auxiliar na prevenção do desordenamento de tramas.

As áreas funcionais de gestão relacionadas com o RPR são:

1. A gestão da configuração que exerce controlo, identifica, colecta dados, e fornece dados às estações e às ligações entre as mesmas. É responsável pela instalação das estações, da sua interligação numa rede (configuração) e aprovisionamento;
2. A gestão de falha (ou manutenção) que permite a detecção, o isolamento, e a correcção de operação anormal das estações e da sua rede. É responsável por detectar e processar quaisquer falhas assim como relatá-las ao sistema de gestão.
3. A gestão do desempenho que avalia e relata sobre o comportamento das estações e da efectividade da rede e das estações para sustentação dos serviços. Fornece mecanismos para medir a qualidade de serviço, monitorizando o desempenho do sistema. Relata também a informação das estatísticas ao sistema de gestão.

As funções OAM suportadas pelo RPR, são baseadas em tramas especiais emitidas entre as estações num anel, que testam o estado operacional de um trajecto entre as mesmas. Estas tramas executam operações de pedido/resposta de eco e fornecem um método para evitar o desordenamento das tramas ao mudar o trajecto da ligação.

Os tipos de tramas OAM são as tramas de:

1. Eco — para a monitorização da ligação quando ordenada, e localização de falha no trajecto entre estações, ou para a determinação do cliente LRTT;
2. *Flush* — para a prevenção de desordenamento, controlada pelo cliente, ao mudar o sentido preferido do anel de um dado fluxo ou de uma determinação de RRTT pelo cliente;
3. *Organization Specific* — para codificação de informação específica de uma organização.

4.8.1 Gestão de falha

A gestão de falha inclui a vigilância do alarme, localização, correcção e teste da falha.

A vigilância do alarme fornece a capacidade de monitorizar as falhas detectadas nas estações. No suporte à vigilância do alarme, as estações executam verificações no *hardware* e no *software* com a finalidade de detectar falhas, e gerar alarmes para estas.

A localização de falha determina a causa da origem de uma falha. Para além da informação inicial da falha, pode usar a informação de falha de outras entidades com a finalidade de correlacionar e localizar aquela falha.

A correcção de falha é responsável por reparar uma falha e pelos procedimentos de controlo que usam recursos redundantes para substituir o equipamento que falhou. Testar a execução de funções de reparação usando rotinas de teste e de diagnóstico. O teste é caracterizado como a aplicação de sinais/mensagens e a sua medição.

Os mecanismos de gestão de falha são úteis para verificar a possibilidade de alcance entre duas estações no anel na camada do MAC, especialmente quando há algumas falhas que não são detectadas na camada 1.

4.8.1.1 Capacidade de pedido/resposta de eco do RPR

A capacidade de pedido de eco permite que uma trama seja introduzida numa estação no anel, e seja retornada uma resposta de eco por uma outra estação através do mesmo ou pelo *ringlet* oposto, com um impacto mínimo no fluxo de dados entre estações. As tramas de pedido/resposta de eco podem ter qualquer classe de serviço e conter um número de bytes especificado pelo utilizador, que pode ir até ao tamanho máximo de trama permitido. O pedido de eco *userData* é copiado para a trama de resposta.

O comando de pedido de eco transfere a informação do requerente ao objector, e é emitido no *ringlet0* ou no *ringlet1*. Um comando de pedido de eco determina o *ringlet* no qual o comando de resposta de eco é retornado.

Contexto de pedido de eco		*ringlet* de resposta
rinlet de recepção	*responseRinglet*	
-	RR_RINGLET0	*ringlet0*
	RR_RINGLET1	*ringlet1*
	RR_DEFAULT	*RingletSelection()*
ringlet0	RR_REVERSE	*ringlet1*
ringlet1	RR_REVERSE	*ringlet0*

Tabela 4.4 – Comandos de resposta a um eco

A estação de origem do pedido de eco gerará uma trama de pedido de eco emitida para a estação objectivo, em resposta a um pedido do cliente do MAC. Se uma estação emitir um pedido de eco, ambos o pedido e a resposta do eco serão explicitamente transmitidos no anel.

O cliente usa a função de MA_CONTROL.request para requerer a geração de uma trama de pedido do eco.

MA_CONTROL.request (OAM_ECHO_REQ, destination_address, service_class, ringlet_id, mac_protection, response_ringlet, user_data)

O MAC usa a função MA_CONTROL.indication para indicar ao cliente a recepção de uma trama de resposta de eco.

MA_CONTROL.indication (OAM_ECHO_IND, source_address, service_class, ringlet_id, protection_mode, response_ringlet, user_data)

A estação destinatária do eco responderá com uma trama de resposta ao eco, emitida para a estação de origem de pedido de eco, com o *ringlet* indicado pelos parâmetros de *responseRinglet* e de *protectionMode* no pedido de eco recebido. A sub-camada de controlo do MAC construirá a trama de resposta ao eco. A trama de resposta de eco inclui o *userData* da trama de pedido de eco.

O tempo previsto entre um pedido de eco e a resposta ao eco é uma função do tamanho do anel, da distância à estação de resposta, do tempo de processamento da estação de resposta e da classe de serviço escolhida para a trama de eco.

O tempo decorrido entre a recepção de uma trama de pedido de eco e a geração da resposta ao eco correspondente, não pode ser maior do que 10ms quando os pedidos de eco são recebidos a uma taxa inferior ou igual a uma vez em cada 100ms.

A unidade de resposta ao eco implementa as funções necessárias para o processamento de pedido de eco por uma estação objectivo e geração e transmissão da trama de resposta a esse mesmo eco.

4.8.1.2 Capacidade de *flush* do RPR

Um *flush* tem o efeito de cancelar no *ringlet* seleccionado, todo o tráfego previamente originado. Um *flush* deve ser usado quando há uma mudança no algoritmo de selecção do *ringlet*, quando os protocolos de selecção do *ringlet* são necessários para alcançar todas as estações (para protecção *steering*), ou para melhorar a utilização da largura de banda (para protecção *wrap*).

Um *flush* é uma trama de controlo especial que é emitida de uma estação para ela própria. Embora sejam necessários *flushes* distintos (um para o *ringlet0* e outro para o *ringlet1*), para se

assegurar a entrega completa de tráfego previamente originado, um *flush* é normalmente suficiente para fazer o *flush* de tráfego relevante. A trama *flush* pode também fazer o *flush* de tráfego previamente originado de um anel circundado.

O *flushing* não é possível num anel com protecção *steering*, dado que o tráfego é rejeitado (ao invés de circundado) nos pontos de extremidade. No entanto pode ser usado um eco endereçado a um ponto de extremidade, de uma forma similar.

O âmbito do *flush* é afectado pela classe do tráfego seleccionado:

a) ClasseA: o tráfego previamente originado da PTQ é *flushed*;

b) ClasseB ou ClasseC: todo o tráfego previamente originado da PTQ ou da STQ é *flushed*;

A estação *flush* de origem gerará uma trama *flush* em resposta a um pedido do cliente do MAC. A sub-camada de controlo do MAC formará a trama flush.

O cliente usa a função de MA_CONTROL.request para requerer a geração de uma trama *flush*.

MA_CONTROL.request (OAM_FLUSH_REQ, service_class, ringlet_id, mac_protection, user_data)

O MAC usa a função de MA_CONTROL.indication para indicar ao cliente a recepção de uma trama *flush*.

MA_CONTROL.indication (OAM_FLUSH_IND, service_class, ringlet_id, user_data)

4.8.1.3 Capacidade OAM especifica de uma organização

Uma estação RPR pode usar esta capacidade para executar funções adicionais de OAM não especificadas por esta norma.

A estação de origem específica de uma organização OAM gerará uma trama específica dessa organização em resposta a um pedido do cliente do MAC. A sub-camada de controlo do MAC formará uma trama específica de uma organização.

O cliente usa a função de MA_CONTROL.request para requerer a geração de uma trama específica de uma organização.

MA_CONTROL.request (OAM_ORG_REQ, destination_address, service_class, ringlet_id, mac_protection, organization_EUI, user_data)

O MAC usa a função de MA_CONTROL.indication para indicar ao cliente a recepção de uma trama específica de uma organização.

MA_CONTROL.indication (OAM_ORG_IND, source_address, service_class, ringlet_id, organization_EUI, user_data)

5 Arquitectura do Sistema

O desenvolvimento do sistema baseado na norma IEEE 802.17, está inserido num projecto composto essencialmente por três componentes: circuitos integrados de lógica programável; circuitos discretos de interface e processamento; e *firmware* de integração no sistema.

Neste trabalho foram apenas desenvolvidos os módulos que compõem o MAC RPR, cuja implementação teve como objectivo a integração em circuitos de lógica programável, como se verá no próximo capítulo.

A arquitectura do sistema segue o *Draft* 3.3 do IEEE 802.17, que corresponde à versão final pré-norma. Foram tomadas algumas opções ao longo do desenvolvimento relativamente a alguns dos modos da norma. Estas opções serão referidas ao longo do texto. Para discutir alguns pontos, será necessário aprofundar conceitos já mencionados no capítulo anterior. No entanto aspectos puramente de implementação serão deixados para o próximo capítulo.

5.1 Arquitectura global da implementação do MAC RPR

Os módulos da camada MAC do RPR e respectivas interfaces para o cliente e para a camada física, foram desenvolvidos tendo como objectivo a posterior implementação de um sistema tal como ilustrado na figura 5.1. Este sistema inclui as sub-camadas de *datapath* e de controlo do MAC, contendo esta última, os módulos de *Fairness*, Topologia e Protecção, e OAM. São também desenvolvidas adaptações para as interfaces MII para um cliente 100Mb/s Ethernet e GMII para a camada física 1000Mb/s Ethernet.

A implementação deste sistema requer a integração num PCB e codificação para um FPGA (figura 5.1), usando memória externa para armazenamento da base de dados da Topologia, um processador externo para cálculo de estatísticas, uso dos dois processadores internos do FPGA para implementação de funções passíveis de serem feitas por *software*, e interfaces externas MII e GMII.

Como ilustrado na figura 5.1 os módulos do MAC RPR foram desenvolvidos para serem implementados num circuito integrado programável (FPGA) e este circuito incluído numa placa PCB com diversos outros componentes necessários à implementação de todo o sistema. As unidades dentro da área quadrangular denominada “Xilinx Virtex II Pro FPGA”, são todas implementadas no FPGA e todas as unidades fora dessa área são componentes integrados no PCB. A figura ilustra apenas os componentes directamente ligados à funcionalidade do protocolo.

A arquitectura da implementação do sistema segue a norma do esboço 3.3 do IEEE 802.17.

Na arquitectura implementada para o trajecto de dados do MAC RPR, são utilizadas duas filas de trânsito (PTQ e STQ), são também usadas filas para entradas de tramas com as classes A, B e C. São também implementados todos os *shapers* que a norma inclui. O trajecto de dados implementado permite ser configurado com ambos os trajectos de protecção *center* ou *edge wrapping*.

Implementa ambos os algoritmos *AgressiveRateAdust* ou *ConservativeRateAdjust,* consoante configuração, para o algoritmo de *Fairness*.

O algoritmo de descoberta de topologia e protecção é implementado usando uma memória externa para armazenamento da base de dados de topologia.

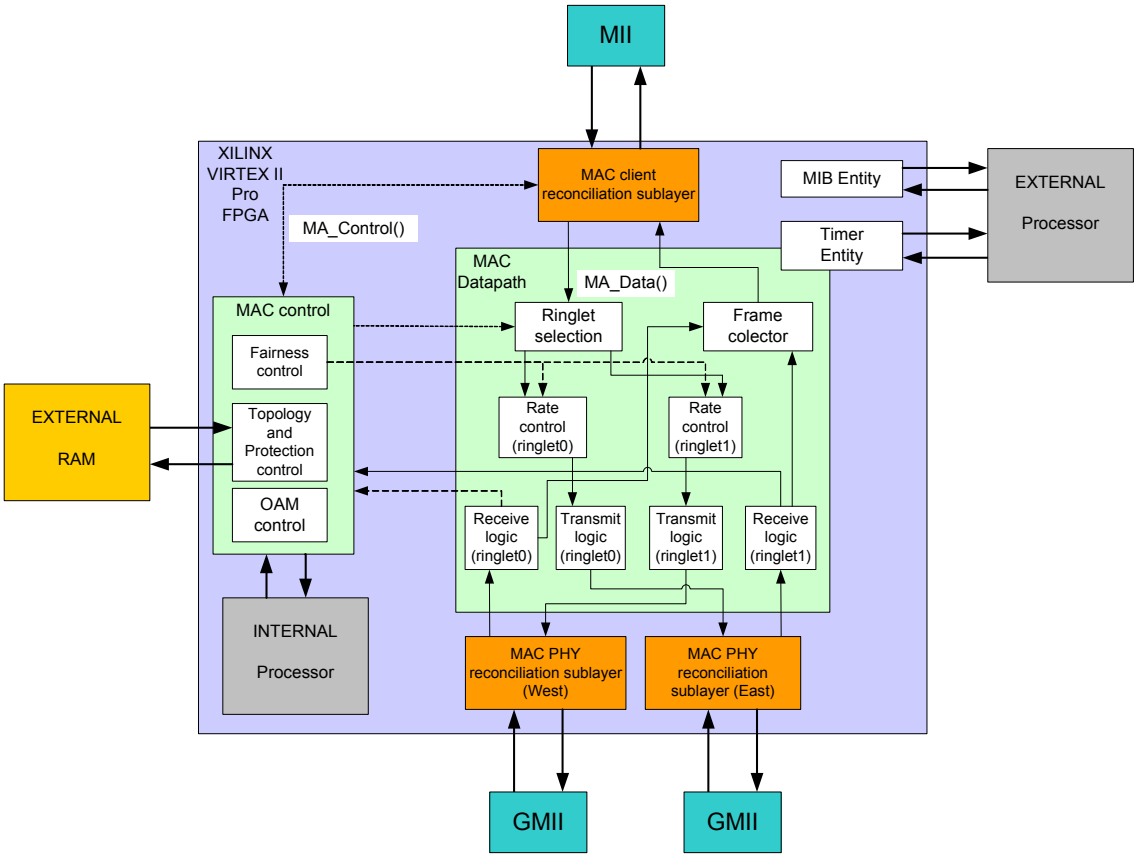


Figura 5.1 – Arquitectura do sistema MAC do RPR para Ethernet a 1000Mb/s

5.2 Trajectos de dados do MAC

A sub-camada dos *datapaths* (trajectos de dados) do MAC RPR, fornece a interacção entre o cliente e a camada física, e a comunicação entre as sub-camadas homologas dos trajectos de dados dos outros MACs no mesmo anel.

É composta por lógica de recepção de tramas que implementa as funcionalidades de: verificar se as tramas estão erradas e então são rejeitadas; contar as tramas para o cálculo de estatísticas de monitorização de tráfego; filtrar as tramas para determinar se devem ser copiadas para o cliente ou para a sub-camada de controlo; decompor as tramas (*stripping*) quando alcançam o seu destino; actualizar as tramas modificando-as para alcançar comportamentos adequados na estação seguinte; e enfileirar as tramas nas filas de trânsito.

É também composta por lógica de trânsito de tramas e de transmissão de tramas que contém os módulos de selecção do *ringlet,* de processamento da fila de estágio, de formatação (*shaping*) por classes de serviço, de selecção de transmissão de tramas e de *wrapping* (circundar) das tramas.

No desenvolvimento do sistema, as várias partes da sub-camada de *datapath* são unidas por intermédio de filas lógicas implementadas por módulos FIFO.

Por exemplo na figura 5.2, uma invocação de MA_DATA.request coloca a trama de entrada numa fila que conduz à máquina de estados de selecção do *ringlet*. Esta processa a trama de entrada, retira-a de uma fila e coloca-a então numa outra fila. Isto continua até que a última fila conduz a um PHY_DATA.request. Similarmente, a invocação de PHY_DATA.indication (pela sub-camada de reconciliação) fornece uma trama a cada uma das máquinas de estado de recepção, através de filas lógicas, e então eventualmente para uma ou ambas as máquinas de estado de transmissão (através da(s) fila(s) de trânsito) e que por fim conduz a um MA_DATA.indication.

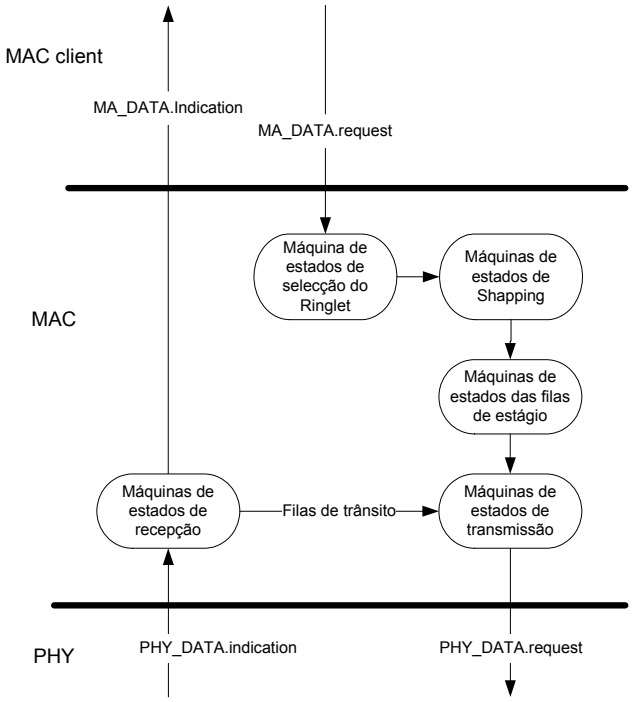


Figura 5.2 – Relações entre os trajectos de dados do MAC

O MAC suporta diversos trajectos ao longo dos quais as tramas se movem, como vamos ver.

As tramas adicionadas pelo MAC, ou pelo cliente, são transmitidas através dos denominados trajectos de adição, que incluem as filas de adição. As tramas recebidas (pretendidas a ser retransmitidas) são retransmitidas através dos chamados trajectos de trânsito, que incluem as filas de trânsito. Estes trajectos (figura 5.3) formam colectivamente os trajectos de dados do MAC, e há um conjunto destes trajectos por cada *ringlet*.

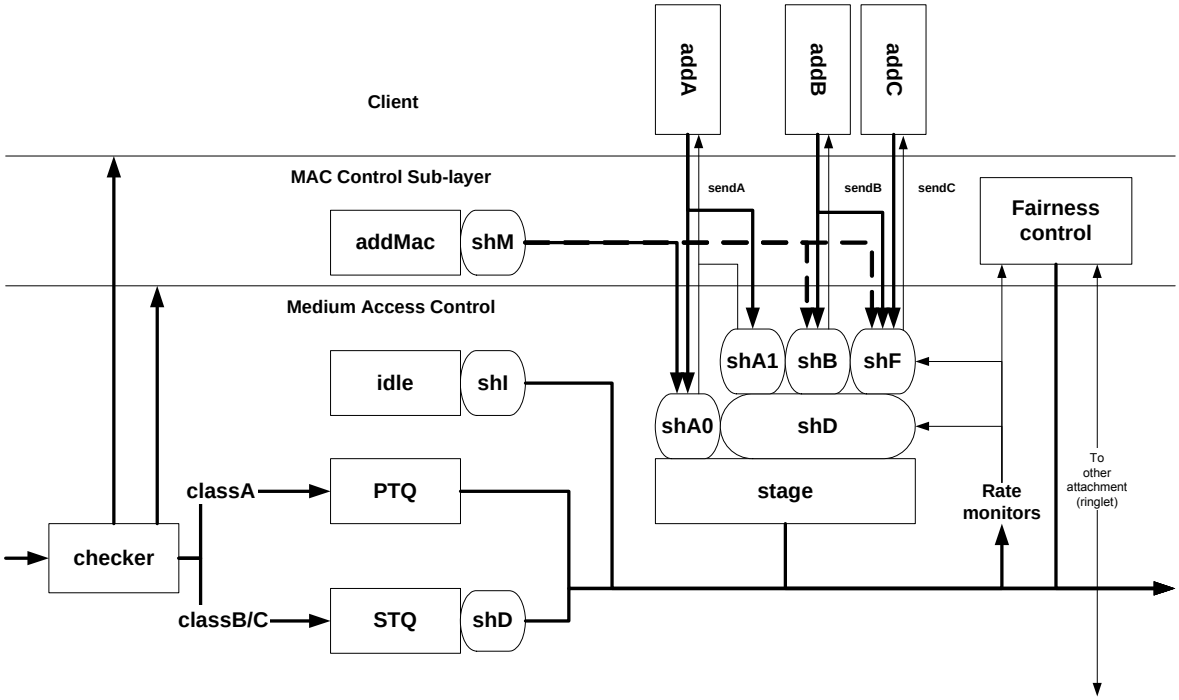


Figura 5.3 – Trajectos de dados (*datapaths*) do MAC

5.2.1 Trajectos de Adição

A arquitectura implementada para os trajectos de adição segue a descrição abaixo e engloba todas as unidades ilustradas na figura 5.3.

O cliente classifica as suas tramas de Classe-A, Classe-B, ou Classe-C. Um cliente de Classe-A adiciona tráfego cujo fluxo é controlado pelo *shaper* de subclasse-A0 (shA0, fig.5.3) ou pelo *shaper* de subclasse-A1 (shA1, fig.5.3), tal como determinado pelo MAC. Por sua vez, um cliente de Classe-B adiciona tráfego cujo fluxo é controlado pelo *shaper* de Classe-B (shB, fig.5.3) ou pelo *shaper fairness-eligible* (shF, fig.5.3), sendo também isto determinado pelo MAC. Por sua vez, um cliente de Classe-C adiciona tráfego cujo fluxo é controlado pelo *shaper fairness-eligible*.

O tráfego do cliente que é aceite, é colocado numa fila de estágio (*stage*), uma trama de cada vez.

As tramas adicionadas pela sub-camada de controlo do MAC são controladas pelo *shaper* de controlo do MAC (shM, fig.5.3). Estas tramas são geralmente classificadas de subclasse-A0 e controladas também pelo *shaper* de subclasse-A0, mas também podem ser classificadas de Classe-B ou Classe-C e serem controladas pelo *shaper* de Classe-B ou pelo *shaper fairness-eligible*, respectivamente.

Todos os *shapers,* à excepção do *downstream shaper* (shD, fig.5.3)*,* actuam apenas na adição de tráfego. O tráfego em trânsito é controlado apenas pelo *downstream shaper*. Este tráfego é monitorizado com a finalidade de ajustar o *shaper fairness-eligible* e o *downstream shaper* para além das suas actualizações baseadas em tráfego adicionado. Todo o tráfego adicionado não reservado e tráfego transitado são controlados pelo *downstream shaper*.

São adicionadas pelo MAC, tramas de *idle* de sincronização da taxa que são limitadas pelo *idle shaper* (shI, fig.5.3)*.*

5.2.2 Trajectos de Trânsito

Um MAC transita tramas que não são originadas nem terminadas nesse mesmo MAC. Há dois tipos de implementação de enfileiramento de trânsito no MAC: fila-única e fila-dupla, como referido atrás.

A implementação de fila-única coloca todo o tráfego em trânsito numa fila de trânsito primária (PTQ). A implementação de fila-dupla coloca o tráfego em trânsito de Classe-A, numa fila de trânsito primária de elevada precedência, e o de Classe-B e de Classe-C numa fila de trânsito secundária de baixa precedência (STQ). Nenhum destes trajectos suporta a interrupção de transmissão das tramas em trânsito ou no ingresso. Uma vez que uma trama tenha iniciado a transmissão, essa transmissão não pode ser interrompida pela transmissão de uma outra trama.

Na arquitectura implementada foi usado o enfileiramento em fila-dupla, como ilustrado na figura 5.3. Isto porque como se verá mais à frente, permite o uso de filas de trânsito com o comportamento de FIFOs simples, sendo assim mais fácil o controlo do tráfego prioritário de Classe-A que transita sempre pela PTQ.

5.2.3 Modo *Passthrough*

Quando uma estação determina (por exemplo, através das verificações de auto consistência ou dos monitores de *heartbeat*) que a sua sub-camada do MAC já não tem capacidade para executar com fiabilidade os procedimentos especificados para o MAC, essa estação pode entrar no modo de *passthrough*, no qual actua similarmente a um repetidor. Se uma estação estiver a operar neste modo, deixa de ser reconhecida como uma estação no anel.

O trajecto de trânsito e o PHY devem estar operacionais para que uma estação transite para o modo *passthrough*. O motivo de entrada no modo *passthrough* é específico da implementação, e deve estar dentro de determinados limites tais que a estação deva ainda poder executar as acções requeridas para suportar o trajecto de trânsito.

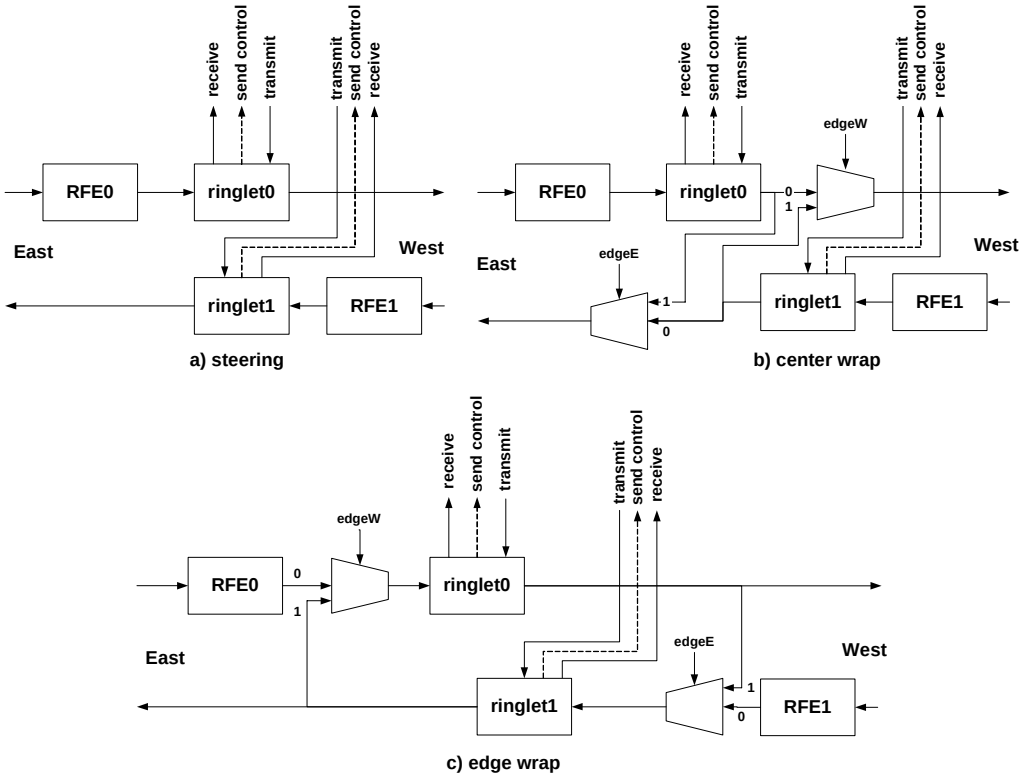
Se uma estação estiver *wrapped* (circundada), i. e, o seu porto de saída está ligado ao porto de entrada, deixa de o estar quando entra no modo de *passthrough.* Da mesma forma uma estação não pode fazer *wrapping* a ela mesma, quando está no modo de *passthrough*.

Na arquitectura implementada, neste modo de *passthrough* todas as tramas de dados, controlo e *fairness,* recebidas num dado *ringlet,* são retransmitidas no mesmo *ringlet* sem qualquer modificação. Isto tem as implicações do *ttl* não ser decrementado, do *bit* do campo *ps* não ser ajustado (ver anexo A) e das tramas não serem verificadas para ver se há erros, não sendo portanto rejeitadas. As tramas *idle* continuam a ser rejeitadas quando são recebidas e continuam a ser adicionadas dado que a função *idle* é suportada. As tramas de dados, controlo, e *fairness* não são adicionadas. As tramas transitadas continuam a ser enfileiradas na fila de trânsito apropriada e retiradas dessa fila por ordem de prioridade.

5.2.4 Trajectos de Protecção

Na implementação dos trajectos de protecção das estações, estas suportam trajectos protegidos tipo *steering*, *center-wrap* ou *edge-wrap*. Ambas as parcelas de recepção e de transmissão das estações são afectadas pelo tipo de protecção que uma estação suporta (definido por *myProtectMethod,* ver anexo A) e pelo estado actual de protecção da extensão que está em recepção ou em transmissão (definido por *myEdgeState,* ver anexo A).

O direccionamento (*steering*) das tramas (figura 5.4a) é controlado pela existência de uma fronteira (*edge*) entre a origem e o destino. As tramas são dirigidas para fora do trajecto pretendido se a estação for uma estação de *steering* (definido por *protConfig,* ver anexo B), se existe uma fronteira entre a origem e o destino, e se a trama é elegível para ser re-direccionada (definido por *mac_protection*).



RFE0: Bloco de Receive From Edge Control que faz a decomposição de tramas entrantes no *ringlet0*
RFE1: Bloco de Receive From Edge Control que faz a decomposição de tramas entrantes no *ringlet1*
edgeE: Multiplexador que selecciona receber de East ou desde o trajecto circundado se for uma fronteira para East
edgeW: Multiplexador que selecciona receber de West ou desde o trajecto circundado se for uma fronteira para West

Figura 5.4 – Métodos de *Wrap*

O *wrapping* (circundar) das tramas (figuras 5.4b e 5.4c) é controlado por diversas condições. As tramas são circundadas se a estação for uma estação *wrapping* (determinado por *protConfig*), se a estação local está circundada (determinado por *myEdgeState*), e se a trama é elegível para ser circundada (determinado pelo campo *we* da trama).

Cada estação está implementada com capacidade de *wrapping* e tem trajectos circundáveis, que permitem que as tramas façam a ligação para trás para o *ringlet* oposto, depois das falhas da ligação terem sido detectadas. Há dois métodos de executar o *wrapping*, cada um com efeitos diferentes na adição de tráfego do cliente, e com diferentes formas de possível redundância. Na arquitectura implementada, os sistemas *wrapping* podem escolher um ou outro método.

Há duas excepções em que as tramas não são transmitidas através de uma fronteira. As excepções são tramas de *fairness* e tramas de topologia e protecção, transmitidas de uma estação numa fronteira à estação vizinha na fronteira do outro lado da extensão. Estes tipos de tramas são explicitamente emitidos para dentro da fronteira, para permitir funcionamentos correctos dos protocolos de topologia e de protecção. Se suceder que estas tramas cruzem a fronteira da extensão, então são decompostas na estação vizinha. Todas as restantes tramas são rejeitadas (se não forem *wrap eligible*) ou circundadas (se forem *wrap eligible*) numa fronteira.

A unidade de recepção da trama requer o conhecimento da posição relativa de qualquer falha local. Este conhecimento é mantido pela variável *myEdgeState*, que indica o estado da fronteira em que a unidade de recepção está a operar, e é descrito pelos valores: NORMAL em que não há qualquer falha num ou noutro lado; PASSTHROUGH em que a estação está em *passthrough* em ambos os *ringlet0* e ringlet1; INTO_EDGE em que a unidade está a transmitir numa extensão defeituosa; e FROM_EDGE em que a unidade está a receber de uma extensão defeituosa. Nestes dois últimos casos, se a localização da falha é na extensão EAST o *myEdgeState* do trajecto do *ringlet0* tem o valor INTO_EDGE e o do *ringlet1* tem o valor FROM_EDGE. Se caso contrário a localização da falha é na extensão WEST o *myEdgeState* do trajecto do *ringlet1* tem o valor INTO_EDGE e o do *ringlet0* tem o valor FROM_EDGE.

5.3 *Rate Control*

Na arquitectura implementada a taxa de transmissão de todo o tráfego adicionado e de algum tráfego em trânsito é controlada para serem mantidas garantias de classe de serviço. Isto tem também o efeito de limitar a precedência estrita das decisões de transmissão, tal que cada classe de serviço receba a sua parte justa das transmissões.

5.3.1 Resumo do MAC *shaper*

Estão implementados diversos *shapers* (ver fig. 5.3) que geram os sinais *sendA, sendB* e as indicações de sendC que são fornecidos ao cliente, os sinais *sendI* e *sendM* que são fornecidos à sub-camada de controlo do MAC e a indicação de *sendD* que é fornecida à sub-camada do trajecto de dados do MAC. Todos os *shapers* e indicações operam por *ringlet*, portanto existe uma unidade de MAC *shaper* por cada *ringlet*.

Os comportamentos da maioria destes *shapers* podem ser caracterizados por um algoritmo comum com parâmetros específicos.

A maioria dos *shapers* consiste num *token bucket*. O *token bucket* tem uma profundidade máxima por defeito de pelo menos *mtuSize* bytes. Os créditos no *token bucket* são incrementados pelo tamanho de incremento em todos os intervalos de actualização. O número máximo de créditos que podem acumular num *token bucket* depende do tipo de *shaper*. Quando uma trama está a aguardar pelo acesso ao anel no início de uma fila, é-lhe concedido o acesso ao anel somente se o número de créditos no *token bucket* for superior ou igual ao limite inferior. O número de créditos num *token bucket* é decrementado pelo tamanho de cada trama enquanto está a ser transmitido. Os créditos serão reduzidos assim que cada byte é emitido, ou pelo menos com uma frequência de cada 256 bytes emitidos, e sempre no final de cada transmissão da trama. Nunca deve ser possível que o número de créditos se torne negativo após se ter subtraído o número de bytes numa trama transmitida.

Assim sendo, os créditos dos *shapers* são ajustados para baixo ou para cima. Os valores de decremento e de incremento representam tipicamente os tamanhos de uma trama transmitida e os créditos em cada intervalo de actualização, respectivamente.

O cruzamento abaixo do nível do limite mínimo gerará uma indicação de limitação de taxa (a remoção de uma indicação de *send*), de modo que o tráfego oferecido possa parar antes de alcançar um número de créditos nulo, onde as transmissões em excesso são rejeitadas. Quando nenhuma trama está pronta para a transmissão e para limitar o *burst* de tráfego após intervalos de inactividade, os créditos são reduzidos ao limite mínimo (se estiverem acima deste limite mínimo) e podem acumular até um valor não superior a este limite mínimo. Na maioria dos *shapers* o limite mínimo é ajustado *a mtuSize* com a finalidade de permitir que seja feita a transmissão de uma trama completa sem reduzir os créditos abaixo de zero.

O nível de limite máximo limita os créditos positivos, para evitar *overflow*. Quando as tramas estiverem prontas para ser transmitidas (e estão a ser obstruídas pelo trânsito de tráfego), os créditos podem acumular até ao limite máximo. O valor do limite máximo será pelo menos *mtuSize*. Se for ajustado a exactamente *mtuSize* o *burst* de pior caso, para uma trama pequena, é mínimo (não é superior a uma das tramas maiores).

5.3.2 Controlo de fluxo da fila de adição

O MAC não mantém filas para fazer o *shaping* ao tráfego adicionado pelo cliente, mas fornece indicações por *ringlet* e indicações de controlo de fluxo por classe. O MAC também não aceita pedidos do cliente quando a indicação apropriada de *send* não estiver activa. As indicações de controlo de fluxo são requeridas para serem enviadas ao cliente somente quando há mudanças nos valores *sendA*, *sendB* e *sendC*.

Alguns dos trajectos de transmissão são afectados apenas por um destes *shapers*, outros são influenciados por múltiplos *shapers*.

As tramas *idle* da fila de adição, para sincronização da taxa do MAC, são controladas pelo *idle shaper* (*sendI*).

As tramas de controlo da fila de adição do MAC são controladas pelo *shaper* de controlo do MAC (*sendM*). As tramas de controlo são também controladas pelo *shaper* para a classe de serviço escolhida. Desde que o tráfego de controlo tenha precedência sobre o tráfego do cliente, os *shapers* baseados na classe estrangulam somente o tráfego do cliente.

Todo o tráfego adicionado de Classe-A é controlado pelo *shaper* de Classe-A (*sendA*) para evitar que o cliente exceda as suas taxas reservadas de Classe-A. O *shaper* de Classe-A é dividido logicamente no *subshaper* subclasse-A0 e no *subshaper* subclasse-A1.

Todo o tráfego adicionado de Classe-B é controlado pelo *shaper* de Classe-B (*sendB*) e/ou pelo *shaper fairness eligible*, para confinar o cliente dentro das suas taxas de Classe-B-CIR e de Classe-B-EIR.

Todo o tráfego de Classe-C é controlado pelo *shaper fairness eligible* (*sendC*), para confinar o cliente dentro do seu uso ponderado da justa parte de largura de banda não utilizada e reclamável.

Todo o tráfego adicionado de subclasse-A1, Classe-B e Classe-C, e todo o tráfego em trânsito de Classe-B e de Classe-C de uma STQ é adicionalmente controlado pelo *shaper downstream* (*sendD*), para confinar o MAC para poder suportar a taxa de subclasse-A0 reservada para jusante.

5.4 Operação de Recepção

A operação de recepção é iniciada quando da recepção de um PHY_DATA.indication e as suas funções incluem os seguintes componentes:

a) Verificação: As tramas, que entram erradas, são rejeitadas ou ajustadas. As estatísticas de erro, relacionadas com o MIB, são actualizadas;

b) Contagem: Os parâmetros das tramas, que entram válidas, são usados para actualizar as estatísticas de fluxo relacionadas com o MIB;

c) Filtragem: É passada uma cópia para o filtro de recepção;

d) *Strip*: É passada uma cópia ao componente de transmissão, ou decomposta (*stripped*) no anel;

e) Ajuste: Os parâmetros dentro da trama (os campos *ttl* e *ps*) são ajustados. O campo de *hec* baseado no CRC16 do cabeçalho é ajustado sempre que os campos protegidos pelo *hec* tenham sido alterados.

5.4.1 Operação de recepção para tramas de dados estritas

As tramas de dados estritas são tramas com o campo *so* (*strict order*) ajustado a 1. Em qualquer circunstância é necessário que estas tramas sejam entregues estritamente por ordem (dentro de qualquer conjunto {*sa, da, service_class*}) e sem quaisquer duplicações.

5.4.1.1 Conteúdo de contexto

Um contexto é uma imagem do anel que inclui as estações, extensões e estados das ligações, tal como visto da estação local, e está armazenado na base de dados de topologia e de estado desta estação. Cada trama é transmitida dentro de um contexto implícito. Por exemplo, tramas transmitidas antes de uma mudança na imagem do anel podem ter um contexto obsoleto.

Com a finalidade de evitar qualquer duplicação ou desordenamento das tramas, as estações de recepção usam um mecanismo de conteúdo de contexto que remove do anel as tramas de dados estritas que foram transmitidas usando um contexto obsoleto, antes da transmissão das mesmas usando um contexto actualizado.

O mecanismo de conteúdo de contexto é accionado pela recepção de uma trama de controlo de protecção o que resulta em mudanças na base de dados local da topologia e do estado da estação. O mecanismo de conteúdo de contexto envolve o rejeitar das tramas de dados estritas que poderiam ser desordenadas ou duplamente recebidas. Quando o conteúdo de contexto é accionado, todas as tramas de dados estritas numa fila secundária de trânsito (STQ), serão rejeitadas, incrementando o *containedFrames* para cada trama rejeitada. Quando o conteúdo de contexto é accionado, todas as tramas de dados estritas que se encontram na fila Q_TX_SS (Transmitter Stage Queue Selection.) serão também rejeitadas, sem incrementar o *containedFrames* para estas tramas, dado que não foram transmitidas. Adicionalmente, enquanto o conteúdo de contexto permanece activo, as tramas de dados estritas são rejeitadas antes de terem permissão para serem transitadas ou transmitidas.

Para além das regras de verificação do *ttl* usadas para todas as tramas, as tramas de dados estritas serão também rejeitadas antes de serem emitidas para o cliente, se o número de estações (*hops)* que atravessaram não for consistente com a base de dados de topologia e de estado da estação de recepção. Esta verificação extra de descarte, assegura que não ocorra qualquer duplicação da trama devido a que por exemplo, uma estação intermédia entre em *passthrough*, ou para todas as sequências perigosas de *wrap/unwrap*. Ambos os sistemas *wrapping* e *steering* executam esta verificação nas tramas de dados estritas que estão a ser recebidas para serem copiadas para o cliente.

Qualquer estação pode opcionalmente executar a verificação de consistência como parte da unidade de verificação e recepção, com a finalidade de conservar a largura de banda do anel removendo as tramas possivelmente defeituosas assim que a inconsistência for detectada.

5.4.1.2 Recepção em sistemas *steering*

Os sistemas *steering,* que tratam das tramas de dados estritas, usam o mecanismo de conteúdo de contexto quando uma estação recebe um evento de protecção. Quando da recepção de tal trama de controlo de protecção, as estações de recepção *steering* removem todas as tramas de dados estritas que estão efectivamente dentro das filas de trânsito de ambos os *ringlets*, e

continuam a remover todas as tramas de dados estritas recebidas até ordem em contrário por parte das funções de topologia e de protecção. Após a duração ter expirado, a estação retorna às operações normais e transita todas as tramas tal como requerido. O resultado final desta operação é a remoção das tramas de dados estritas no anel, que foram despachadas usando um contexto que já não está actualizado.

O reordenamento pode ocorrer durante uma comutação da protecção, tal como quando uma estação de origem transmite tramas usando um contexto seguido por um contexto diferente. O mecanismo de conteúdo de contexto assegura que não haja qualquer reordenamento das tramas. O conteúdo de contexto remove as tramas despachadas usando um contexto antigo antes que estas sejam despachadas usando um contexto novo, para impedir qualquer reordenamento das mesmas.

5.4.1.3 Recepção em sistemas *wrapping*

O acto de *unwrapping* do anel (isto é, recuperação do anel) pode cativar tramas no *ringlet* errado. Estas tramas são rejeitadas do *ringlet* secundário se não existir qualquer evento de protecção. No entanto, no caso de uma segunda falha que ocorra imediatamente após a recuperação do anel, é provável que nem todas as tramas sejam rejeitadas, e fazer o *wrapping* destas tramas no seu *ringlet* primário (se permitido) causaria reordenamento. Consequentemente, a seguir ao acto de *unwrapping* de um anel, todas as estações eliminam todas as tramas de dados estritas em trânsito e nas filas de trânsito de ambos os *ringlets* cujo *o ri* não coincide com o *ringlet* em que viajam. Este estado persiste até que a topologia e os protocolos de protecção determinem que já não é necessário. Se ocorrer um novo *wrap* dentro deste período, pode ocorrer alguma perda adicional de tramas (das tramas recentemente circundadas) mas não ocorre qualquer reordenamento.

Os sistemas *wrapping* executam verificações adicionais para além daquelas verificações executadas em sistemas *steering*, pois é possível que uma série de falhas possa deixar a topologia significativamente diferente de quando a trama foi originalmente lançada.

5.4.2 Unidades da operação de recepção

Esta secção descreve as unidades implementadas, de recepção das tramas desde a camada física ou desde um ponto de *wrap*.

5.4.2.1 Componentes da recepção

Cada estação tem componentes de transmissão, de recepção e de filtragem que usam regras projectadas para suportar tráfego RPR local e *bridged*. As funções de recepção incluem os seguintes componentes:

a) Edge: As tramas recebidas através de uma fronteira são decompostas;

b) Check: As tramas erradas são rejeitadas;

c) Count: As estatísticas de fluxo de tráfego são actualizadas;

d) Strip: As tramas seleccionadas são decompostas para remoção, ou transitadas;

e) Adjust: Os cabeçalhos das tramas são ajustados;

f) Queue: As tramas são enfileiradas na PTQ ou na STQ.

g) Filter: As tramas são descartadas se forem inválidas ou são contadas e transitadas para o cliente ou para o controlo do MAC, se forem de dados ou de controlo, respectivamente.

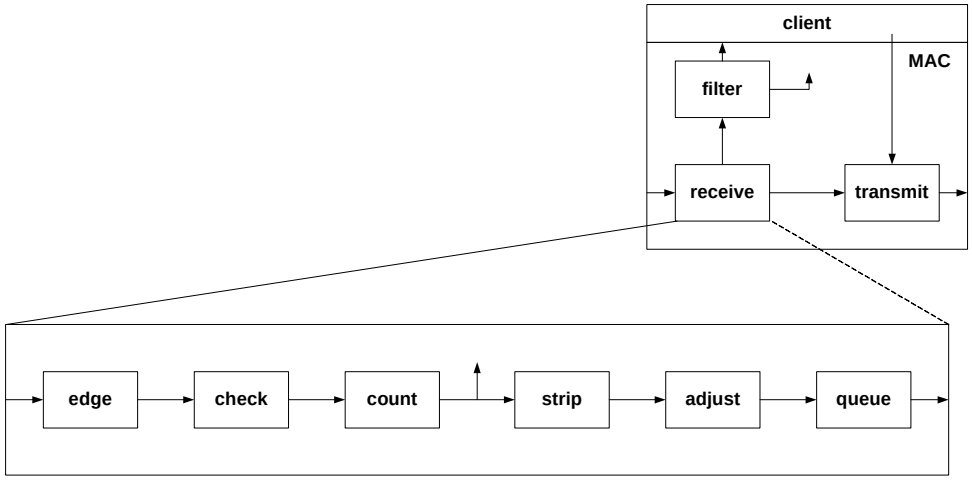


Figura 5.5 — Componentes da recepção

5.4.2.2 Resumo das regras de remoção (*strip*)

Em geral, as acções de remoção são permitidas somente quando o *ringlet* da trama e o *ringlet* do *datapath* são o mesmo. A excepção a esta regra é para as estações de fronteira que são *center wrapped*, onde algumas tramas serão removidas no anel oposto. Quando assim permitidas, as regras de remoção afectam a remoção das tramas do anel, ao não as transitar.

Condição do campo da trama			**Acção**
fi	*da*	*sa*	
-	-	*sa == myMacAddress*	*(remoção)*
fi == FI_NONE	*da == myMacAddress*	*sa != myMacAddress*	
	da != myMacAddress	*sa != myMacAddress*	*CopyToTransit()*
fi != FI_NONE	-	*sa != myMacAddress*	

Tabela 5.1 — Regras de remoção (*strip)*

5.4.2.3 Filtragem

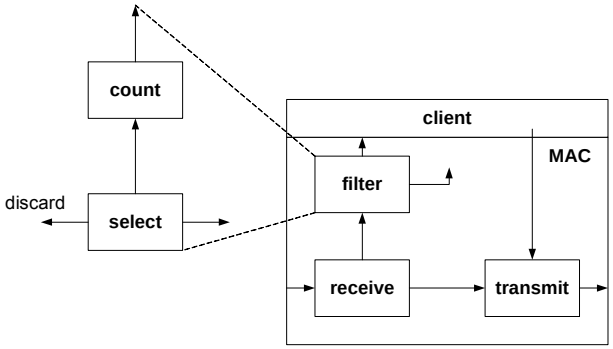


Figura 5.6 — Componentes da filtragem

As funções do filtro incluem os seguintes componentes:

a) *Select*: As tramas são seleccionadas para uma das seguintes acções:

1. *Data*: as tramas de dados são passadas ao cliente;
2. *Control*: as tramas de controlo são passadas à entidade de controlo do MAC;

3. *Discard*: as tramas inválidas de dados e de controlo são rejeitadas;

b) *Count*: As tramas seleccionadas são contadas como se segue:

1. *Data*: as tramas de dados e os bytes são contados por classe de serviço antes de serem passadas ao cliente;
2. *Control*: as tramas de controlo são contadas por *controlType* antes de serem passadas ao cliente.

Em geral, as acções de filtragem copiam as tramas para o cliente ou para a sub-camada de controlo do MAC, somente quando o *ringlet* da trama e o *ringlet* do *datapath* são o mesmo. A excepção a esta regra é para as estações de fronteira que são *center wrapped*, onde algumas tramas serão copiadas do anel oposto.

Condição			**Acção**
Condições dos parâmetros das tramas		*myRxFilter*	
da	*sa*		
da == myMacAddress	*sa != myMacAddress*	-	*CopyToClient()*
Multicast(da)	*sa != myMacAddress*	-	
da != myMacAddress && !Multicast(da)	*sa != myMacAddress*	RX_FLOOD	
		RX_BASIC	-
-	*sa = myMacAddress*	-	

Tabela 5.2 — Sumário da selecção da filtragem do cliente

As diferenças básicas entre o conjunto das duas regras de filtragem são que as tramas *self-sourced* (*sa = myMacAddress*) não são emitidas para o cliente, mas são emitidas para a sub-camada de controlo do MAC, e um cliente pode eleger a recepção das tramas não dirigidas a ele, mas as tramas de controlo não são emitidas à sub-camada de controlo do MAC a menos que dirigidas à estação.

5.5 Operação de Transmissão

A operação de transmissão é iniciada na recepção de um MA_DATA.request, de um MA_CONTROL.request, ou de qualquer trama originada na sub-camada de controlo. A operação de transmissão consiste na selecção do *ringlet*, seguida do processamento da fila de estágio (*stage*) e, por fim, seguida da selecção de transmissão.

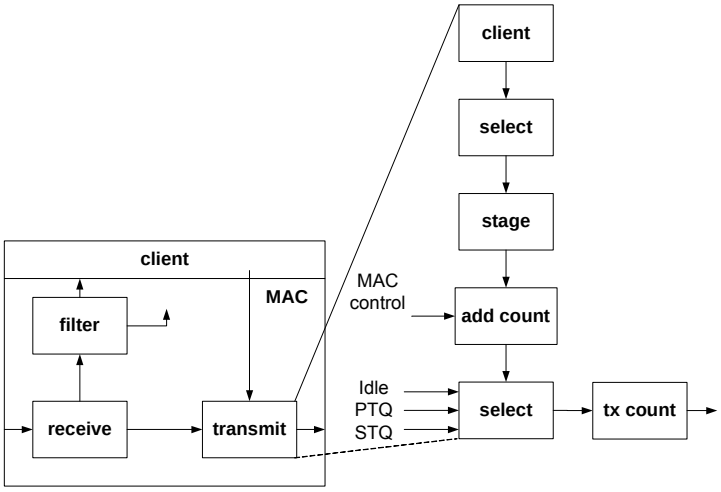


Figura 5.7 — Componentes da Transmissão

5.5.1 Selecção do *Ringlet*

Na maioria dos casos é escolhido um único *ringlet* (*ringlet0* ou *ringlet1*). Nalguns casos podem ser escolhidos ambos os *ringlets* e, neste caso, a unidade de selecção do *ringlet* é também responsável pela repetição da trama para ambos os *ringlets*.

A selecção do *ringlet* é implementada para a adição de tráfego do cliente e para a adição de tramas OAM. Executa as seguintes acções:

a) Escolhe o(s) *ringlet*(s) apropriado(s) no(s) qual(is) vai emitir tramas;

b) Redirige porventura as tramas, baseando-se nos ajustes de protecção para a trama e para a estação;

c) Altera porventura o endereço de destino inserido na trama, baseando-se na avaliação opcional de ajustes do endereço de destino secundário;

d) Determina se as tramas necessitam ser *flooded*;

e) Ajusta os campos *sa, da, saExtended, daExtended, ef, ttl, ttlBase, ri, we, fi, ps* e *so,* das tramas.

A primeira acção para a selecção do *ringlet* é a de escolher o *ringlet* apropriado para a adição de tramas do cliente e do OAM. A escolha é baseada nos parâmetros de *ringlet_id* e de *mac_protection*, e na base de dados de topologia e de estado da estação. O cliente pode exercer o controlo completo, nenhum controlo ou controlo parcial sobre o(s) trajecto(s) que uma trama toma.

Para os valores do parâmetro de *destination_address* que não estão contidos na base de dados de topologia e de estado, a selecção do *ringlet* faz o *flood* da trama para todas as estações no anel. Baseado no estado de protecção do anel e na configuração de preferência do *flooding*, este pode ser feito somente num *ringlet* ou em ambos os *ringlets*.

A selecção do *ringlet* usa a informação da base de dados de topologia e de estado da estação:

a) Topologia do anel: é usada para determinar a distância a uma outra estação desde a estação local e para decidir fazer o *flood* da trama se essa posição não puder ser determinada;

b) Disponibilidade de outras estações: a disponibilidade do trajecto a uma estação de destino é usada para modificar a escolha do *ringlet* para tramas protegidas em anéis *steering*;

c) Mecanismo de protecção do anel: este mecanismo é usado para determinar se usar o *steering* ou o *wrapping* como mecanismo de protecção para a estação.

Quando a protecção está activa, a selecção do *ringlet* redirige (opcionalmente) as tramas adicionadas para anéis *steering* ou para falhas locais de ligação em anéis *wrapping* usando um esquema *center wrap*, ou opcionalmente repete as tramas *multicast* para transmissão em ambos os *ringlets*. O campo *we* (*wrap eligible*) é ajustado baseado no parâmetro opcional de *mac_protection* e no mecanismo de protecção activa do anel. Este mecanismo permite a co-existência de tramas *steered* e *wrapped* do mesmo cliente, em anéis *wrapping*.

5.5.1.1 Controlo do cliente na selecção do *ringlet*

A selecção do *ringlet* é invocada para cada trama gerada pela primitiva MA_DATA.request, e para cada trama OAM gerada pela sub-camada de controlo do MAC. A selecção é controlada através do parâmetro opcional *ringlet_id* e do parâmetro opcional *mac_protection*, ambos são ajustados pela primitiva MA_DATA.request fornecida pelo cliente, ou pelo componente OAM da sub-camada de controlo do MAC.

Se não forem especificados, o valor para o *ringlet_id* será por defeito RI_DEFAULT e o valor para o *mac_protection* será TRUE por defeito.

Um cliente só pode requisitar a emissão de uma trama quando permitido pela indicação de *send* para a classe de serviço e para o *ringlet* requisitados. Para uma escolha RI_DEFAULT do *ringlet*, ou uma escolha do *ringlet* que possa ser cancelada devido à protecção *steering*, o cliente não

pode saber antecipadamente qual o *ringlet* que deve ser escolhido, e deve consequentemente usar as indicações de emissão de ambos os *ringlets*.

5.5.1.2 Determinação de *flooding*

A selecção do *ringlet* usa os parâmetros de MA_DATA.request, a preferência de *flooding* dada por defeito em *myFloodingForm* e a informação contida na base de dados de topologia e de protecção, determinam se e como fazer o *flood* de uma trama. A preferência por defeito do *flooding* não é alterável, excepto no momento em que há mudanças na topologia ou na protecção, com a finalidade de evitar requisitar novamente as tramas.

São fornecidas duas alternativas de *flooding* para tramas de dados:

Unidireccional: Transferência de trama que envolve a emissão de uma trama de *flooding* somente num *ringlet*, para todas as estações nesse *ringlet.* O *sa* encontrado no cabeçalho da trama é o endereço do MAC local de origem. O campo *fi* é ajustado a FI_UNIDIR para esta alternativa de *flooding*.

Bidireccional: Transferência de trama que envolve a emissão de duas tramas de *flooding*, uma em cada *ringlet*, onde cada trama é dirigida às distintas estações adjacentes. O intuito das tramas *flooded* é primeiramente orientado pelo *ttl* dentro do cabeçalho da trama. O campo *fi* é ajustado a FI_BIDIR para esta alternativa de *flooding*.

Parâmetros do cliente		Parâmetros do cabeçalho	Parâmetros do *payload*	
destinationAddress	*sourceAddress*	*fi*	*daExtended*	*saExtended*
-	!myMacAddress	!FI_NONE	*destinationAddress*	*sourceAddress*
local	=myMacAddress	=FI_NONE	-	-
-	=myMacAddress	!FI_NONE		

Tabela 5.3 — Regras de *flooding*

5.5.1.3 Substituição do endereço secundário do MAC

A selecção do *ringlet* suporta a substituição do endereço MAC de destino. Isto é permitido por estações que anunciam que desejam receber o tráfego que está a ser emitido a um ou dois endereços extras do MAC (endereços secundários do MAC) para além dos seus endereços MAC da estação (endereços primários do MAC). Quando ocorre um pedido de transmissão de uma trama com um endereço de destino com um valor do endereço MAC secundário de uma outra estação, a estação de emissão pode substituir o endereço MAC primário associado ao endereço MAC secundário fornecido. Isto permite que estas tramas sejam emitidas sem *flooding* e tenham o benefício de serem decompostas no destino. A substituição de endereços secundários do MAC é transparente a todas as parcelas do MAC (incluindo a selecção do *ringlet*) à excepção da própria substituição opcional.

5.5.2 Determinação do ponto de *cleave*

Uma trama pode ser repetida e emitida sobre ambos os *ringlets* no caso de uma trama bidireccional *flooded* ou no caso de uma trama de *broadcast* ou *multicast* emitida num anel em que ocorre um evento de protecção. Em ambos os casos, às tramas repetidas não é permitido sobrepor a sua possível entrega, e não podem consequentemente ser emitidas para além de um ponto no anel comum a ambas as tramas, o ponto de *cleave*. A potencial sobreposição é impedida pelo ajuste dos valores do *ttl* em ambas as tramas, ao número de extensões até ao ponto de *cleave*, determinado pelo número de extensões até à estação mais próxima do ponto de *cleave*.

No caso de um anel em que esteja a ocorrer um evento de protecção, o ponto de *cleave* para tramas repetidas é o ponto em que o evento de protecção ocorre. Se houver mais do que um

evento de protecção, o ponto de *cleave* pode ser escolhido relativamente a algumas estações em estado de estação de fronteira.

No caso de um anel onde não esteja a ocorrer um evento de protecção, a determinação do ponto de *cleave* para tramas repetidas é específico da implementação. No entanto, o ponto de *cleave* não pode mudar a menos que mude tanto o tipo de topologia como os conteúdos da mesma.

5.5.3 Ajuste do *ttl* e *ttlBase*

Cada trama transmitida desde o cliente local ou da sub-camada local de controlo do MAC, terá o seu campo *ttl* ajustado de acordo com as regras de condicionamento do valor do *ttl,* e o seu campo *ttlBase* ajustado ao valor do seu campo *ttl*.

Valor de *ft*	Valores relevantes adicionais	Condições ao valor inicial do *ttl*
FT_IDLE	-	*ttl* == 1
FT_CONTROL	*frame.controlType* == CT_STATION_ATD	*ttl* == MAX_STATIONS
	frame.controlType == CT_TOPO_CHKSUM	*ttl* == 1
	frame.controlType == CT_TOPO_PROT	*ttl* == MAX_STATIONS
	frame.controlType == CT_LRTT_REQ	*ttl >= numberOfHops(frame.da) && ttl* <= MAX_STATIONS
	frame.controlType == CT_LRTT_RSP	*ttl >= numberOfHops(frame.da) && ttl* <= MAX_STATIONS
	frame.controlType == CT_FDD	*ttl >= numberOfHops(frame.da) && ttl* <= MAX_STATIONS
	frame.controlType == CT_OAM_ECHO_REQ	*ttl >= numberOfHops(frame.da) && ttl* <= MAX_STATIONS
	frame.controlType == CT_OAM_ECHO_RSP	*ttl >= numberOfHops(frame.da) && ttl* <= MAX_STATIONS
	frame.controlType == CT_OAM_FLUSH	*ttl >= (numStations - 1) && ttl* <= MAX_STATIONS
	frame.controlType == CT_OAM_ORG	*ttl >= numberOfHops(frame.da) && ttl* <= MAX_STATIONS
FT_FAIRNESS	*frame.ffType* == SINGLE_CHOKE	*ttl* == MAX_STATIONS
	frame.ffType == MULTI_CHOKE	*ttl* == MAX_STATIONS
FT_DATA	*frame.fi* == FI_BIDIR	*ttl == numberOfHops(cleavePoint)*
	frame.fi == FI_UNIDIR	*ttl >= (numStations - 1) && ttl* <= MAX_STATIONS
	frame.fi == FI_NONE	*ttl >= numberOfHops(frame.da) && ttl* <= MAX_STATIONS

Tabela 5.4 — Valores iniciais do campo *ttl*

O ajuste do *ttl* para um valor superior ao necessário pode interferir com a reclamação da largura de banda o que impede a utilização total da mesma em toda a extensão do anel.

5.5.4 Selecção da *Stage Queue*

É fornecida uma fila de estágio de entrada única para o cliente e para as transmissões de controlo sem atraso de acesso. A fila de estágio pode guardar em qualquer altura no máximo uma trama. Assim que esvaziar, a trama seguinte, com prioridade mais elevada, é seleccionada para ser a seguinte trama adicionada a ser transmitida.

A fila de estágio retira as tramas de uma fila lógica de selecção que permite a análise das mesmas e o desfilar selectivo de uma trama. Esta fila não fornece um comportamento de um FIFO (alternativamente, pode ser vista como um agrupamento de filas FIFO, com uma para cada um dos conjuntos {classe de serviço, tipo de trama}). O tamanho da fila de selecção depende da implementação e pretende-se que seja suficientemente grande para armazenar durante um período de uma volta pelo anel, desde uma indicação de emissão ao cliente até à acção de recepção de uma trama transmitida de volta.

Não é necessário que uma trama inteira esteja disponível se a fila de estágio for implementada em *cut-through* ou tenha atrasos suficientemente pequenos para que possa receber o resto da trama à taxa máxima da mesma. A quantidade que necessita estar disponível, uma entrada, é a quantidade que o MAC necessita para a transmissão à taxa máxima. A entrada tem de conter pelo menos o cabeçalho.

Uma trama que já se encontra na fila de estágio é considerada ter sido transmitida, do ponto de vista do cliente do MAC. Isto inclui todas as medidas de atraso ou *jitter*. A aceitação na fila de estágio é o equivalente à aceitação para transmissão. A selecção de uma trama para aceitação na fila de estágio, e consequentemente para transmissão, é controlada pelos *shapers* de classe, o *shaper downstream*, e o algoritmo de *fairness*.

A aceitação de tramas de controlo e de tramas de dados do cliente, para transmissão, é feita na ordem estrita de precedência, limitada somente por todo o controlo da taxa pelas indicações de *send*. Os controlos da taxa têm o efeito de limitar a precedência estrita das decisões de transmissão tal que cada classe de serviço tenha a sua parte justa de transmissões.

5.5.5 Unidades de Data e Control Add Count

Estas unidades suportam a contagem necessária para actualizar as estatísticas das tramas de dados e de controlo adicionadas, respectivamente.

5.5.6 Cálculos de congestionamento do MAC

As implementações de fila-única e de fila-dupla calculam e detectam o congestionamento de diferentes maneiras.

As implementações de fila-única medem o congestionamento com temporizadores de atraso de acesso e com a utilização da ligação. Há temporizadores de acesso separados para tráfego de Classe-A, de Classe-B e de Classe-C, cada um mantido para cada *ringlet*. O temporizador que corresponde à classe da trama é iniciado no momento em que a trama se torna na *head-of-line* no MAC. O temporizador é restaurado quando o MAC transmite essa trama com sucesso. Para tramas de Classe-A, se o MAC for incapaz de transmitir a trama antes que o temporizador expire, este trata isto como um erro reportável do aprovisionamento. Para tramas de Classe-B e de Classe-C, se o MAC for incapaz de transmitir a trama antes que o temporizador expire, este trata isto como tendo sido excedida a condição de *lowThreshold*.

As implementações de fila-dupla medem o congestionamento com níveis da STQ. Existem ambos um nível baixo, *lowThreshold* e um nível elevado, *highThreshold,* para cada *ringlet*. Um nível é excedido sempre que o número de bytes enfileirados na STQ é maior do que o valor desse nível.

5.5.7 Sincronização da taxa de transmissão

As estações e as redes RPR podem ser implementadas com PHYs síncronos ou assíncronos. Numa rede síncrona, o relógio de transmissão para cada estação é referenciado a uma fonte de sincronismo comum que pode ser recuperada dos dados de recepção ou fornecida através de uma interface externa de sincronismo, e a taxa de dados de transmissão de cada estação é exactamente idêntica à taxa de dados recebida. No entanto, numa rede assíncrona, a taxa de dados de transmissão em cada estação é determinada por uma fonte de relógio local, e a taxa de dados de transmissão de cada estação, varia ligeiramente da taxa de dados nominal da rede. No caso de uma estação que transmita a uma taxa de dados mais lenta do que a estação precedente, isto poderá causar um *overflow* da fila de trânsito primária.

O *idle shaper* descreve uma função opcional de sincronização da taxa de transmissão que elimina a possibilidade de *overflow* da fila de trânsito primária, introduzindo um número variável de tramas *idle* no fluxo de transmissão. A função de sincronização da taxa de transmissão faz parte do *datapath* do MAC. Embora projectado para a operação com estações assíncronas, esta função pode ser incluída para todas as implementações, incluindo as de operação síncrona da rede. Nesta implementação está incluída esta unidade.

A função da sincronização da taxa de transmissão suporta PHYs com tolerâncias de relógio de dados até ±100 PPM.

5.5.8 Implementação de um MAC de fila-dupla

O comportamento da transmissão de um MAC de fila-dupla é descrito pelos seus protocolos de selecção/transmissão e pelas funções de *shaping*. O protocolo de selecção/transmissão e as funções de *shaping* são essencialmente independentes, mas a sua cooperação (através do *shaper* interno que fornece o *sendM* e as indicações de *send* específicas associadas à classe de serviço) é necessária para assegurar a compatibilidade das transmissões da trama de controlo do MAC. Este acoplamento adicional (através do *shaper* interno que fornece a indicação de *sendD*) é necessário para sustentar correctamente as transmissões *downstream* de subclasse-A0.

Um MAC da fila-dupla usa duas filas de trânsito, a PTQ para tráfego de Classe-A, e a STQ para tráfego de Classe-B e de Classe-C. O tamanho da STQ determina os seus valores limite de controlo de fluxo.

Tendo como referência a figura 5.3, os seguintes comportamentos serão suportados:

a) Ordenamento da PTQ – O ordenamento do FIFO será mantido quando as entradas passam através da PTQ;

b) Ordenamento da STQ – O ordenamento do FIFO será mantido quando as entradas passam através da STQ;

c) Ordenamento cruzado – Uma entrada da STQ não sairá antes de uma entrada previamente recebida na PTQ.

5.5.9 Selecção da transmissão numa fila-dupla

O comportamento da transmissão de um MAC de fila-dupla é descrito pelos seus protocolos de selecção/transmissão e pelas funções de *shaping*. O protocolo de selecção/transmissão e as funções de *shaping* são independentes para *sendA, sendB* e *sendC.* Mas é necessário o acoplamento *de sendM* e das indicações de *send* associadas às classes de serviço, para assegurar a compatibilidade das transmissões da trama de controlo.

O resultado final desta unidade é que uma trama é transmitida através de uma invocação de PHY_DATA.request. Quando é requerido ao MAC para fornecer a especificação do comprimento para a transmissão do PHY, este processamento é dependente da implementação.

Um MAC de fila-dupla pode transmitir tramas de dados de quatro possíveis filas internas (*idle*, PTQ, STQ e *Stage*) por *ringlet*.

A unidade de selecção da transmissão numa fila-dupla implementa as funções necessárias para seleccionar que trama deve transmitir num MAC de fila-dupla. A intenção é a de esvaziar sempre a PTQ antes das transmissões da trama do cliente, mas permitir que a STQ encha de alguma forma enquanto adiciona tramas de transmissão do cliente. Os créditos dos algoritmos de *shaping* são decrementados enquanto uma trama é processada.

5.5.10 Contagem da transmissão

A unidade de contagem da transmissão (fig. 5.7) executa as funções necessárias para contar bytes e tramas transmitidos, e implementa as funções necessárias para actualizar as estatísticas do MIB para o fluxo de tráfego.

5.6 Fairness

Esta secção descreve o algoritmo usado para o cálculo de taxas justas, para o acesso ao *ringlet,* de tráfego elegível para *fairness*. O uso de taxas justas impede que uma estação ocupe uma parte

desproporcional da capacidade disponível no *ringlet,* relativamente a outras estações no mesmo *ringlet*.

5.6.1 Âmbito

Uma estação contém dois módulos do algoritmo de fairness. Cada módulo regula o tráfego de dados associado a um dos *ringlets*. Um módulo de *fairness* é identificado pela combinação do endereço da sua própria estação (*myMacAddress*) e do *ringlet* (*myRI*) cujo tráfego é regulado por esse módulo. Cada módulo de *fairness* emite, periodicamente, informação de controlo de congestionamento, para a sua estação vizinha a montante através do *ringlet* oposto (*otherRI*). A informação de controlo de congestionamento está contida numa trama de *fairness* que transporta o identificador do *ringlet* (*otherRI*) no qual é emitida.

A figura 5.8 ilustra o comportamento dos módulos de *fairness* num anel fechado. O módulo de *fairness* do *ringlet0* recebe a informação da taxa de tráfego do *ringlet0* desde um monitor de byte associado a este *ringlet*. Este mesmo módulo regula o tráfego no *ringlet0* fornecendo indicações de policiamento ao módulo de trajecto de dados do mesmo *ringlet*. Este módulo também recebe e emite tramas de *fairness* através do módulo de trajecto de dados do *ringlet1*. As tramas de *fairness* recebidas e emitidas por este módulo transportam um valor igual a 1 (*frame.ri* = 1) na identificação do *ringlet*. O módulo de *fairness* do *ringlet1* comporta-se de um modo similar.

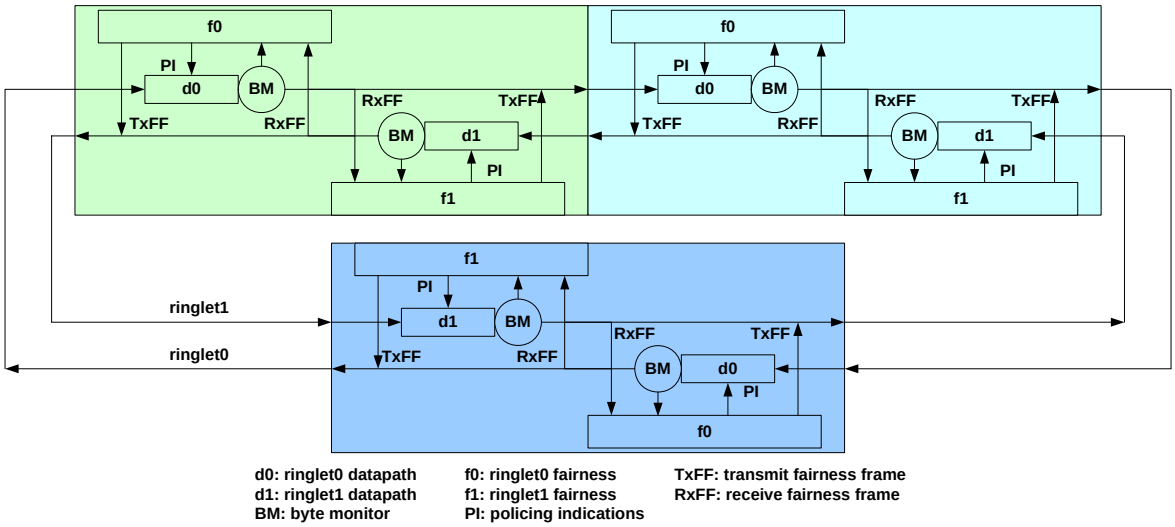


Figura 5.8 — Módulos de *fairness* num anel fechado

A figura 5.9 ilustra os módulos de *fairness* num anel aberto usando o método de protecção *steered*. A identificação e o comportamento dos módulos de *fairness* são idênticos aos associados a um anel fechado excepto que as tramas de *fairness* transmitidas por um módulo de *fairness* numa extensão inactiva, não alcançam o módulo de *fairness* da estação vizinha a montante e não podem, consequentemente, ser recebidas pelo módulo de *fairness* desta estação.

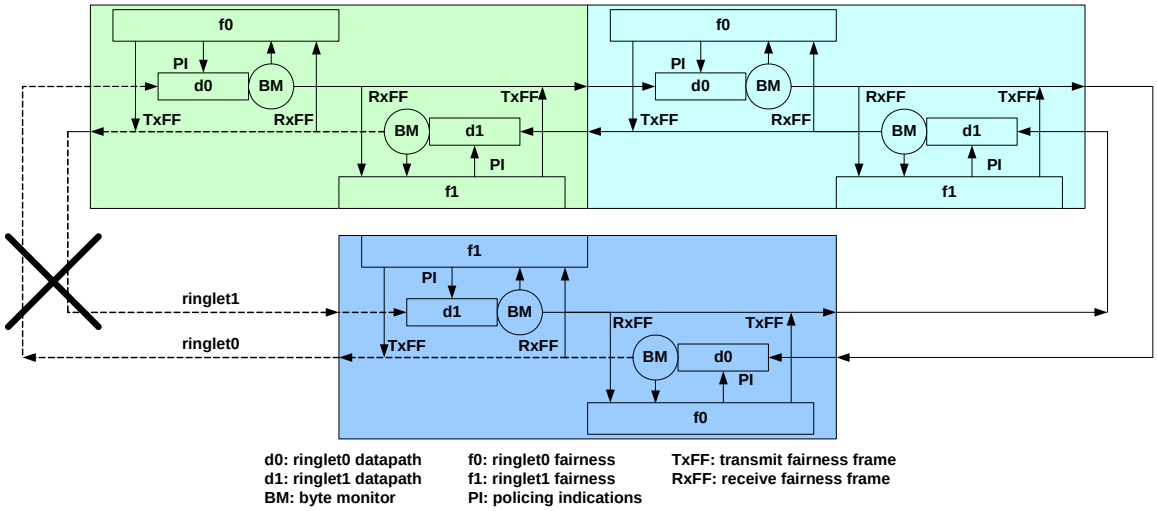


Figura 5.9 — Módulos de *fairness* num anel aberto usando *steering*

A figura 5.10 ilustra os módulos de *fairness* num anel aberto usando o método de protecção *wrapped*. A identificação e o comportamento dos módulos de *fairness* nas estações que não são estações de fronteira, são idênticos aos de estações num anel fechado. A identificação e o comportamento de estações de fronteira dependem se a estação executa *edge-wrapping* ou *center-wrapping*.

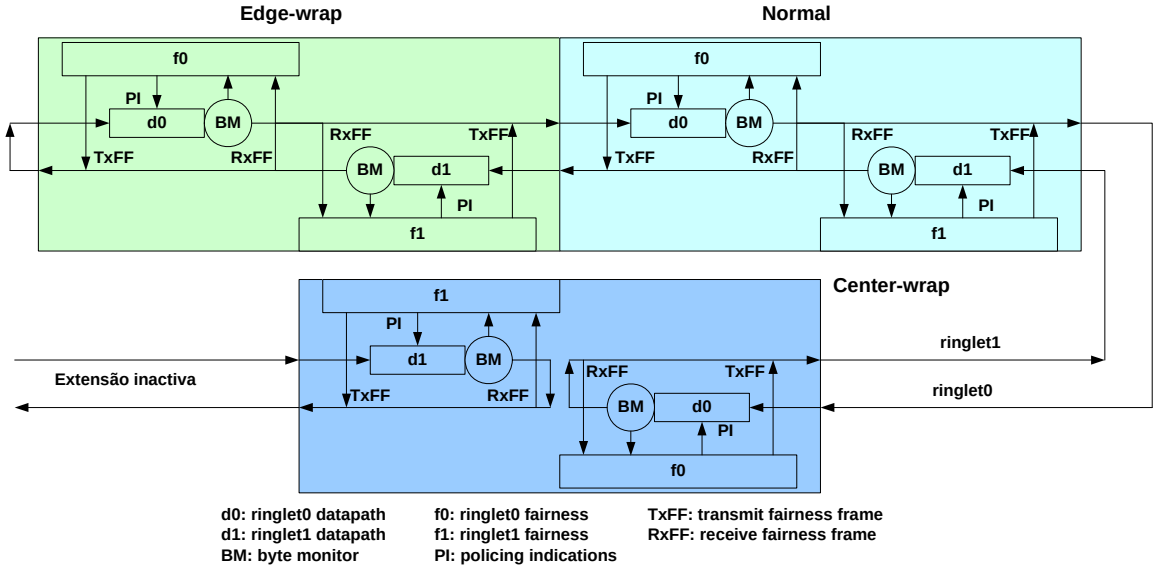


Figura 5.10 — Módulos de *fairness* num anel aberto usando *wrapping*

Como ilustrado pela estação à esquerda em cima na figura 5.10, uma estação *wrapped* que implementa um trajecto *edge-wrapped*, transita os dados através de ambos os módulos de trajecto de dados associados à estação.

A estação ao centro em baixo na figura 5.10, é uma estação *wrapped* que implementa um trajecto de dados *center-wrapped*, e transita dados através de um único módulo de trajecto de dados. O módulo (de *fairness)* associado à extensão activa recebe e emite tramas (de *fairness)* através dessa extensão. O módulo associado à extensão inactiva não receberá quaisquer tramas da sua estação vizinha a jusante, devido à inactividade da extensão. Por sua vez, as tramas emitidas pelo

módulo associado à extensão inactiva não alcançarão o módulo vizinho a montante devido também à inactividade da extensão.

No caso de um anel fechado ou de um anel aberto que implementa a protecção *steering*, o congestionamento é controlado independentemente em cada *ringlet*. A informação de controlo de congestionamento, tal como os dados de tráfego, não é transferida de um *ringlet* ao outro. No caso de um anel aberto, a informação de controlo do congestionamento não pode cruzar uma extensão em falha no *ringlet*.

No caso de um anel aberto que implementa a protecção *wrapping*, a informação de controlo de congestionamento pode ser circundada de um *ringlet* ao outro, permitindo o controlo do congestionamento associado com o tráfego que foi circundado de um *ringlet* ao outro. No caso de uma estação de fronteira implementando *center-wrapping*, um módulo de *fairness* recebe tramas de *fairness* num *ringlet* e emite tramas de *fairness* no *ringlet* oposto. No caso de uma estação de fronteira implementando *edge-wrapping*, uma trama de *fairness* é processada sequencialmente por ambos os módulos de *fairness*. Num ou noutro caso, as tramas de *fairness* são propagadas extensão a extensão ao longo de um trajecto que é o inverso do seguido pelo tráfego associado aos dados.

5.6.2 Sumário do algoritmo de *Fairness*

O algoritmo de *fairness* regula a adição a um *ringlet,* de tráfego elegível para *fairness* no qual:

a) O congestionamento é controlado;

b) O *throughput* é minimamente afectado por actividades de controlo do congestionamento;

c) As limitações da taxa que controlam o congestionamento são razoavelmente aplicadas através das estações que contribuem para o congestionamento.

5.6.2.1 Identificação do congestionamento local

A arquitectura implementada foi a de uma estação com um MAC de fila-dupla que está congestionada quando a ocupação da STQ é excessiva.

Na situação oposta, uma estação implementada com um MAC de fila-única está congestionada quando ocorre qualquer uma ou ambas as seguintes condições:

a) A taxa de transmissão é excessiva relativamente à capacidade de transmissão da ligação;

b) O tráfego é excessivamente atrasado ao esperar pela transmissão.

O congestionamento é indesejável porque pode resultar numa falha nos compromissos extremo-a-extremo associados às classes de serviço. No caso de serviço *best-effort* não associado a um compromisso extremo-a-extremo explícito, o congestionamento pode permitir às estações a montante do ponto de congestionamento, usarem mais do que a sua parte justa da capacidade disponível para tráfego *best-effort* a jusante do ponto de congestionamento.

5.6.2.2 Exemplo de controlo do congestionamento

A figura 5.11 ilustra uma estação congestionada (S6) e o conjunto de estações contíguas (S1-S6) que contribuem para o congestionamento em S6.

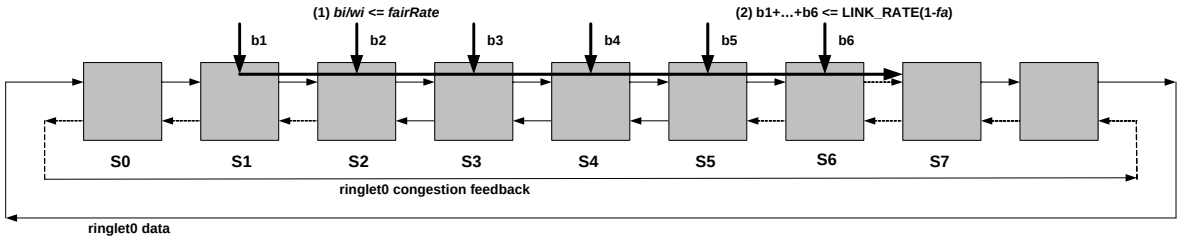


Figura 5.11 — Objectivos do controlo do congestionamento

Cada estação contribuinte está associada a uma taxa (*bi*) à qual adiciona tráfego elegível para *fairness* que cruza a ligação da estação congestionada S6. A taxa *bi* é escalonada por um peso administrativo (*wi*) que permite que uma estação adicione a uma taxa mais alta ou mais baixa do que outras estações, sem violar os princípios de *fairness*. O LINK_RATE representa a capacidade do *ringlet* e o *fa* representa a fracção da capacidade consumida pelo tráfego reservado de precedência elevada (classeA e classeB-CIR). O objectivo do algoritmo de *fairness* é calcular um *fairRate* aplicado às estações contribuintes, tal que sejam alcançados os seguintes objectivos:

a) *bi/wi* ≤ *fairRate*: As estações contribuintes maximizam a sua taxa ponderada-ajustada sem exceder *o fairRate;*

b) *b1+…+b6* ≤ LINK_RATE*(1-fa)*: A soma do tráfego elegível para *fairness,* transmitido pela estação congestionada, maximiza o uso da capacidade disponível sem exceder essa capacidade.

5.6.2.3 Propagação do *fairRate*

Com a finalidade de se alcançar a condição *bi/wi* ≤ *fairRate* em cada estação contribuinte (S1 a S6), o *fairRate* calculado pela estação congestionada é propagado extensão-a-extensão no sentido montante (propaganda de taxa), fazendo a divulgação desse valor a cada uma das estações contribuintes.

A propaganda de taxa permite que cada estação contribuinte limite a sua taxa, *bi*, ao *fairRate* ponderado-ajustado, resultando em mudanças nas estatísticas da taxa medidas na ligação a jusante da estação congestionada. As estatísticas da taxa são usadas para se assegurar que a condição *b1+…+b6* ≤ LINK_RATE*(1-fa)* é alcançada ao se ajustar *o fairRate*. O *fairRate* ajustado é então anunciado para montante.

Uma estação pode anunciar um de três valores possíveis à sua estação vizinha a montante:

a) O seu *fairRate* localmente calculado;

b) O *fairRate* anunciado pela sua estação vizinha a jusante;

c) O valor FULL_RATE que indica que as estações a montante não estão a contribuir para o congestionamento a jusante.

Uma mensagem de propaganda transporta a identidade da sua estação de origem. Uma estação que anuncia o FULL_RATE identifica-se sempre como a origem. Em todos os restantes casos, a origem é a estação em cujos *fairRate* localmente calculados e MAC ADDRESS da origem, são transportados pela propaganda. A propaganda transporta também um campo de *Time to Live* ao qual é atribuído o valor de MAX_STATIONS pela estação de origem e é decrementado por cada estação através da qual passa.

5.6.2.4 Domínio de congestionamento

Uma estação que anuncia um *fairRate*, localmente calculado, e igual a non-FULL_RATE, encontra-se na cabeça de um domínio de congestionamento. A ligação a jusante da estação da cabeça é conhecida como ponto de congestionamento.

O domínio de congestionamento estende-se para montante da estação da cabeça através de todas as estações que propagam o anúncio até alcançar uma estação que já não propague esse mesmo anúncio. A estação final a montante do domínio de congestionamento é conhecida por cauda. Uma estação da cauda recebe uma propaganda que contém um *fairRate* igual a non-FULL_RATE, mas não propaga mais para montante, os anúncios recebidos.

Na figura 5.12, a estação S1 é a cauda do domínio de congestionamento A. S2 recebe o anúncio originado pela estação da cabeça, S3, do domínio de congestionamento A. A estação S1 termina o domínio de congestionamento A ao anunciar o FULL_RATE para montante à estação S0. A estação S4 ilustra um caso em que a cauda do domínio de congestionamento B a jusante, é também a cabeça do domínio de congestionamento A a montante. A estação S3 recebe uma propaganda originada por S4 mas anuncia o seu *fairRate,* localmente calculado, para S2 a montante.

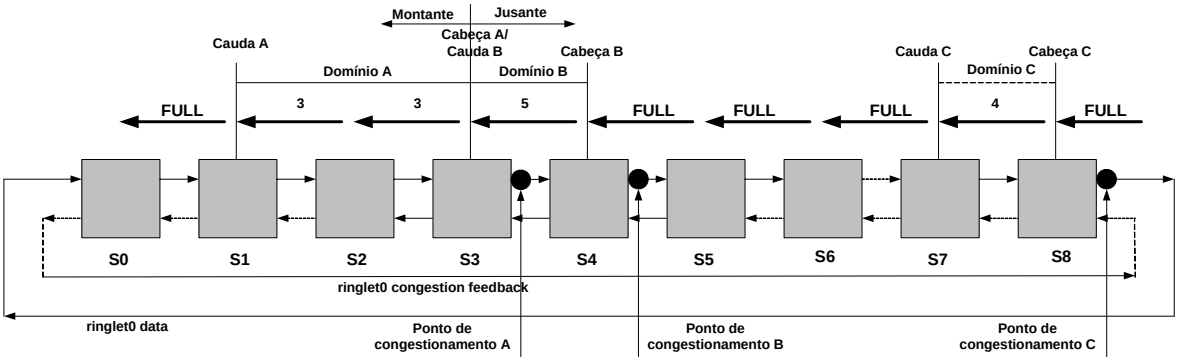


Figura 5.12 — Domínios de congestionamento

Uma estação propaga um *fairRate* recebido somente quando esse *fairRate* é menor do que ou igual ao seu *fairRate* localmente calculado. É uma característica de um domínio de congestionamento que o *fairRate* da cabeça não é maior do que (isto é, pelo menos tão restritivo quanto) aquele de qualquer outra estação dentro do domínio de congestionamento. As estações (sem ser da cabeça) dentro do domínio de congestionamento podem ou não estar localmente congestionadas. Cada estação no domínio de congestionamento ajusta a sua taxa *bi* baseando-se no *fairRate* anunciado pela cabeça. O FULL_RATE é anunciado em qualquer ligação sem ser dentro de um domínio de congestionamento.

A cauda de um domínio de congestionamento é uma estação que recebe uma propaganda de non-FULL_RATE mas que não propaga esse anúncio (isto é, terminando o domínio de congestionamento). Um domínio de congestionamento é terminado quando ocorre qualquer uma das seguintes circunstâncias:

a) A estação local começa um novo domínio de congestionamento (isto é, reúne as condições associadas às da cabeça de um domínio de congestionamento);

b) Nenhuma estação, a montante, adiciona tráfego elegível para *fairness* que transita o ponto de congestionamento, a uma taxa que exceda o *fairRate* anunciado pela cabeça do domínio de congestionamento.

Ao avaliar a última condição, a estação local não está ciente das taxas de adição de estações individuais a montante. Ao invés, a estação local considera o tráfego total medido elegível para *fairness* que transita ambos, ela própria e o ponto de congestionamento. O tráfego contribuído por qualquer estação individual a montante, não pode exceder o tráfego total contribuído pelas outras estações a montante. Este total é tido como um limite superior da taxa adicionada de tráfego elegível para *fairness* de qualquer estação individual a montante.

5.6.2.5 Informação da taxa permitida

Além da restrição ao tráfego adicionado elegível para *fairness* que transita por um ponto de congestionamento, uma estação pode também restringir o tráfego agregado adicionado ao *ringlet* e elegível para *fairness* (isto é, independente de se o tráfego transita um ponto de congestionamento). Esta taxa agregada é conhecida por *allowedRate.* O *allowedRate* não pode exceder *o unreservedRate* do *ringlet*.

A taxa, *bi*, à qual uma estação adiciona tráfego elegível para *fairness* que transita pelo ponto de congestionamento é conhecida dentro do algoritmo de *fairness* como *o allowedRateCongested.* Os *allowedRate*, *allowedRateCongested* e *hopsToCongestion* são conhecidos colectivamente como informação de taxa permitida. Uma estação recalcula periodicamente os seus *fairRate* e informação de taxa permitida. O período entre cálculos é conhecido como o *agingInterval* dado que um dos cálculos executados é o "envelhecimento" dos contadores da taxa. A duração do *agingInterval* é baseada na taxa de dados do anel e tem o mesmo valor em todas as estações no anel.

A informação de taxa permitida é opcionalmente transferida ao cliente do MAC através de um MA_CONTROL.indication que tem um *opcode* de SINGLE_CHOKE_IND. A informação de taxa permitida pode ser usada pelo cliente do MAC para executar actividades de *fairness* para além do âmbito do MAC.

O *fairRate* calculado pela estação é relatado por *broadcast* a todas as restantes estações no *ringlet* usando uma trama de *fairness multi-choke*. O tempo entre *broadcasts* sucessivos do *fairRate* de uma estação é conhecido como o *reportingInterval* e é um múltiplo configurado do *advertisingInterval.* Ao conhecer *o fairRate* de uma estação, este é opcionalmente transferido ao cliente do MAC através de um MA_CONTROL.indication que tem um *opcode* de MULTI_CHOKE_IND. Esta informação pode também ser usada pelo cliente do MAC para executar actividades de *fairness* para além do âmbito do MAC.

5.6.2.6 Normalização da taxa

Uma taxa comunicada de uma estação a outra é normalizada com a finalidade de:

a) Assegurar que a taxa seja interpretada uniformemente pelas estações no *ringlet*;

b) Dimensionar o valor da taxa para permitir a codificação eficiente como um valor inteiro dentro do campo de 16-bit de *fairRate* da trama de *fairness*.

A normalização de uma taxa é executada dividindo a taxa localmente significativa por um coeficiente de normalização (*normCoef*). O *normCoef* é calculado como o produto de três valores, como mostrado na equação 5.1:

$$normCoef = localWeight * rateCoef * ageCoef \quad \text{(Eq. 5.1)}$$

localWeight — normaliza a taxa consistentemente com um padrão de *localWeight* igual a um;

rateCoef — normaliza a taxa consistentemente com um padrão *lineRate* de 2,5 Gb/s;

ageCoef — normaliza a taxa consistentemente com uma unidade padrão de bytes-per-*agingInterval.*

Quando recebida por uma estação, uma taxa normalizada pode ser usada como se segue:

a) Convertida a uma taxa localmente significativa (isto é, localizada) multiplicando pelo *normCoef* local*;*

b) Mantida no formato normalizado para a propagação a outras estações;

c) Comparada a uma taxa local que tenha sido normalizada.

Uma taxa normalizada pode ser convertida a uma taxa localmente significativa. O formato localmente significativo do *fairRate* é conhecido como o *localFairRate.*

5.6.2.7 Medições das taxas de tráfego

Uma estação emprega estatísticas de taxas localmente colectadas para calcular taxas. São mantidas as seguintes taxas:

a) *addRate:* tráfego elegível para *fairness* adicionado pela estação local e limitado para qualquer estação no ringlet;

b) *addRateCongested:* tráfego elegível para *fairness* adicionado pela estação local e limitado para destinos além do ponto de congestionamento;

c) *fwRate*: tráfego elegível para *fairness* que transita a estação local e limitado para qualquer destino no *ringlet*;

d) *fwRateCongested*: tráfego elegível para *fairness* que transita a estação local e limitado para um destino que esteja para além do ponto de congestionamento;

e) *nrXmitRate*: todo o tráfego excepto tráfego transmitido de subclasseA0 (isto é, adicionado ou transitado) pela estação local.

Cada byte adicionado ou transitado pela estação é avaliado para determinar qual dos cinco contadores de taxa (*addRate, addRateCongested, fwRate, fwRateCongested* e *nrXmitRate)* devem ser incrementados.

Periodicamente, cada taxa é suavizada (*smoothed*) relativamente às anteriores medidas da taxa. Isto é feito calculando uma média ponderada do valor da taxa actual e o valor precedente da taxa suavizada. O coeficiente do peso associado com a média ponderada é um parâmetro configurado. Este método de suavização (*smoothing*) é conhecido como filtragem passa-baixo. As taxas suavizadas actualizadas são mantidas separadamente dos contadores de taxa, e são identificadas como *o lpAddRate, lpAddRateCongested, lpFwRate, lpFwRateCongested* e *lpNrXmitRate,* respectivamente. Os contadores de taxa são inalterados pela filtragem passa-baixo.

Depois de os contadores de taxa terem sido referenciados no cálculo de taxas suavizadas, é aplicada uma operação de envelhecimento (*aging*) aos contadores de taxa. Durante o período entre as operações de envelhecimento, conhecidas como um *agingInterval* cada contador de taxa mantém uma contagem de byte. O envelhecimento converte as contagens de byte nos valores que se aproximam assimptoticamente da taxa de dados, assumindo que a taxa de tráfego oferecido no *ringlet* está estável durante um determinado período de tempo.

Para além de manter os valores do contador de taxa como taxas, o envelhecimento tem o efeito de impedir o exceder do contador da taxa e de fazer a suavização do valor do contador da taxa relativamente aos valores precedentes deste contador. Uma contagem de taxa é *aged* multiplicando o seu valor pela expressão *(ageCoef-1)/ageCoef.* O *ageCoef* especifica os pesos relativos atribuídos (a) à mudança no valor do contador da taxa durante o *agingInterval* e (b) ao mais recente valor do contador *aged* da taxa na expiração do *agingInterval* precedente.

5.6.2.8 Cálculo das indicações de policiamento

As taxas permitidas de tráfego elegível para *fairness* e as taxas medidas de tráfego são usadas para manter indicações de policiando booleanas. A capacidade suficiente a jusante está disponível quando qualquer uma das seguintes condições é alcançada:

a) As estações a montante não têm necessidade de mais capacidade a jusante desde que o tráfego não esteja a acumular na STQ;

b) As estações a montante não estão *starved* para a capacidade a jusante e a estação não está inteiramente congestionada relativamente à ocupação da STQ.

As indicações de policiamento são avaliadas após um byte, ou um grupo de até 256 bytes, ter sido contado. Estas indicações são referenciadas pelo trajecto de dados no ajuste da indicação de *sendC.*

5.6.2.9 Ajuste do *fairRate*

Como anteriormente descrito, uma estação ajusta o seu *fairRate* em cada *agingInterval.* Executa um dos dois seguintes métodos de ajuste de taxa:

a) Agressivo: Fornece ajustes responsivos que favorecem a utilização da capacidade sobre a estabilidade da taxa;

b) Conservador: Fornece ajustes altamente "amortecidos" que favorecem a estabilidade da taxa sobre a utilização da capacidade.

O método agressivo é o mais simples dos dois métodos, e é descrito da seguinte maneira:

a) A estação está congestionada: O *fairRate* é ajustado ao *lpAddRate.* Se a estação for a cabeça de um domínio de congestionamento, o seu *fairRate* será propagado a todas as estações dentro do domínio de congestionamento;

b) A estação não está congestionada: Os *fairRate* não necessitam ser modificados. Uma estação que implementa o método agressivo não referencia o valor de *fairRate* quando no estado descongestionado.

O método conservador difere do método agressivo no facto de uma estação poder permanecer num estado congestionado depois das condições de congestionamento terem sido removidas. Isto fornece, na transição entre os estados congestionado e descongestionado, uma histerese que impede a oscilação da taxa. Na maioria dos casos, o método conservador requer que *o fairRate* não seja ajustado até que tenha decorrido o tempo suficiente para se assegurar que o efeito de qualquer ajuste precedente esteja efectivo. Este período de espera é conhecido como FRTT (fairness round trip time). O método conservador actua da seguinte forma:

a) Uma estação no estado descongestionado torna-se localmente congestionada: Ao *fairRate* é atribuído um valor inicial igual a uma parte justa ponderada do *unreservedRate;*

b) Uma estação no estado congestionado está localmente congestionada e decorreu um FRTT desde o último ajuste do *fairRate*: O *fairRate* é decrementado, ou reduzido por uma fracção do seu valor actual. A decrementação reduz gradualmente o congestionamento em contraste com o ajuste agressivo da taxa que reduz o *fairRate* directamente ao *lpAddRate;*

c) Uma estação no estado congestionado não está congestionada localmente e decorreu um FRTT desde o último ajuste do *fairRate*: O *fairRate* é incrementado, ou aumentado de uma fracção da diferença entre o seu valor actual e o *unreservedRate.* O incremento aumenta as taxas gradualmente em contraste com o ajuste agressivo da taxa que aumenta o *fairRate* directamente ao *unreservedRate;*

d) Uma estação está no estado congestionado e *o fairRate* está perto ou acima do *unreservedRate* (isto é, é inteiramente incrementado): O *fairRate* é ajustado ao *unreservedRate* e a estação entra no estado descongestionado. Isto está em contraste com o método agressivo onde o *fairRate* é ajustado ao *unreservedRate* assim que a estação já não esteja localmente congestionada;

e) Uma estação está seriamente congestionada devido à ocupação elevada da STQ: Ao *fairRate* é atribuída uma parte justa ponderada da soma do *lpAddRate* e *lpFwRate,* tendo em consideração que o valor é menor do que o *fairRate* actual. Esta condição não ocorre no método agressivo dado que o *fairRate* é imediatamente reduzido quando ocorre o congestionamento.

Estações que implementem os métodos agressivo ou conservador podem interoperar no *ringlet*. Ambos os métodos convergem para o valor de *fairRate* quando o tráfego oferecido nas estações no *ringlet* é constante.

5.6.2.10 Cálculo do *allowedRate*

O *allowedRate* especifica a taxa agregada à qual o tráfego elegível para *fairness* pode ser adicionado ao *ringlet* pela estação local. Como no exemplo do cálculo do *fairRate*, o método de calcular o *allowedRate* depende do método de ajuste de taxa implementado pela estação. No exemplo do ajuste agressivo de taxa, ao *allowedRate* é atribuído sempre a taxa máxima configurada (*maxAllowedRate)* da estação.

No exemplo do ajuste de taxa conservador, o *allowedRate* é calculado como se segue:

a) A estação está no estado descongestionado: O *allowedRate* é incrementado para *o maxAllowedRate.* Isto está em contraste com o método agressivo em que o *allowedRate* é ajustado directamente ao *maxAllowedRate;*

b) A estação está no estado congestionado: Ao *allowedRate* é atribuído o *fairRate* localmente calculado (ou o *maxAllowedRate* se o *fairRate* exceder o *maxAllowedRate).*

Ao usar o método conservador, a histerese associada com a transição desde o estado congestionado para o descongestionado impede a oscilação da taxa.

5.6.2.11 Cálculo do allowedRateCongested

O *allowedRateCongested* especifica a taxa à qual o tráfego elegível para *fairness,* destinado a estações para além do ponto de congestionamento, pode ser adicionado ao *ringlet*. Ao contrário dos cálculos do *fairRate* e do *allowedRate*, o cálculo do *allowedRateCongested* não depende do método de ajuste de taxa. Se a estação se encontrar dentro de um domínio de congestionamento, o *allowedRateCongested* é ajustado ao *fairRate* da estação da cabeça. Para as estações que se encontram a montante da cabeça, este valor é contido no anúncio da taxa recebida mais recentemente. Esta atribuição reflecte a regra que a uma estação que se encontra dentro de um domínio de congestionamento, não lhe é permitido adicionar tráfego elegível para *fairness* que transite o ponto de congestionamento a uma taxa superior à do *fairRate* da cabeça. Se a estação não se encontrar dentro de um domínio de congestionamento, o *allowedRateCongested* é incrementado, permitindo que as estações usem eficazmente a capacidade disponível até que o congestionamento seja encontrado.

O *allowedRateCongested* é acompanhado pelo *hopsToCongestion* que identifica a distância, em número de extensões (*hops*), entre a estação local e a cabeça do domínio de congestionamento em que a estação se encontra. O *hopsToCongestion* é calculado normalmente como a diferença entre o valor do *ttl* introduzido na estação de origem (MAX_STATIONS) e o valor do *ttl* da propaganda recebida após o decréscimo do *ttl*. Existe uma excepção quando a propaganda recebida foi circundada entre a origem e a estação local. Neste caso, o *hopsToCongestion* é ajustado ao número de extensões entre a estação local e o ponto de *wrap* a jusante, mantido pelo protocolo de protecção.

Ao *hopsToCongestion* é atribuído o valor MAX_STATIONS quando a estação local não se encontra dentro de um domínio de congestionamento.

5.6.3 Unidades do módulo de Fairness

As unidades do módulo de *Fairness,* à excepção do processamento de tramas de *fairness* recebidas, executam as suas actividades em intervalos fixos. O valor destes intervalos é sumariado na tabela 5.5.

Na arquitectura implementada foram desenvolvidos módulos de processamento para os métodos agressivo (*aggressive rate adjustment*) e conservador (*conservative rate adjustment*). A opção de escolha é configurada na própria estação RPR, através de operações de gestão sobre o sinal *conservativeMode* que selecciona o bloco de multiplexagem *agressiveOrConservativeRateAdjust* e todos os outros módulos que permitem processamento num dos modos (fig.5.13).

Intervalo	Processamento associado	Valor
Byte inter-arrival	Contagem de bytes de dados Indicações de policiamento	8/*lineRate*
agingInterval	Cálculo de taxa local	100µs para *lineRates* iguais ou superiores a 622 Mb/s e 400µs para *lineRates* abaixo de 622 Mb/s
advertisingInterval	Divulgação do *fairRate*	(tamanho da trama em bits)/(*lineRate* em b/s * *advertisementRatio* configurado)
reportingInterval	Broadcast do *fairRate*	*reportCoef * advertisingInterval*
activeWeightsInterval	Cálculo opcional de *activeWeights*	*activeWeightsCoef* (configurado) * *agingInterval*

Tabela 5.5 — Contadores para as operações de *fairness*

Foram implementadas as seguintes unidades do módulo de *fairness*:

a) PerByte: As actividades especificadas pela unidade PerByte são executadas para cada byte de dados que transita pela estação ou que fica disponível para adição ao *ringlet*. Estas actividades incluem o incremento de contadores de taxa e o ajuste de indicações de policiamento.

b) PerAgingInterval: A unidade de PerAgingInterval executa a filtragem passa-baixo dos contadores de taxa, a normalização das taxas *suavizadas*, o ajuste da taxa específico ao modo da estação (agressivo ou conservador), o cálculo do *allowedRateCongested,* o envio opcional de um MA_CONTROL.indication para o cliente, o envelhecimento dos contadores de taxa e o ajuste dos níveis alto e baixo da STQ.

c) AggressiveRateAdjust: A unidade de AggressiveRateAdjust é invocada pela unidade PerAgingInterval numa estação que usa o método agressivo para o cálculo da taxa. As actividades efectuadas por esta unidade incluem o cálculo e a normalização do *localFairRate.* O *allowedRate* mantém o seu valor inicial de *maxAllowedRate*. Quando se usa o método agressivo, o tráfego que não transita o ponto de congestionamento é restringido apenas pelo *maxAllowedRate*.

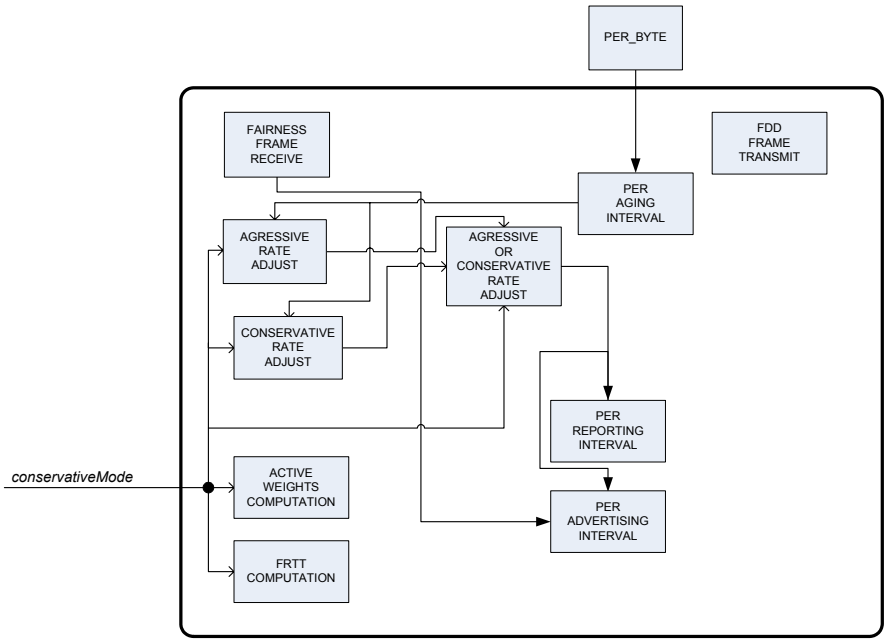


Figura 5.13 – Unidade de *Fairness*

d) ConservativeRateAdjust: A unidade de ConservativeRateAdjust é invocada pela unidade PerAgingInterval numa estação que usa o método conservador para o cálculo da taxa. As actividades efectuadas por esta unidade incluem o cálculo e a normalização do *localFairRate.*

e) PerAdvertisingInterval: Na expiração de cada *advertisingInterval* a estação executa uma das seguintes acções:

- Anuncia o seu *fairRate* localmente calculado;
- Propaga o *fairRate* recebido do vizinho a jusante;
- Anuncia o FULL_RATE ao vizinho a montante.

A acção reflecte a regra de que uma estação anuncia o *fairRate* associado com a cabeça do domínio de congestionamento em que se encontra, a menos que seja a cauda desse domínio. Uma estação que não se encontra dentro de um domínio de congestionamento ou que seja a cauda de um domínio de congestionamento mas não a cabeça de um outro (isto é, a montante) domínio de congestionamento, anuncia o FULL_RATE.

f) PerReportingInterval: Na expiração de um *reportingInterval* a estação local transmite um relatório da taxa a todas as estações no *ringlet*.

g) ActiveWeightsComputation: Esta unidade é aplicável somente ao ser usado o método conservador de ajuste de taxa e quando o *activeWeightsDetection* for TRUE.

Na expiração de um *activeWeightsInterval,* a estação local calcula a soma dos pesos das estações das quais uma trama elegível para *fairness* foi recebida durante o *activeWeightsInterval* decorrido. O *activeWeightsInterval* é configurado como um múltiplo inteiro do *agingInterval*. O valor de *activeWeights é* referenciado pela unidade de PerAgingInterval.

h) FairnessFrameReceive: Esta unidade faz o processamento de uma SCFF recebida (propaganda da taxa) ou de uma MCFF recebida (relatório da taxa). A informação contida numa propaganda é guardada para processamento na expiração do *advertisingInterval* seguinte. A informação *multi-choke* associada com um relatório da taxa é opcionalmente transferida ao cliente do MAC.

i) FddFrameTransmit: Ao se tornar na cauda de um domínio de congestionamento, uma estação emite periodicamente um par de tramas de controlo FDD (fairness differential delay) para a cabeça do domínio de congestionamento quando a cabeça do domínio de congestionamento continuar a ser uma estação que implementa o método conservador do cálculo da taxa. Às duas tramas são atribuídos o mesmo número de sequência mas uma é transmitida com uma classe de serviço de subclasseA0 e a outra com uma classe de serviço de classeC. A trama FDD de classeA é emitida antes da FDD de classeC. A trama FDD de classeC é emitida tão depressa quanto possível após a trama FDD de classeA (dentro dos limites das regras de transmissão do trajecto de dados do MAC). Uma estação de cabeça que recebe tramas FDD, calcula o valor FDD. A estação de cabeça considera válida uma trama FDD somente quando chegarem ambas as tramas FDD de classeA e de classeC, transportando o mesmo número de sequência.

j) FrttComputation: Esta unidade descreve o cálculo do valor do FRTT (fairness round trip time) que é usado pela unidade ConservativeRateAdjust. O FRTT é somente calculado por uma estação que efectue o cálculo conservador da taxa e seja a cabeça de um domínio de congestionamento. O valor do FRTT é recalculado quando um par válido de tramas FDD da estação da cauda tenha sido recebido pela estação da cabeça. Esse par de FDDs é emitido pela cauda a cada *fddInterval*. O valor de FRTT é suavizado sobre os cálculos precedentes de *ageCoef*. A cabeça faz o *flush* dos resultados precedentes (isto é, não os usa para finalidades de *smooth*) se a cauda do domínio de congestionamento mudar.

A estação da cabeça estima o FRTT referenciando os FDD e LRTT medidos e *suavizados*. O FRTT é a soma de FDD com LRTT, arredondada ao múltiplo mais próximo do *agingInterval.*

$$FRTT = FDD + LRTT \qquad \text{(Eq. 5.2)}$$

O FDD é uma medida da diferença em atraso entre os trajectos de classeA e de classeC da estação da cauda à estação da cabeça. O LRTT é uma medida do atraso da ligação pelas tramas de classeA desde a cabeça até à cauda do domínio de congestionamento e de volta. Os valores do LRTT são mantidos na base de dados da topologia.

5.7 Topologia e Protecção

A entidade de Topologia e Protecção tem as seguintes características:

a) Responsividade:

1) Restauração de tráfego protegido abaixo de 50ms;

2) Disseminação rápida pelo anel, das mudanças no estado da protecção;

b) Robustez:

1) Suporte de uma hierarquia de protecção detalhada;

2) Suporte de adição e remoção dinâmicas, de estações de/para o anel;

3) Tolerância no controlo da perda de tramas;

4) Operação sem qualquer estação *master* no anel;

5) Validação da topologia, incluindo a detecção de falha de ligação entre estações;

6) Operação independente e na ausência de quaisquer sistemas de gestão;

7) Garantia de que todas as estações no anel convergem para uma imagem uniforme e actualizada da topologia;

8) Suporte do conteúdo de contexto, que permite a garantia de que não haja novamente a duplicação e requisição de tráfego de modalidade estrita;

c) Flexibilidade:

1) Suporte de modos de comutação de protecção *revertive* e *non-revertive;*

2) Suporte das topologias em anel fechado e aberto;

3) Escalabilidade de uma até 255 estações;

4) Meios de partilhar a informação adicional entre estações;

d) Eficiência:

1) Necessita de um tráfego insignificante pelo anel;

2) Consome um tempo mínimo de execução do *software*;

3) Requer pouco equipamento de *hardware*.

5.7.1 Sumário do protocolo

O protocolo de descoberta da topologia do RPR fornece a cada estação no anel o conhecimento do número, das potencialidades básicas e do arranjo das outras estações no anel. Esta colecção de informação é referida como a base de dados da topologia. Quando os conteúdos desta base de dados cessarem de ser alterados, então a topologia está estável.

A informação crítica da base de dados da topologia é derivada da informação fornecida pelas tramas de controlo da topologia e da protecção (TP) recebidas em cada *ringlet*. As tramas TP incluem o estado de protecção da ligação, o estado da estação de fronteira e a informação de configuração da protecção. As tramas TP são emitidas periodicamente e quando estimuladas por mudanças do estado de protecção.

As tramas de controlo de descoberta de atributo (ATD) comunicam informação menos crítica em termos de tempo, para ser armazenada dentro da base de dados de topologia. As tramas ATD

incluem preferências normalizadas da estação e também informação opcional específica de uma organização.

O protocolo de protecção usa a informação do estado da protecção da ligação, armazenada na base de dados da topologia, para determinar se as suas extensões podem ser usadas para transportar tramas de dados. Esta informação do estado da estação de fronteira é fornecida à sub-camada de trajecto de dados do MAC, que faz o direccionamento e o *wrapping* do tráfego protegido quando necessário. O protocolo de protecção usa os parâmetros de configuração, tais como os tempos de *hold-off* e *wait-to-restore,* e uma hierarquia de estados derivados da monitorização das ligações e de comandos do operador da rede.

Um anel RPR providencia protecção para tramas de dados relaxadas dentro da combinação de tempo de detecção e tempo de restauração, e para tramas de dados estritas dentro da combinação de tempo de detecção, tempo de restauração e tempo de estabilização da topologia.

Todas as estações são notificadas das mudanças no estado de protecção dentro de um tempo aproximado a uma volta em torno do anel (RRTT) mais os atrasos de processamento.

A protecção de tráfego de modo relaxado ocorre na notificação de uma mudança na topologia, onde a protecção de tráfego de modo estrito ocorre após a validação da topologia.

O tempo de convergência da topologia consiste no tempo de notificação (geralmente um RRTT) e o tempo de estabilização (40ms por defeito).

Uma estação que implementa o método conservador no cálculo das taxas de *fairness* depende das medições do tempo de uma volta fechada em torno do anel (LRTT). O LRTT é medido enviando tramas de requisição para cada estação no *ringlet*. As estações respondem a cada pedido enviando tramas de resposta. Os valores do LRTT são calculados pelo conteúdo e tempo de chegada das respostas, e armazenados na base de dados de topologia.

A validação da topologia é executada após a estabilização da mesma. Esta validação relata os defeitos da topologia à unidade de OAM. A topologia deve estar válida antes que o conteúdo de contexto possa ser apagado.

A definição de estação de fronteira quer dizer que a mesma está ligada a uma extensão que não está a ser usada para tráfego de dados devido ao seu estado de protecção. A detecção de uma fronteira num anel torna este anel num anel aberto e resulta na protecção do tráfego. Apenas as tramas TP continuam a ser transmitidas entre estações vizinhas através de uma fronteira para monitorizar a disponibilidade da estação e da extensão.

O modo opcional de *passthrough* permite que as estações sejam incorporadas ou retiradas do anel sem desligar as fibras (por exemplo quando da detecção de condições de falha internas) e sem depoletar um evento de falha de sinal. O *passthrough* permite que uma estação se retire do anel enquanto mantém uma topologia em anel fechado, evitando uma comutação de protecção.

A transmissão periódica de tramas TP permite a detecção de uma estação a entrar em *passthrough*. Quando qualquer estação aparentar ter mudado de localização, a transmissão de tramas TP é despoletada para facilitar a rápida redescoberta da topologia e a restauração do tráfego de modo estrito.

5.7.1.1 Manutenção da base de dados de topologia

Inicialmente uma estação cria uma base de dados de topologia consistindo apenas nela própria, de seguida divulga tramas TP e ATD em ambos os *ringlets*. Cada estação usa as tramas TP recebidas para actualizar a base de dados da topologia.

Quando uma estação é adicionada a um anel, as suas estações vizinhas detectam-na como sendo uma nova estação vizinha ao receberem a sua trama TP (indicando que a trama percorreu exactamente uma extensão) e um *sa* (*source* MAC *address*) diferente do observado da estação vizinha anterior. Da mesma forma, quando uma estação deixa de estar no estado de *passthrough*, comporta-se como sendo uma nova estação inicializada.

Uma nova estação no anel transmite tramas TP em ambos os *ringlets*. As outras estações detectam esta nova estação ao receberem a sua trama TP, e cada uma responde com tramas TP para ambos os *ringlets*. As tramas TP recebidas são usadas para actualizar a base de dados da topologia.

Quando é determinado que uma extensão é de fronteira, a informação na base de dados de topologia para além dessa fronteira (exceptuando informação para a estação vizinha) é marcada como inválida, e as tramas TP são enviadas para relatar a existência dessa fronteira. Isto inclui o caso de uma estação ser removida de um anel.

Quando uma ou mais estações entram em *passthrough*, elas efectivamente desaparecem do anel. O modo de *passthrough* é detectado quando uma estação recebe uma trama TP com um *sa* (*source* MAC *address*) diferente do esperado para a contagem de ligações (*hop*) recebida. Cada estação transmite tramas TP quando detecta um evento de *passthrough*.

5.7.1.2 Conteúdo de Contexto

A finalidade do conteúdo de contexto é a de prevenir a duplicação e reordenamento de tráfego de modo estrito durante alterações da topologia. O conteúdo de contexto entra em *passthrough*, quando se altera o estado de uma fronteira num anel *steering*, ou quando uma fronteira é removida num anel *wrapping*. A detecção de que o *checksum* de uma estação de fronteira não coincide com o *checksum* da estação vizinha, não despoleta o conteúdo de contexto.

O conteúdo de contexto não é abandonado até que seja determinado que a topologia está estável e válida, e que o *checksum* da base de dados de topologia local seja determinado que coincide com os das suas estações vizinhas alcançáveis.

Na ausência de um evento de *passthrough*, a duração do conteúdo de contexto é dominada pela estabilização da topologia. Caso contrário, a duração é significativamente influenciada por um maior tempo de descoberta da topologia.

O *checksum* da topologia está contido nas tramas de *checksum* de topologia (TC). As tramas TC são enviadas periodicamente e quando os seus conteúdos são alterados.

5.7.1.3 Hierarquia de protecção

O MAC suporta as seguintes hierarquias de protecção, listadas abaixo na ordem de severidade decrescente.

a) Forced Switch (FS) — Directiva de gestão que força uma ligação a ser desactivada;

b) Signal Failure (SF) — Perda de sinal que faz desactivar a ligação;

c) Signal Degrade (SD) — Degradação do sinal que pode desactivar a ligação;

d) Manual Switch (MS) — Directiva de gestão que pode desactivar a ligação;

e) Wait To Restore (WTR) — Um temporizador que melhora a estabilidade na presença de falhas transitórias;

f) IDLE – nenhuma das condições anteriores.

5.7.2 Descoberta da topologia e funções de protecção

As funções da topologia e da protecção para uma única estação são mostradas na figura 5.14. Estas funções gerem a transmissão e recepção de tramas relacionadas com a topologia, a recepção de pedidos administrativos e o processamento associado a estas funções.

ReceiveMonitor: O par de máquinas de estado ReceiveMonitor monitoriza o estado da extensão. Cada uma das extensões de uma estação tem uma máquina de estados ReceiveMonitor.

TopologyControl: A máquina de estados TopologyControl processa pedidos de configuração de gestão e invoca as máquinas de estados ParseTpFrame e ProtectionUpdate.

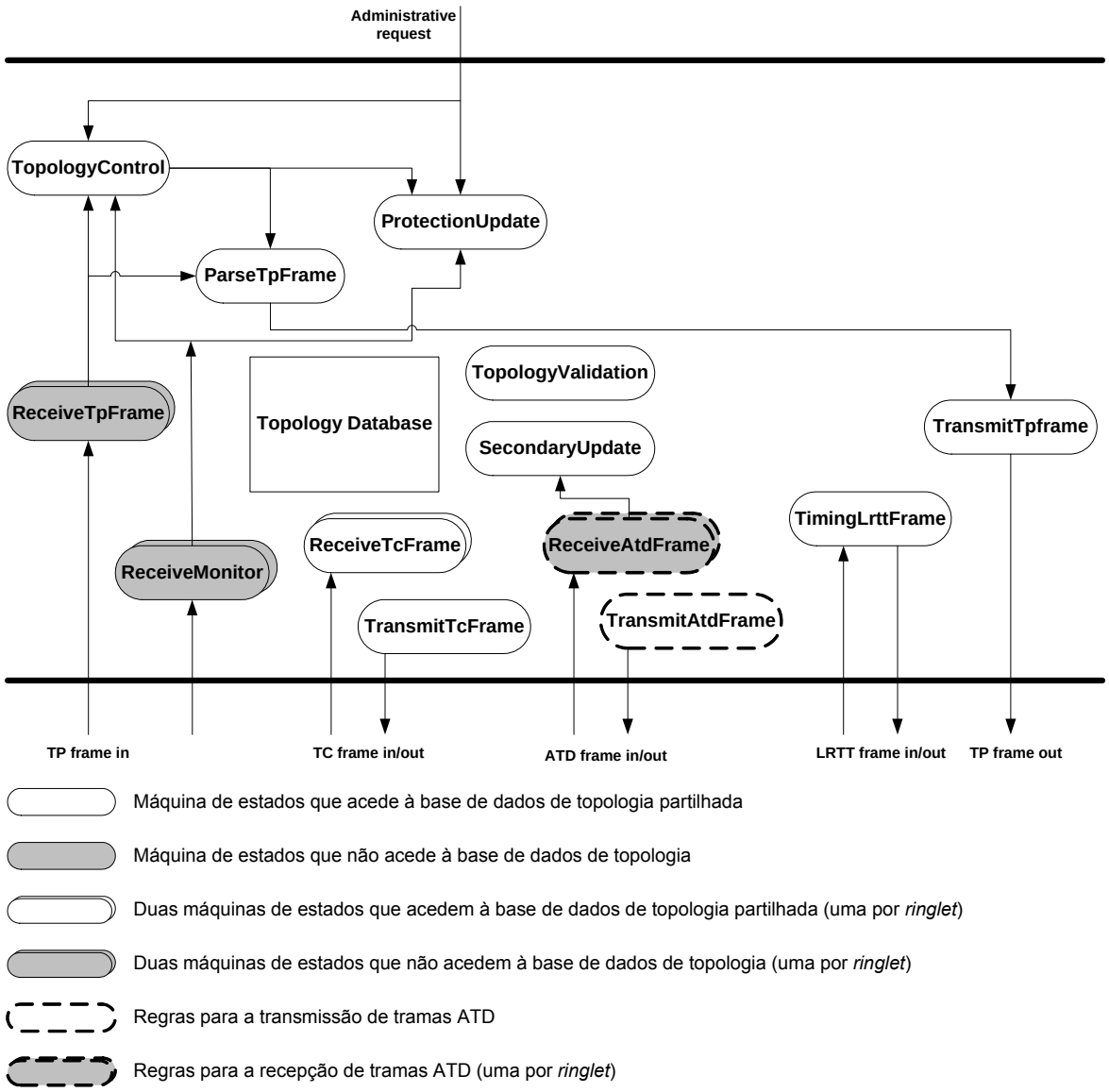


Figura 5.14 – Relações da topologia e protecção

ParseTpFrame: A máquina de estados ParseTpFrame processa as tramas TP recebidas do anel, para actualizar a base de dados de topologia.

ProtectionUpdate: A máquina de estados ProtectionUpdate é invocada pela máquina de estados TopologyControl, e actua numa extensão especificada. Esta máquina de estados usa a informação contida na base de dados de topologia para determinar o estado de protecção das suas extensão e fronteira, antes de actualizar a base de dados de topologia. Actua para ambos os tipos de protecção *steering* e *wrapping*. Actua também na extensão (este ou oeste) de uma estação para a qual é invocada.

TopologyValidation: A máquina de estados TopologyValidation determina quando é que a topologia está estável e válida.

TransmitTpFrame: A máquina de estados TransmitTpFrame é usada para transmitir tramas TP e determinar quando estas são enviadas.

ReceiveTpFrame: A máquina de estados ReceiveTpFrame recebe e pré-processa as tramas TP antes de as passar para a máquina de estados TopologyControl. Uma máquina de estados actua em cada uma das extensões de uma estação.

TransmitTcFrame: A máquina de estados TransmitTcFrame é usada para transmitir tramas TC e determinar quando estas são enviadas.

ReceiveTcFrame: A máquina de estados ReceiveTcFrame recebe tramas TC. Uma destas máquinas de estados actua em cada extensão de uma estação.

TransmitAtdFrame: A máquina de estados TransmitAtdFrame gera tramas segundo regras que descrevem as acções tomadas para transmitir tramas ATD.

ReceiveAtdFrame: A máquina de estados ReceiveAtdFrame recebe tramas ATD e contém regras que descrevem as acções tomadas para processar essas tramas.

SecondaryUpdate: A máquina de estados SecondaryUpdate actualiza os endereços secundários do MAC, baseado nas entradas ATT das tramas ATD recebidas, ou por directivas de pedidos de gestão.

TimingLrttFrame: A máquina de estados TimingLrttFrame gera o pedido de tramas LRTT e processa as tramas de resposta recebidas de volta, com o objectivo de calibrar o LRTT do anel.

5.8 Unidade OAM

5.8.1 Unidade de requisição de OAM

A arquitectura implementada inclui uma unidade de requisição de uma trama OAM que implementa as funções necessárias para o processamento da requisição da trama OAM pela estação de origem.

5.8.2 Recepção OAM

As estações usarão a função de MA_CONTROL.indication para enviar para o cliente o código e os operandos da trama OAM recebida. A sub-camada de controlo do MAC usa o MA_CONTROL.indication para transferir as indicações de estado do OAM ao cliente.

A arquitectura implementada inclui também uma unidade de recepção da trama OAM que implementa as funções necessárias para o processamento dessa trama OAM.

5.8.3 Manuseamento das tramas OAM durante falhas

As tramas OAM podem ser emitidas no *ringlet* por defeito ou num *ringlet* especificado, e podem ser protegidas ou desprotegidas.

As tramas OAM não serão transmitidas sobre ligações falhadas mesmo que a falha seja unidireccional.

5.8.4 Monitorização de desempenho

As estações podem acumular os parâmetros de desempenho relacionados com o RPR para permitir a detecção de falhas antes que seja detectada uma falha total.

scffErrors: Representa o número de SCFFs errados. Um SCFF errado é definido como um SCFF com qualquer paridade errada ou má verificação do *fcs*.

erroredSeconds: Conta os segundos durante os quais o *scffErrors* foi mudado.

severelyErroredSeconds: Conta os segundos durante o aumento do *scffErrors* de pelo menos *sesThreshold* contagens.

unavailableSeconds: Contagens em intervalos de 1s para os quais o serviço RPR está indisponível.

O serviço do RPR torna-se indisponível no início de dez severelyErroredSeconds contíguos. Os dez *severelyErroredSeconds* são incluídos no tempo de indisponibilidade.

Uma vez indisponível, o serviço do RPR torna-se disponível no início de 10s contíguos com nenhum *severelyErroredSeconds.* Os 10s com nenhum *severelyErroredSeconds* são excluídos do tempo de indisponibilidade.

As contagens de *scffErrors, erroredSeconds*, *e* de *severelyErroredSeconds* são inibidas durante a indisponibilidade.

6 Aspectos de implementação

Os circuitos electrónicos para implementar os módulos constituintes do MAC do RPR, foram desenvolvidos tende como objectivo a sua implementação num dispositivo de lógica programável FPGA – Field Programmable Gate Array.

Devido à complexidade do sistema, foi estabelecida inicialmente uma abordagem de desenvolvimento baseada numa metodologia *top-down* e posteriormente enriquecida com contribuição *bottom-up,* através da descrição em SystemC [21] comportamental e ao nível da lógica de transferência de registos (RTL). Ao longo do desenvolvimento teve de ser iterada esta abordagem mista envolvendo ambas as metodologias, pois houve determinados pressupostos que tiveram de ser resolvidos e que a utilização de somente uma das referidas metodologias não o permitia. O desenvolvimento em SystemC foi suportado por ferramentas de desenvolvimento da empresa Synopsys [17] [18] [25] [35].

O projecto engloba os módulos da camada MAC do RPR, descritos no capítulo anterior. Isto inclui as sub-camadas de trajecto de dados e de controlo do MAC, contendo esta última, os módulos de *Fairness*, Topologia e Protecção, e OAM.

Os módulos do MAC do RPR necessitam de elevada capacidade de processamento para cálculo de estatísticas; para implementação de funções necessárias nos diversos módulos, que requerem acessos à base de dados armazenada em memória externa; e também para controlo de alguns periféricos que são necessários a todo o sistema. A escolha do dispositivo FPGA também teve de ser feita tende em consideração estes pressupostos, e o objectivo foi um FPGA com grande capacidade de armazenamento de memória e também capacidade de processamento embebida no próprio circuito integrado do FPGA.

Mas como referido no capítulo anterior, o sistema tem outros componentes que não o FPGA. Efectivamente, os módulos integrantes do sistema, foram desenvolvidos tendo como objectivo a implementação num FPGA. No entanto, este tipo de dispositivos ainda não tem disponível uma suficiente capacidade de memória para armazenamento de um relativamente elevado número de dados, portanto foi desenvolvida a interface e controlo necessários para usar memória externa para armazenamento da base de dados de Topologia e Protecção. Além disso, estratégias típicas de controlo requerem processamento complexo pelo que existem interfaces para processadores externos.

As simulações do sistema foram desenvolvidas da seguinte forma:

1. usando o ambiente em SystemC;
2. fazendo a co-simulação de vectores de teste em SystemC com os módulos do MAC do RPR compilados de SystemC para Verilog [22];
3. e fazendo testes com o código sintetizado para um FPGA da Xilinx [26], permitindo comparações directas entre as simulações e os resultados práticos.

Toda a arquitectura das unidades descritas nas secções seguintes, foi implementada em SystemC e em Verilog. Cada um dos módulos foi implementado usando máquinas de estados com processamento de dados integrado nas mesmas e descritas em SystemC. Foram também usadas memórias descritas em Verilog e/ou SystemC, para armazenamento de variáveis temporárias que seriam usadas como contadores, ou somente como variáveis auxiliares. Estas memórias foram implementadas de forma a serem sintetizadas como RAM distribuída das macro-células de um FPGA (no nosso caso nos CLBs de um FPGA Xilinx).

Os referidos módulos foram englobados dentro de unidades para teste e simulados independentemente uns dos outros, usando vectores de teste descritos em SystemC sintetizável, e específicos para cada um destes módulos. Para esta fase foi usado o programa CoCentricSystemStudio [17], da empresa Synopsys, segundo uma descrição orientada por objectos.

Posteriormente nestas unidades para teste, foi feita a co-simulação das máquinas de estados compiladas para Verilog, com os mesmos vectores de teste em SystemC, e com as memórias

referidas acima e descritas também em Verilog. Isto para comparar os resultados das simulações em CAD, nos ambientes SystemC e Verilog.

O uso de SystemC sintetizável nos blocos de teste permite que seja feito o teste num FPGA, integrando tanto o bloco a ser testado como também o bloco que gera os vectores de teste, necessitando apenas de um relógio externo para accionar os processos do circuito. Sendo assim, cada módulo de teste foi implementado num pequeno FPGA para teste e foi feita a comparação entre as simulações no CAD e os resultados práticos em *hardware*.

6.1 Ferramentas utilizadas

A linguagem SystemC é definida como uma biblioteca de classes vocacionada para a descrição de sistemas, tanto *software* como *hardware*, construída sobre a linguagem C/C++. Neste trabalho foi usada a versão 2.0.1 da biblioteca de classes SystemC [32], conforme disponível no *site* da Open SystemC Initiative (OSCI). Esta versão está integrada no programa de desenvolvimento CoCentricSystemStudio da empresa Synopsys.

O SystemC pode ser usado para a descrição de *hardware* em diversos níveis de abstracção. É possível utilizar descrições em SystemC como descrição de entrada para fluxos automatizados de geração de *hardware*, desde que se restrinja as construções em SystemC a um subconjunto (dito sintetizável). Este subconjunto é denominado de SystemC de nível de transferência entre registos ou SystemC RTL.

A ferramenta principal para processar SystemC RTL usada neste trabalho é o CoCentric SystemC Compiler da empresa Synopsys [18], que permite implementar a síntese lógica de SystemC RTL para formatos EDIF ou HDL, ou apenas traduzir SystemC RTL para código HDL RTL (nas linguagens Verilog ou VHDL [23]).

Como algumas das técnicas para gerar *hardware* a partir de SystemC envolvem a passagem por código HDL intermédio, além do CoCentric SystemC Compiler foram também usadas as ferramentas de síntese lógica XST da Xilinx (integrada no ambiente ISE [26] da Xilinx), FPGA Compiler II [24] da Synopsys e Synplify Pro [34] da Synplicity, obtendo com esta última ferramenta, melhores resultados do que com as outras duas. Para a prototipagem, baseada em FPGAs Xilinx, de *hardware* a partir de código Verilog ou ficheiros EDIF foi também usado o ambiente da ferramenta ISE da Xilinx.

6.2 Projecto, validação e prototipagem de módulos descritos em SystemC

Existem algumas directivas de codificação para SystemC ao nível RTL, visando obter descrições passíveis de síntese automatizada. Existe um subconjunto do SystemC definido como uma norma para viabilizar a síntese automática em *hardware*. O documento que define este subconjunto é a referência [33].

Existem referências amplas e completas tendo como objectivo a geração de SystemC RTL sintetizável, tais como o guia para a geração de SystemC RTL sintetizável, da Synopsys [33], e o guia de modelização RTL da ferramenta CoCentric SystemC Compiler, também da Synopsys [34].

Neste trabalho emprega-se uma grande parte dos tipos sintetizáveis. Por exemplo, a recomendação em [33] do uso do tipo C++ nativo *bool* para modelar valores e sinais que transportam exactamente um bit de informação. O tipo *sc_bit* é suportado mas não recomendado. O tipo *sc_logic* serve para modelizar uma ligação que suporta quatro estados (0, 1, X e Z), o que permite modelizar conflitos de valores e ligações em estado de alta impedância, um subconjunto dos estados suportados pelos tipos *std_logic* e *std_logic_vector* em VHDL. Os tipos *sc_bv* e *sc_lv* modelam vectores de comprimento arbitrário de *sc_bit* e *sc_logic*, respectivamente. No entanto o tipo *sc_uint* deve ser usado sempre que possível ao invés de *sc_bv*.

Outros tipos C++/SystemC suportados [33] servem para modelizar representações sintetizáveis de:

- números naturais de precisão fixa;

- números inteiros de precisão fixa;
- caracteres;
- tipos enumerados, para uso, por exemplo em codificação simbólica de estados em máquinas de estados finitas;
- agregados de tipos sintetizáveis.

Além do subconjunto de construções e dos tipos sintetizáveis, a referência [33] define o conjunto de operadores a usar com estes para produzir código sintetizável. Estas recomendações foram estritamente seguidas.

6.3 Prototipagem em FPGA

Quando o objectivo de uma prototipagem de um sistema é um FPGA, é necessário ter em atenção alguns critérios adicionais para ser feita a partição do sistema.

Para se prototipar um sistema real usando FPGAs, é por vezes necessário fazer a partição da lógica do sistema por diversos FPGAs, ou se se usar apenas um FPGA é necessário desenvolver código eficiente que permita, baseado nos recursos específicos disponíveis no FPGA, utilizar eficientemente os mesmos. O FPGA usado foi um Virtex 2 Pro da empresa Xilinx (XC2VP70) e neste circuito em particular, os referidos recursos do FPGA, são: o número de macro-células/CLBs (74448); número de pinos de entrada/saída (996); pinos de entrada/saída especiais como por exemplo LVDS ou PCI; número de *global clock buffers* (8)*;* número de blocos de memória RAM embebidos e suas respectivas dimensões (328 blocos de 18Kb); dois micro-processadores embebidos (PowerPC); blocos de multiplicadores embebidos (328 multiplicadores de 18x18bit); e número de células de lógica (74448).

A prototipagem de *hardware* em FPGAs pode ser implementada por diferentes processos. Usando SystemC como linguagem de partida para a prototipagem de *hardware*, existem vários fluxos possíveis.

Primeiro, pode-se partir de SystemC e implementar a síntese lógica usando ferramentas Synopsys ou Synplicity, seguido de síntese física com ferramentas específicas do vendedor de FPGAs. Pode-se também empregar ferramentas Synopsys para a síntese física, desde que dispondo das bibliotecas específicas do vendedor de FPGAs instaladas no ambiente Synopsys. Outra possibilidade é usar ferramentas Synopsys apenas para traduzir o código SystemC para HDL independente do dispositivo, entrando a partir daí com o HDL gerado num ambiente de CAD específico, tal como o ISE da Xilinx.

Neste trabalho foi usado o fluxo de prototipagem Synopsys Design Capture → Synopsys Design Compiler → Synplicity Synplify Pro → Xilinx ISE. Este fluxo foi escolhido, após diferentes experiências com as ferramentas disponíveis, por fornecer a síntese mais fiável e eficiente.

A Figura 6.1 ilustra o fluxo para implementação em FPGA, com os exemplos das ferramentas empregues em cada passo. Entre cada dois passos consecutivos, o formato de comunicação de informações é via uma linguagem textual, seja de alto nível como HDL e/ou SystemC, ou seja de baixo nível como EDIF.

Tal como referido anteriormente, neste trabalho partiu-se de uma descrição e co-simulação em SystemC e em Verilog usando o programa CoCentric System Studio da Synopsys; posteriormente foi utilizada a tradução automática dos módulos codificados em SystemC RTL para Verilog utilizando a ferramenta CoCentric SystemC Compiler também da Synopsys.

Como as ferramentas FPGA Compiler II e Synplify Pro não suportam SystemC directamente, é necessária a tradução inicial de SystemC, utilizando a ferramenta SystemC Compiler. Seguidamente, é necessário entrar nas ferramentas FPGA Compiler II ou Synplify Pro para proceder à síntese física, seguida da geração do *bitstream* de configuração (ficheiro EDIF). A síntese lógica foi feita usando a ferramenta Synplify Pro da Synplicity, utilizando directamente

ficheiros HDL como entrada nesta ferramenta de síntese lógica, para a prototipagem em FPGAs Xilinx ou outras.

Finalmente, deve-se gerar o arquivo de configuração (*bitstream*) do FPGA através da ferramenta Xilinx ISE, conforme já referido acima.

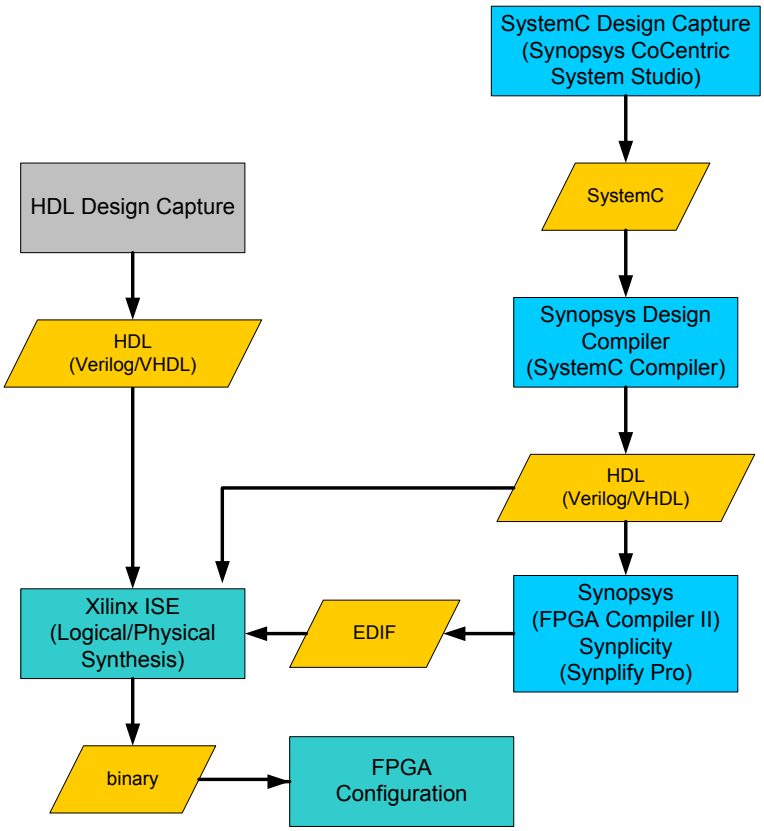


Figura 6.1 – Fluxos de projecto usados para prototipagem em FPGA

Um outro fluxo possível seria utilizando as ferramentas SystemC Compiler e FPGA Compiler II, da Synopsys, no entanto os resultados obtidos foram inferiores relativamente aos do fluxo de síntese lógica anterior, pois geram muito mais lógica o que significa uma maior ocupação da FPGA.

6.4 Validação

A simulação em CAD e o teste dessa simulação na implementação *hardware* em FPGA foram elaboradas segundo o fluxo ilustrado na figura 6.2., de forma a validar o projecto.

Partindo inicialmente da descrição em SystemC usando o ambiente Design Center do CAD CoCentric System Studio da Synopsys, foi elaborada a estrutura de topo de um sistema e o desenvolvimento dos módulos constituintes do mesmo.

Posteriormente foi feita a simulação do sistema pela ferramenta Simulator. Numa abordagem totalmente baseada em SystemC, é feita a simulação dentro do CoCentric usando blocos de *testbench,* com vectores de teste também desenvolvidos em SystemC, que geram sinais para os blocos (DUT) que são testados. Os sinais resultantes do DUT são visualizados e analisados através da ferramenta VirSim integrada no ambiente CoCentric.

Após a obtenção de um projecto e simulação coerentes, o código em SystemC é compilado para código HDL através da ferramenta SystemC Compiler. O código resultante é integrado novamente

no Design Center e co-simulado no ambiente SystemC, com os vectores de teste originais em SystemC.

Dentro da co-simulação no ambiente do Design Center, é percorrido todo o processo idêntico à simulação normal em SystemC. Após a obtenção de uma simulação coerente, o código HDL é sintetizado para FPGA, como já foi descrito atrás, simulado consoante os mesmos vectores de teste originais e analisado num Analisador Lógico. O processo termina se o resultado das simulações em FPGA forem coerentes com os das simulações em CAD, caso contrário é reiniciado o processo a partir do código HDL originalmente obtido através do Design Compiler.

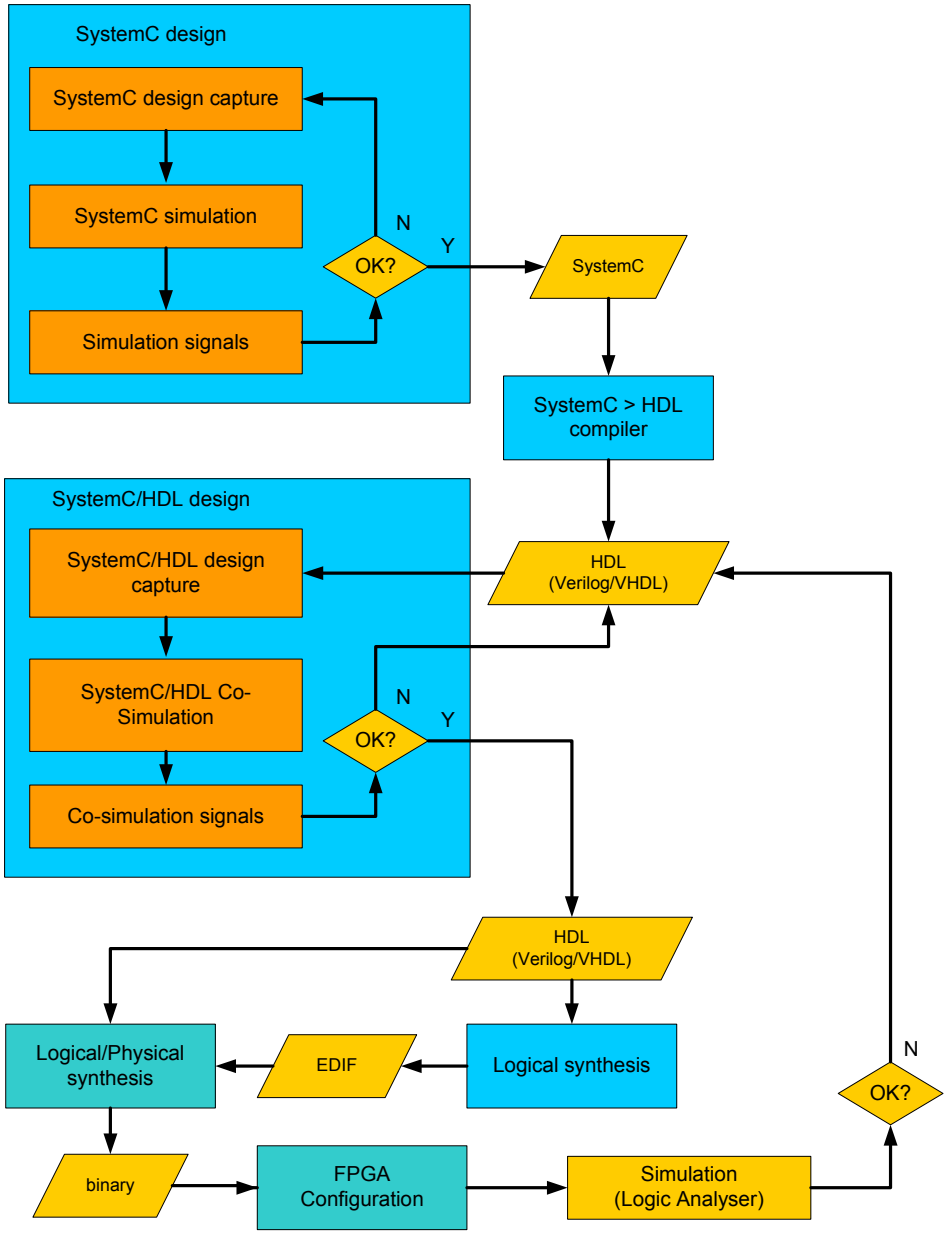


Figura 6.2 – Fluxo de simulação em CAD e em FPGA

O código em SystemC é passível de ser simulado extensivamente e de uma forma simples. No entanto a co-simulação de SystemC e HDL (Verilog/VHDL) é extremamente importante para todo o fluxo de desenvolvimento de um sistema, embora mais morosa. Como referido, o desenvolvimento deste sistema contém uma mistura de códigos em SystemC e em HDL. Muitos

dos módulos contêm blocos funcionais em código HDL pertencentes a fabricantes de FPGAs ou outros, tais como blocos de memória, FIFOs, etc.

Actualmente não existe um modo automático para verificar a equivalência entre códigos SystemC e HDL. Para se garantir que o código HDL gerado implementa exactamente o código desenvolvido em SystemC, é necessário co-simular o módulo DUT (código HDL gerado a partir de SystemC) dentro do ambiente SystemC (figura 6.3).

A ferramenta CoSim (ambiente CoCentric) de co-simulação entre SystemC e HDL gera um conjunto de interfaces que permitem ao ambiente SystemC, comunicar com o simulador HDL. Durante a simulação, o CoSim sincroniza o *kernel* do SystemC com o simulador HDL (VCS) e troca dados entre os dois ambientes.

Há dois modos de processar uma co-simulação. Um modo de importação onde é usado o VCS para a co-simulação, em que o CoSim gera as interfaces de comunicação para importar um DUT em HDL para dentro de um ambiente SystemC. A simulação em SystemC é a mestre e a simulação em HDL é a escrava. Por outro lado existe o modo de exportação, em que o CoSim gera as interfaces de comunicação para exportar um DUT em SystemC para dentro de um ambiente HDL. Neste caso a simulação HDL é a mestre e a simulação em SystemC é a escrava.

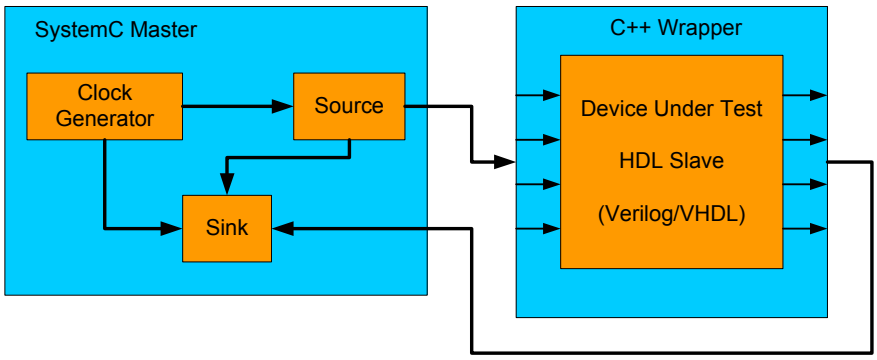


Figura 6.3 – Ambiente de co-simulação em SystemC com HDL

O modo usado foi o de importação devido ao facto de inicialmente todo o projecto ter sido elaborado em ambiente SystemC e também às potencialidades de simulação em SystemC, inerentes ao ambiente da ferramenta CoCentric System Studio da empresa Synopsys.

6.5 Desenvolvimento das unidades do RPR

Nesta secção são descritos os aspectos de implementação das unidades constituintes do MAC RPR. Para cada unidade é feita uma descrição sumária do seu funcionamento. A descrição exaustiva de todos os blocos seria fastidiosa, no âmbito desta dissertação.

A descrição da unidade de Rate Control é seguida de um exemplo prático de desenvolvimento de um módulo constituinte dessa unidade. O exemplo parte da descrição elaborada em SystemC, eventualmente com integração de blocos em Verilog (*cores* OEM ou desenvolvidos), simulação da unidade, compilação para Verilog, co-simulação da unidade, síntese lógica e física para FPGA, e finalmente a simulação real no dispositivo FPGA.

6.5.1 Operações de Recepção e de Transmissão

Os circuitos das unidades de operação de recepção e de transmissão que estão descritos no capítulo cinco, estão ilustrados de uma forma simplificada nas figuras 6.4 e 6.5, respectivamente.

Cada uma das unidades funciona com interfaces de 32 bits em paralelo tanto para a camada da PHY como para a camada do cliente.

As estações e redes RPR podem ser implementadas tanto com PHYs síncronas como assíncronas. Esta implementação é para uma rede síncrona, onde o relógio de transmissão para cada estação está síncrono e é dependente de uma fonte de relógio comum que pode ser recuperada das tramas de dados de recepção ou providenciada por uma fonte de relógio externa e, neste caso, a taxa de dados originada de cada estação é exactamente idêntica à taxa de dados recebida dessa estação.

Nesta implementação há um relógio para a unidade de recepção e outro relógio para a unidade de transmissão. Estes dois relógios funcionam com a mesma frequência de 31,25MHz, e são dependentes um do outro.

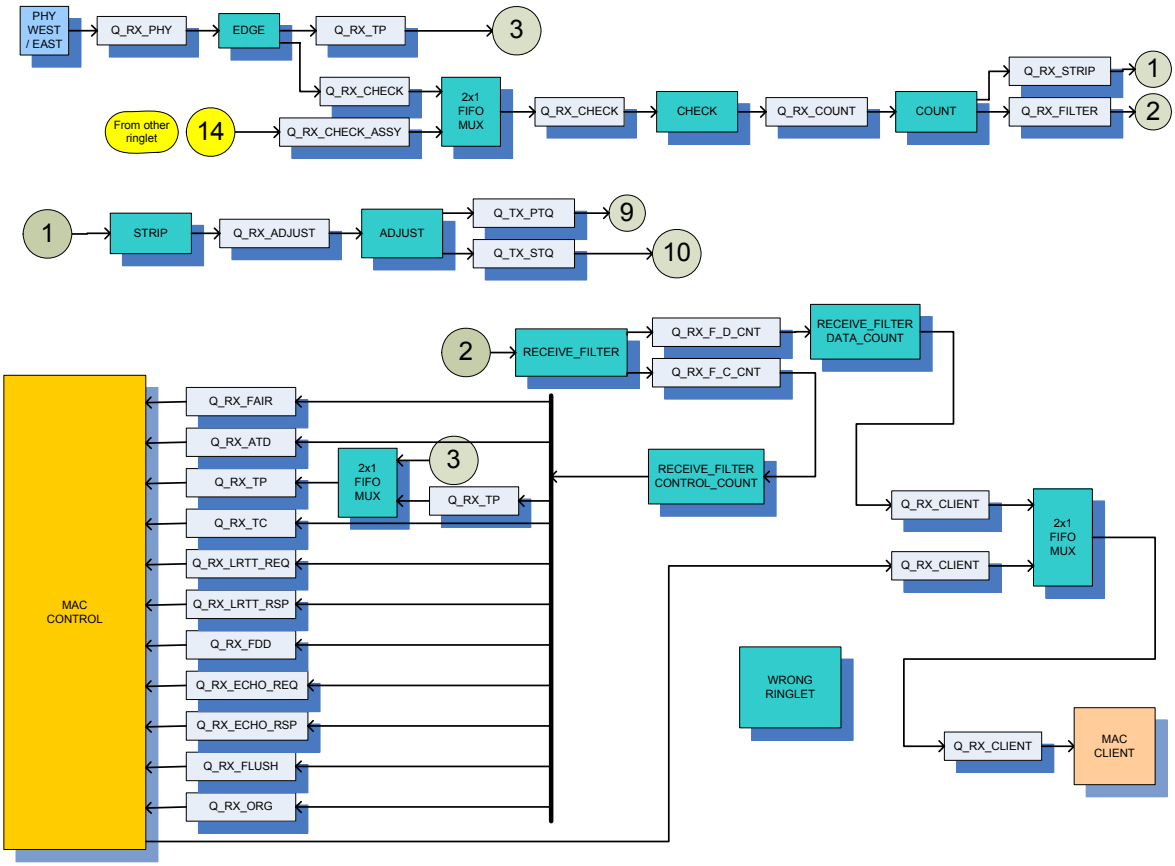


Figura 6.4 — Trajecto de dados da recepção

As figuras 6.4 e 6.5 estão ambas relacionadas, cada uma das referências numa das figuras corresponde à mesma na outra figura. Quando é referido "*to other ringlet*" ou "*from other ringlet*" significa que a referência correspondente vai para ou vem da outra unidade homóloga do *ringlet* oposto, respectivamente.

Todas as unidades Q_..., desta secção e das seguintes, correspondem a filas lógicas com o comportamento de FIFOs síncronos, excepto as filas Q_RX_CHECK_ASSY, Q_TX_PTQ e Q_TX_STQ, que correspondem a filas lógicas com o comportamento de FIFOs assíncronos pois encontram-se situadas entre as unidades de recepção e de transmissão, e as filas Q_RX_CLIENT e Q_TX_CLIENT que correspondem também a filas lógicas com o comportamento de FIFOs assíncronos mas nas interfaces entre o MAC do RPR e a camada cliente.

A unidade 2X1_FIFO_MUX corresponde a uma máquina de estados que retira uma trama completa de cada um dos dois FIFOs de entrada, e coloca-a no FIFO de saída sem interrupção, comutando para a entrada seguinte no final da transmissão de uma trama completa. Isto permite percorrer de uma forma simples, todos os FIFOs de entrada com tramas em espera. Apesar de

não fornecer qualquer selecção discriminando a qualidade de serviço da trama em espera, no entanto é relativamente eficiente e simples de implementar.

A unidade 1X3_FIFO_DEMUX corresponde a uma máquina de estados que retira uma trama completa do FIFO de entrada e coloca-a no FIFO de saída correspondente ao caminho que a trama deve seguir.

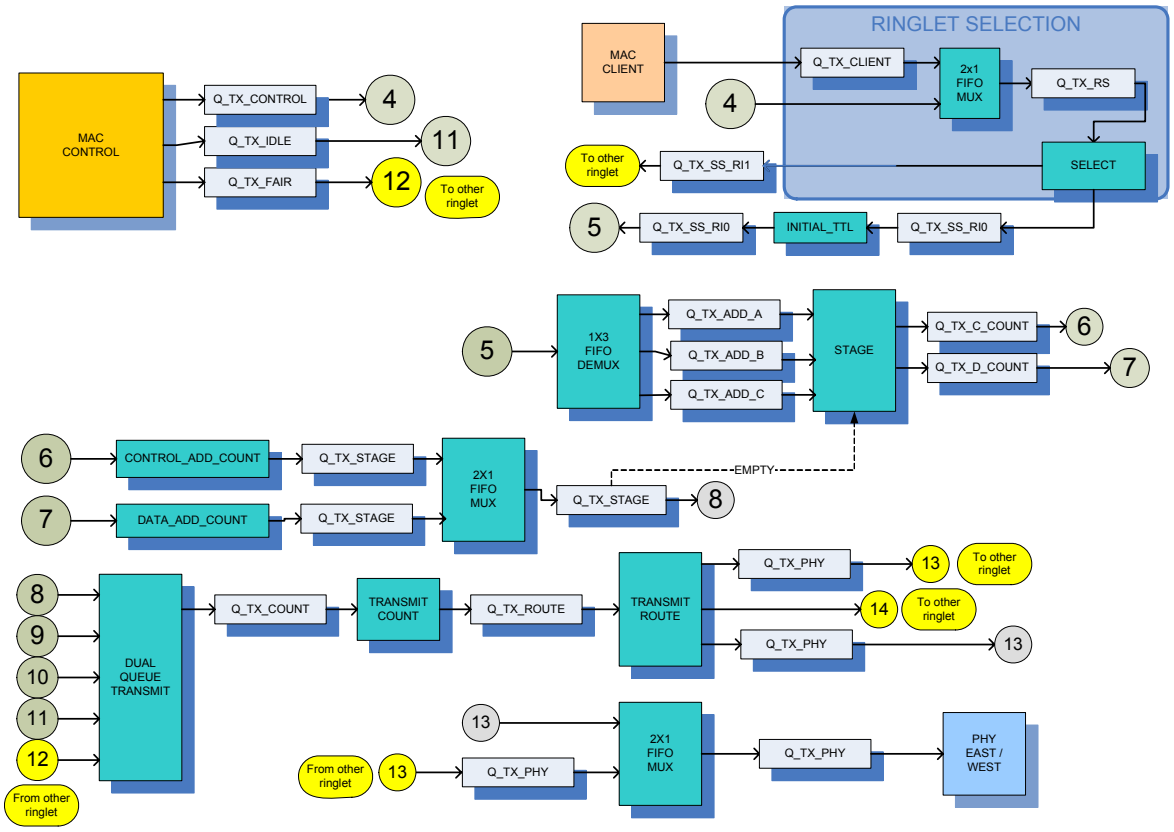


Figura 6.5 — Trajecto de dados da transmissão

Descrevendo de uma forma sucinta a unidade do trajecto de dados de recepção, na figura 6.4, a invocação de PHY_DATA.indication, pela sub-camada de reconciliação da interface Oeste (ou Este) da PHY, fornece uma trama na fila de entrada Q_RX_PHY que a conduz à máquina de estados de EDGE.

A máquina de estados EDGE retira a trama da fila de entrada Q_RX_PHY, processa-a e coloca-a então na fila Q_RX_CHECK se a variável *myEdgeState* for diferente de FROM_EDGE ou então coloca-a na fila Q_RX_TP se a variável *myEdgeState* for igual a FROM_EDGE, e se a trama for de controlo (FT_CONTROL) e de topologia e protecção (CT_TOPO_PROT). A unidade EDGE executa as funções necessárias para copiar determinadas tramas de uma estação vizinha, através de uma fronteira para a sub-camada de controlo do MAC, e para rejeitar todas as restantes tramas recebidas através de uma fronteira.

A máquina de estados 2X1_FIFO_MUX retira tramas, alternadamente, das filas Q_RX_CHECK e Q_RX_CHECK_ASSY desde a entrada e coloca-as na fila Q_RX_CHECK na saída.

A máquina de estados CHECK retira tramas da fila de entrada Q_RX_CHECK, processa-as e coloca-as na fila Q_RX_COUNT na saída. A unidade CHECK executa as funções necessárias para rejeitar tramas inválidas.

A máquina de estados COUNT retira a trama da fila de entrada Q_RX_COUNT, processa-a, duplica-a e coloca-a então simultaneamente em ambas as filas Q_RX_STRIP e Q_RX_FILTER. A unidade COUNT executa as funções necessárias para actualizar as estatísticas de taxa de fluxo.

A máquina de estados STRIP retira tramas da fila Q_RX_STRIP desde a entrada, processa-as e coloca-as na fila Q_RX_ADJUST na saída. A unidade STRIP executa as funções necessárias para fazer a decomposição das tramas recebidas do *ringlet* quando estas transitaram até ao seu máximo permitido.

A máquina de estados ADJUST retira a trama da fila de entrada Q_RX_ADJUST, processa-a e coloca-a então na fila Q_TX_PTQ ou Q_TX_STQ consoante a classe de serviço da mesma. A unidade ADJUST implementa as funções necessárias para actualizar os campos de controlo das tramas que transitam pela estação.

A máquina de estados RECEIVE_FILTER retira a trama da fila de entrada Q_RX_FILTER, processa-a e coloca-a então na fila Q_RX_F_D_CNT se a trama for de dados (FT_DATA), ou então coloca-a na fila Q_RX_F_C_CNT se a trama for de *fairness* (FT_FAIRNESS) ou de controlo (FT_CONTROL). A unidade RECEIVE_FILTER implementa as funções necessárias para copiar as tramas desde o *ringlet* para o cliente ou para a sub-camada de controlo do MAC.

A máquina de estados RECEIVE_FILTER_DATA_COUNT retira tramas da fila de entrada Q_RX_F_D_CNT, processa-as e coloca-as na fila Q_RX_CLIENT. A unidade RECEIVE_FILTER_DATA_COUNT implementa as funções necessárias para actualizar as estatísticas para tramas de dados recebidas e copiadas.

Posteriormente a máquina de estados 2X1_FIFO_MUX retira tramas, alternadamente, de ambas as filas de entrada Q_RX_CLIENT vindas da unidade de MAC Control ou de RECEIVE_FILTER_DATA_COUNT e coloca-as na fila Q_RX_CLIENT na saída para o cliente do MAC. Esta última fila conduz a um MA_DATA.indication ao cliente, fornecendo-lhe por fim a trama.

A máquina de estados RECEIVE_FILTER_CONTROL_COUNT retira a trama da fila de entrada Q_RX_F_C_COUNT, processa-a e coloca-a então na fila Q_RX_FAIR se a trama for de *fairness* (FT_FAIRNESS) ou então coloca-a nas filas Q_RX_ATD, Q_RX_TP, Q_RX_TC, Q_RX_LRTT_REQ, Q_RX_LRTT_RSP, Q_RX_FDD, Q_RX_ECHO_REQ, Q_RX_ECHO_RSP, Q_RX_FLUSH ou Q_RX_ORG, se for uma trama de controlo tipo CT_STATION_ATD, CT_TOPO_PROT, CT_TOPO_CHKSUM, CT_LRTT_REQ, CT_LRTT_RSP, CT_FDD, CT_OAM_ECHO_REQ, CT_OAM_ECHO_RSP, CT_OAM_FLUSH ou CT_OAM_ORG, respectivamente. Por fim cada uma daquelas filas envia a respectiva trama para a unidade de controlo do MAC. A unidade RECEIVE_FILTER_CONTROL_COUNT implementa as funções necessárias para actualizar as estatísticas para tramas de controlo recebidas e copiadas.

Existe ainda outra máquina de estados 2X1_FIFO_MUX que retira tramas, alternadamente, de ambas as filas de entrada Q_RX_TP vindas do EDGE ou do RECEIVE_FILTER_CONTROL_COUNT e coloca-as na fila Q_RX_TP na saída para o controlo do MAC.

Existe também uma unidade chamada WRONG_RINGLET que controla a variável de *tossWrongRingletIDs* usada na unidade de recepção. Esta variável faz com que o MAC rejeite as tramas recebidas com o *ringlet* errado, que é necessário para suportar o conteúdo de contexto em sistemas *wrapping*. As tramas com o *ringlet* errado são rejeitadas como parte do conteúdo de contexto durante eventos *unwrap*. Durante a operação normal, as tramas com o *ringlet* errado podem ocorrer devido somente a uma execução não compatível que injecta a trama, e são rejeitadas pela unidade *Check*. Esta unidade funciona em paralelo às unidades que estão a processar as tramas recebidas.

Descrevendo também de uma forma sucinta a unidade do trajecto de dados de transmissão, na figura 6.5, a invocação de MA_DATA.request pela sub-camada do cliente, fornece uma trama na fila de entrada Q_TX_CLIENT que a conduz à máquina de estados de 2X1_FIFO_MUX.

Por sua vez a máquina de estados 2X1_FIFO_MUX retira tramas, alternadamente, das filas de entrada Q_TX_CLIENT e Q_TX_CONTROL (vinda da unidade de controlo do MAC), e coloca-as na fila Q_TX_RS na saída.

A máquina de estados SELECT retira a trama de entrada da fila Q_TX_RS, processa-a e coloca-a então na fila Q_TX_SS_RI1 para enviar no *ringlet1*, ou coloca-a na fila Q_TX_SS_RI0 para enviar no *ringlet0*, ou então coloca-a em ambas as filas para enviar em ambos os *ringlets*. A unidade SELECT implementa as funções necessárias para a admissão de tramas do cliente e ao processamento do cabeçalho associado às mesmas.

A máquina de estados INITIAL_TTL retira tramas da fila Q_TX_SS_RI0 desde a entrada, processa o campo do *ttl* das mesmas e coloca-as na fila Q_TX_SS_RI0 na saída. A unidade INITIAL_TTL proporciona as alterações ao valor do campo *ttl* das tramas, segundo as regras às quais o *ttl* deve obedecer (Tabela 5.4), e também do correspondente campo *ttlBase* das mesmas.

A máquina de estados 1X3_FIFO_DEMUX retira a trama da fila de entrada Q_TX_SS_RI0 e coloca-a então nas filas Q_TX_ADD_A, ou Q_TX_ADD_B, ou Q_TX_ADD_C consoante a classe de serviço da mesma.

A máquina de estados STAGE retira a trama das filas de entrada Q_TX_ADD_A, ou Q_TX_ADD_B, ou Q_TX_ADD_C, consoante a classe mais prioritária, e se o sinal EMPTY da Q_TX_STAGE estiver activo, processa a trama e coloca-a então na fila Q_TX_C_COUNT se a trama for de controlo (FT_CONTROL), ou na fila Q_TX_D_COUNT se a trama for de dados (FT_DATA). A unidade de STAGE implementa as funções necessárias para seleccionar uma trama de controlo ou uma trama de dados vinda do cliente, para admissão na fila Q_TX_STAGE, e também implementa as funções necessárias para o processamento do cabeçalho das referidas tramas.

A máquina de estados CONTROL_ADD_COUNT retira tramas da fila de entrada Q_TX_C_COUNT, processa-as e coloca-as na fila Q_TX_STAGE na saída. A unidade CONTROL_ADD_COUNT conta as tramas de controlo adicionadas.

A máquina de estados DATA_ADD_COUNT retira tramas da fila de entrada Q_TX_D_COUNT, processa-as e coloca-as noutra fila Q_TX_STAGE na saída. A unidade DATA_ADD_COUNT conta os bytes e as tramas de dados adicionados.

Por sua vez a máquina de estados 2X1_FIFO_MUX retira tramas, alternadamente, de ambas as filas de entrada Q_TX_STAGE e coloca-as na fila Q_TX_STAGE na saída.

Existe ainda outra máquina de estados DUAL_QUEUE_TRANSMIT que retira tramas das filas de entrada Q_TX_STAGE, Q_TX_PTQ, Q_TX_STQ, Q_TX_IDLE ou Q_TX_FAIR sendo esta última fila originada do outro *ringlet*, e coloca-as na fila Q_TX_COUNT na saída. Esta unidade DUAL_QUEUE_TRANSMIT implementa as funções necessárias para seleccionar que trama deve ser transmitida num MAC de fila-dupla. O objectivo é de esvaziar sempre a PTQ antes da transmissão de tramas do cliente, mas permitir que a STQ encha enquanto são transmitidas tramas adicionadas pelo cliente.

A máquina de estados TRANSMIT_COUNT retira tramas da fila de entrada Q_TX_COUNT, processa-as e coloca-as na fila Q_TX_ROUTE na saída. A unidade TRANSMIT_COUNT implementa as funções necessárias para actualizar as estatísticas de taxas de fluxo do MIB.

A máquina de estados TRANSMIT_ROUTE retira a trama da fila de entrada Q_TX_ROUTE, processa-a e coloca-a então nas filas Q_TX_PHY para ser enviada no outro *ringlet*, ou Q_RX_CHECK_ASSY do outro *ringlet*, ou Q_TX_PHY para ser enviada no próprio *ringlet*. A unidade TRANSMIT_ROUTE implementa as funções necessárias para transmitir uma trama, baseada no estado de protecção da extensão para a qual o trajecto de dados está a transmitir.

Por fim a máquina de estados 2X1_FIFO_MUX retira tramas, alternadamente, de ambas as filas de entrada Q_TX_PHY uma do próprio *ringlet* e outra originada no *ringlet* oposto, e coloca-as na fila Q_TX_PHY na saída, directamente para a interface com a PHY. Esta última fila Q_TX_ PHY, conduz a um PHY_DATA.request para a sub-camada de reconciliação da interface Este (ou Oeste) da PHY

6.5.2 Unidade de *Fairness*

Os circuitos da unidade de *Fairness* que estão descritos no capítulo cinco, estão ilustrados de uma forma simplificada na figura 6.6.

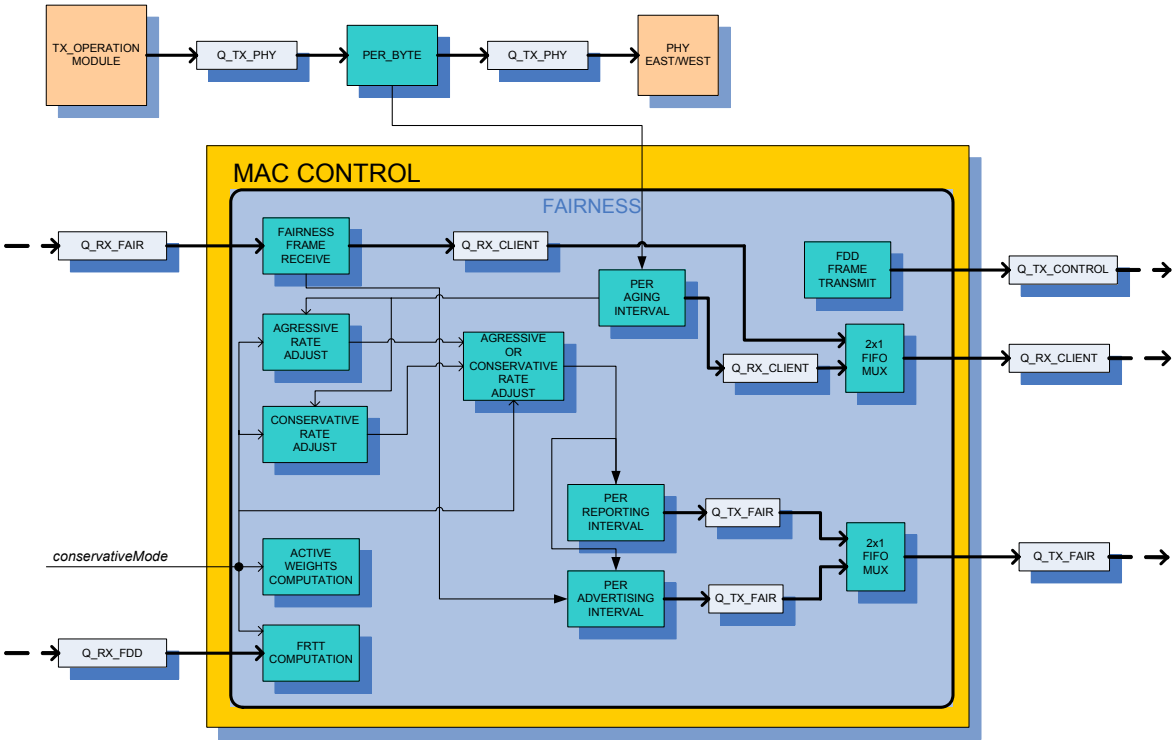


Figura 6.6 – Unidade de *Fairness* integrada na unidade de MAC CONTROL

Aqui na unidade de *Fairness* tal como nas unidades de trajecto de dados foram mantidas as interfaces de 32 bits em paralelo. O objectivo foi o de manter um controlo mais fácil no trajecto das tramas, dentro e nas entradas e saídas desta unidade.

Nesta unidade é usado o relógio de recepção em todas as máquinas de estado excepto para as máquinas de estado FDD_FRAME_TRANSMIT, PER_REPORTING_INTERVAL e PER_ADVERTISING_INTERVAL (figura 6.6), onde é usado o relógio de transmissão. Na figura 6.6 estão ilustradas as diversas máquinas de estados desta unidade e as ligações entre as mesmas.

A máquina de estados PER_BYTE executa as suas actividades à saída da unidade de transmissão na interface com a PHY. Esta unidade recebe e transmite as tramas que flúem para o anel, através da fila Q_TX_PHY. Cada byte de dados que transita pela estação ou que fica disponível para adição ao *ringlet* é contabilizado com o incremento de contadores de taxa e o ajuste de indicações de policiamento. Esta unidade aparece apenas ilustrada na figura 6.6 mas efectivamente o troço à qual faz parte, pertence à unidade de transmissão na figura 6.5, onde não aparece ilustrada após a fila Q_TX_PHY na saída para a PHY.

A máquina de estados PER_AGING_INTERVAL executa a filtragem passa-baixo dos contadores de taxa, a normalização das taxas, o ajuste da taxa específico ao modo da estação (agressivo ou conservador), o cálculo do *allowedRateCongested,* o envio opcional de um MA_CONTROL.indication para o cliente, o *envelhecimento* dos contadores de taxa e o ajuste dos níveis alto e baixo da STQ. O envio de um MA_CONTROL.indication para o MAC cliente é feito mediante o envio de uma trama *fairness* com um *opcode* igual a SINGLE_CHOKE_IND através da fila Q_RX_CLIENT.

A máquina de estados AGRESSIVE_RATE_ADJUST é invocada pela máquina de estados PER_AGING_INTERVAL, numa estação que usa o método agressivo para o cálculo da taxa. As actividades efectuadas por esta máquina de estados incluem o cálculo e a normalização do *localFairRate.* O *allowedRate* mantém o seu valor inicial de *maxAllowedRate*. Quando se usa o método agressivo, o tráfego que não transita o ponto de congestionamento é restringido apenas pelo *maxAllowedRate*.

A máquina de estados CONSERVATIVE_RATE_ADJUST é invocada pela máquina de estados PER_AGING_INTERVAL, numa estação que usa o método conservador para o cálculo da taxa. As actividades efectuadas por esta unidade incluem o cálculo e a normalização do *localFairRate.*

A unidade AGRESSIVE_OR_CONSERVATIVE_RATE_ADJUST actua como um comutador para a selecção através do sinal *conservativeMode*, do método conservador ou agressivo para o cálculo da taxa.

A máquina de estados PER_ADVERTISING_INTERVAL é usada pela unidade de *fairness* para que na expiração de cada *advertisingInterval* anuncie o seu *fairRate* localmente calculado, ou propague o *fairRate* recebido da estação vizinha a jusante, ou anuncie o FULL_RATE à estação vizinha a montante. A máquina de estados PER_ADVERTISING_INTERVAL envia tramas *fairness* com aquela informação no campo *fairRate* das mesmas, através da fila Q_TX_FAIR.

A máquina de estados PER_REPORTING_INTERVAL é usada pela unidade de *fairness* para que na expiração de cada *reportingInterval* transmita um relatório da taxa a todas as estações no *ringlet*. A máquina de estados PER_REPORTING_INTERVAL envia tramas *fairness* MULTI_CHOKE_IND com aquela informação no campo *fairRate* das mesmas, através da fila Q_TX_FAIR.

A máquina de estados ACTIVE_WEIGHTS_COMPUTATION executa apenas ao ser usado o método conservador de ajuste de taxa e quando o *activeWeightsDetection* for TRUE. Na expiração de um *activeWeightsInterval,* a estação local calcula a soma dos pesos das estações das quais uma trama elegível para *fairness* foi recebida durante o *activeWeightsInterval* decorrido. O *activeWeightsInterval* é configurado como um múltiplo inteiro do *agingInterval*. O valor de *activeWeights é* referenciado pela máquina de estados PER_AGING_INTERVAL.

A máquina de estados FAIRNESS_FRAME_RECEIVE faz o processamento de uma SCFF recebida através da fila Q_RX_FAIR (propaganda da taxa) ou de uma MCFF recebida também através da fila Q_RX_FAIR (relatório da taxa). A informação contida numa propaganda é guardada para processamento na expiração do *advertisingInterval* seguinte. A informação *multi-choke* associada com um relatório da taxa é opcionalmente transferida ao cliente do MAC através da fila Q_RX_CLIENT.

A máquina de estados FDD_FRAME_TRANSMIT emite periodicamente um par de tramas FDD através da fila Q_TX_CONTROL, quando a estação é a cauda de um domínio de congestionamento.

A máquina de estados FRTT_COMPUTATION efectua o cálculo do valor do FRTT que é usado pela máquina de estados CONSERVATIVE_RATE_ADJUST.

As máquinas de estados 2X1_FIFO_MUX retiram tramas, alternadamente, das filas de entrada e colocam-nas na fila de saída.

6.5.3 Unidade de Topologia e Protecção

Os circuitos da unidade de Topologia e Protecção que estão descritos no capítulo cinco, estão ilustrados de uma forma simplificada na figura 6.7.

Descrevendo de uma forma sucinta a unidade de Topologia e Protecção, ilustrada na figura 6.7.

As máquinas de estados RECEIVE_MONITOR são responsáveis por monitorizar o estado da extensão à qual cada uma está ligada. Cada uma activa uma falha de sinal baseada na detecção de uma falha da ligação com a PHY, ou na perda de *keepalives,* ou na falta de cabo/fibra. Cada uma activa também um SD baseado na detecção de uma condição de degradação da ligação com a PHY. As SCFFs são usadas como *keepalives*. A transmissão de *keepalives* do RPR deve

ocorrer apenas se for possível uma operação normal do MAC. Estas máquinas de estados geram a indicação *spanOperStatus.[ri]* para a máquina de estados de PROTECTION_UPDATE, baseado no estado da ligação de recepção.

A máquina de estados TOPOLOGY_CONTROL actualiza a base de dados de topologia baseando-se na recepção de tramas TP desde o anel através de Q_RX_TP_PARSE. Esta máquina de estados é também responsável por invocar a máquina de estados PROTECTION_UPDATE, quando ocorrem alterações no estado de protecção local ou pedidos de gestão locais. Desempenha também as verificações de preferência de tramas *jumbo* e de configuração da protecção.

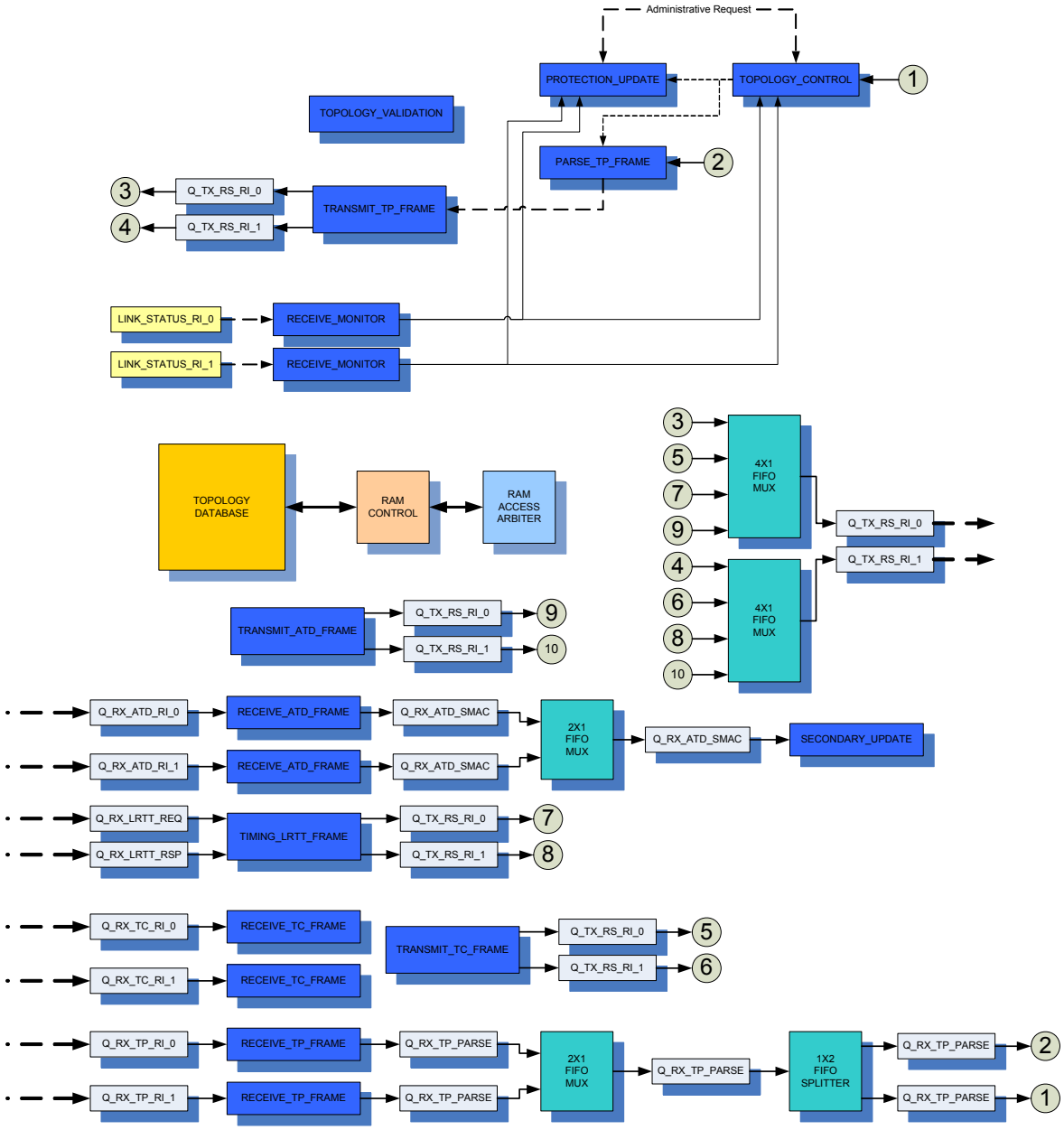


Figura 6.7 – Unidade de Topologia e Protecção integrada na unidade de MAC CONTROL

É expectável que todas as estações dentro de um anel estejam configuradas para usar o mesmo mecanismo de protecção. Cada estação indica a sua configuração de protecção enviando tramas

TP. A indicação na trama TP é feita através do campo *wc*. No caso de uma estação detectar que uma qualquer outra estação tem uma configuração de protecção que não coincide com a sua própria, então é despoletado um sinal de defeito. Cada estação usa o mecanismo de protecção ajustado pelo seu *protConfig* local.

A máquina de estados PARSE_TP_FRAME para além de processar as tramas TP recebidas do anel através de Q_RX_TP_PARSE, para actualizar a base de dados de topologia, é também responsável por detectar quando é que a máquina de estados PROTECTION_UPDATE deve ser invocada. Ajusta o *topoChanged* para as máquinas de estados TOPOLOGY_CONTROL e TOPOLOGY_VALIDATION, o *protectChanged* para a TOPOLOGY_CONTROL, o *newNeighbor.[rid]* para a PROTECTION_UPDATE e o *transmitTpFrame* para a TRANSMIT_TP_FRAME.

A máquina de estados PROTECTION_UPDATE é invocada pela máquina de estados TOPOLOGY_CONTROL, e actua numa extensão especificada. Esta máquina de estados usa a informação contida na base de dados de topologia para determinar o estado de protecção das suas extensão e fronteira, antes de actualizar a base de dados de topologia. Actua para ambos os tipos de protecção *steering* e *wrapping*. Actua também na extensão (este ou oeste) de uma estação para a qual é invocada.

Aquelas condições de protecção afectam a topologia do RPR. Por exemplo, uma extensão pode tornar-se numa fronteira se houver uma condição de protecção suficientemente severa numa ou em ambas as ligações da extensão. Uma directiva FS ou uma condição SF forçam a existência de uma fronteira.

Esta máquina de estados para além de determinar o estado da fronteira da extensão directamente ligada, actualiza a base de dados de topologia baseando-se nesse estado. Ajusta também o *topoChanged* para a TOPOLOGY_VALIDATION e o *transmitTpFrame* para a TRANSMIT_TP_FRAME.

Ambos, o estado de protecção da estação vizinha (*neighborState*) e o estado mais elevado de protecção no resto do anel (*distantState*) podem sobrepor/interromper um estado de protecção local não fatal. Se o resto do anel é IDLE, ambos o estado de protecção local e o estado de protecção da estação vizinha devem tornar-se IDLE para terminar uma fronteira.

A máquina de estados TOPOLOGY_VALIDATION, valida a base de dados de topologia, ajusta de acordo o *topologyStable* e o *topologyValid,* e relata os defeitos detectados. O valor do *topologyStable* é ajustado quando a topologia (posições das estações e das fronteiras) não se alterou durante o tempo *stabilityTimeout* e quando todas as estações na topologia estão válidas. É relatado um sinal *topoInstabilityDefect* se a topologia não estabilizar dentro do tempo *instabilityTimeout*.

Quando a topologia está estável, o *topologyValid* é ajustado se as verificações da consistência da topologia e a contagem das estações forem correctas. Se for encontrada uma entrada inválida dentro da topologia, é relatado um *topoEntryInvalidDefect*. Se o número de estações no anel exceder o MAX_STATIONS, é relatado um *maxStationsDefect*. Se for determinado que a topologia está inconsistente, é relatado um *topoInconsistencyDefect*. Se a configuração de protecção das estações no anel é não uniforme, é relatado um *protMisconfigDefect*.

Assim que ajustados, estes defeitos podem ser apagados após uma alteração da topologia, seguida de uma re-estabilização.

A variável de entrada *topoChanged* (originada da máquina de estados PARSE_TP_FRAME) indica que um endereço de MAC de origem que está a ter impacto na estabilidade ou que o estado de uma fronteira se alterou. Esta máquina de estados determina se ocorreu alguma alteração com impacto na topologia, durante o período do temporizador de estabilidade. São relatados defeitos se falhar a verificação de validade. Se passarem todas as verificações de validação, a base de dados da topologia é actualizada.

A variável *lrttRequest* é ajustada para a máquina de estados TIMING_LRR_FRAME. A variável *transmitTcFrame* é ajustada para a máquina de estados TRANSMIT_TC_FRAME. A variável *needSecondaryMacValidation* é ajustada para a máquina de estados SECONDARY_UPDATE. A

base de dados *sourceCheck* é preenchida. Quando a topologia é validada e os *checksums* de topologia coincidem com os *checksums* vizinhos, então o conteúdo de contexto é apagado.

Tal como referido no capítulo cinco, a máquina de estados TRANSMIT_TP_FRAME é usada para transmitir tramas TP e determinar quando estas são enviadas. As tramas TP são transmitidas em dois períodos de tempo distintos, *txFastTimeout* e *txSlowTimeout*. Um número fixo de tramas TP é transmitido no temporizador rápido quando o *transmitTpFrame* é ajustado a TRUE, antes de reverter para a taxa de *txSlowTimeout*.

As máquinas de estados RECEIVE_TP_FRAME recebem tramas TP através das Q_RX_TP_RI_0 e Q_RX_TP_RI_1 de cada uma das extensões, e processam-nas antes de as passar para a máquina de estados TOPOLOGY_CONTROL.

A recepção de uma trama TP contendo nova informação, faz com que a sub-camada de controlo do MAC actualize a sua base de dados de topologia. Quando uma trama TP indica ausência de fibra/cabo, a base de dados de topologia não é actualizada, mas é ajustada a variável *miscablingDefect[ri]* para ser usada nas máquinas de estados RECEIVE_MONITOR. As tramas TP providenciam um número de sequência para indicar quando é que foi alterado o conteúdo da trama TP. Isto permite ignorar tramas redundantes.

Tal como referido no capítulo cinco, a máquina de estados TRANSMIT_TC_FRAME é usada para transmitir tramas TC e determinar quando estas são enviadas. A variável de entrada *transmitTcFrame* despoleta as transmissões das tramas TC baseadas no temporizador rápido.

As máquinas de estados RECEIVE_TC_FRAME recebem tramas TC através das Q_RX_TC_RI_0 e Q_RX_TC_RI_1 de cada uma das extensões.

A recepção de uma trama TC num *ringlet* faz com que a sub-camada de controlo do MAC actualize a sua base de dados de topologia, tendo em conta que não esteja presente uma condição SF na extensão de recepção. Se a topologia está estável, as mudanças nas tramas TC recebidas também despoletam a transmissão de tramas TC adicionais.

A máquina de estados TRANSMIT_ATD_FRAME gera tramas ATD que devem ser transmitidas independentemente por cada estação através de Q_TX_RS_RI_0 e de Q_TX_RS_RI_1 para cada uma das extensões, quando da expiração do temporizador *atdTimeout* iniciado na inicialização da estação. Estas tramas também podem ser enviadas quando da alteração de qualquer dos seus conteúdos.

De modo a permitir que outras estações tenham tempo de reagir a alterações nos atributos relatados de estações, aquelas devem relatar os valores alterados que usam recursos (por exemplo ATT_STATION_BW, ATT_STATION_NAME, ATT_STATION_SEC_MAC), da seguinte maneira:

a) Antes de adicionar um recurso (ex. mais largura de banda reservada ou um endereço secundário de MAC), relatar o uso do recurso, depois iniciar o uso do mesmo pelo menos RRTT + 10ms;

b) Após remover a utilização de um recurso (ex. Nome de estação), esperar pelo menos RRTT + 10 ms antes de remover o relato do mesmo;

Para atributos para os quais existe um total de recursos que não deve ser excedido (ex. largura de banda reservada), não existe controlo no MAC para prevenir a ocorrência desta condição.

Para atributos para os quais existem um ou mais itens de cada recurso (ex. nome de estação ou endereço secundário do MAC), nenhuma estação reivindicará posse se outra estação já o tiver feito. Uma condição de “corrida” que permita que duas estações relatem o mesmo valor, forçará ambas as estações a remover a sua reivindicação e a emitir um alarme.

A máquina de estados RECEIVE_ATD_FRAME recebe tramas ATD de cada *ringlet* desde as filas Q_RX_ATD_RI_0 e Q_RX_ATD_RI_1 de cada uma das extensões, o que faz com que a sub-camada de controlo do MAC actualize a base de dados de topologia. As tramas ATD são ignoradas nas seguintes condições:

a) Se recebidas de uma estação que não se encontra na base de dados de topologia;

b) Se recebida de uma estação cuja entrada na base de dados de topologia está marcada como INVALID;

c) Se recebida de uma estação que não está na mesma localização na base de dados de topologia tal como indicado pelo valor de *ttl* da trama ATD recebida.

As entradas ATT nas tramas ATD são ilegais e são anuladas sob as seguintes condições:

a) Uma entrada ATT do mesmo tipo está previamente contida dentro da trama;

b) O ATT tem um tamanho ilegal;

c) O tipo ATT não é suportado;

d) O ATT estende-se para além do fim da trama.

O *offset* para o próximo ATT é sempre baseado no *offset* e no tamanho relatado do ATT anterior, mesmo que o tipo não seja reconhecido ou que o comprimento seja diferente do valor esperado.

As regras de processamento do ATT, acima referidas, são baseadas nas definições dos diversos tipos de ATT.

Após o cumprimento das verificações ATD e ATT, acima referidas, para o ATT do MAC secundário é passada uma cópia da trama, através da fila Q_RX_ATD_SMAC, para a máquina de estados SECONDARY_UPDATE que faz a interpretação dos seus conteúdos. Para outros tipos de ATT, a informação contida no ATT é escrita na base de dados de topologia.

Quando a base de dados de topologia é actualizada, a máquina de estados SECONDARY_UPDATE é activada para validar os endereços secundários do MAC que tenham sido modificados. A validação é executada quando o *topologyValid* é ajustado pela máquina de estados TOPOLOGY_VALIDATION.

Se qualquer endereço MAC secundário na topologia, for alcançável e estiver duplicado por qualquer endereço MAC primário ou secundário no anel, o *duplicateSecMacAddressDefect* é ajustado. Se o número de endereços secundários não duplicados do MAC, exceder MAX_SEC_MAC, o *maxSecMacAddressDefect* é ajustado.

Tal como referido no capítulo cinco, a máquina de estados TIMING_LRTT_FRAME gera o pedido de tramas LRTT e processa as tramas de resposta recebidas de volta, com o objectivo de calibrar o LRTT do anel.

O valor do LRTT é usado para calcular o FRTT. Sempre que uma topologia é validada, uma estação que implementa o modo conservador do *fairness,* transmite tramas *unicast* de pedido de tempo de uma volta fechada em torno do anel (LRTT) para todas as outras estações em cada *ringlet*. Se a topologia se tornar instável enquanto a medição do LRTT estiver a decorrer, esta medição do LRTT é abortada.

As estações respondem a um pedido de LRTT enviando uma resposta LRTT através do *ringlet* oposto. Ao receber a sua resposta de LRTT, uma estação calcula o LRTT e mantém este valor na base de dados de topologia.

Uma estação pode medir ou estimar a quantidade de tempo de latência adicionado entre a recepção da trama de pedido de LRTT e a transmissão da trama de resposta de LRTT, e devolve esse tempo de latência através da trama de resposta.

6.5.4 Unidade de Operação, Administração e Manutenção

Os circuitos da unidade de OAM que estão descritos no capítulo cinco, estão ilustrados de uma forma simplificada na figura 6.8 dentro da unidade de controlo do MAC.

A unidade OAM_FRAME_TRANSMIT implementa as funções necessárias para processar o pedido de tramas OAM por parte do cliente da estação de origem. É requisitada a operação de geração de uma trama OAM pelo cliente do MAC. O cliente envia uma trama de dados com o campo *ft*=FT_CONTROL e tendo os oito bits mais significativos do campo de *protocolType* dessa trama de dados, a informação correspondente ao campo de *controlType* de uma trama de

controlo. Se a informação contida no campo *controlType* for OAM_ECHO_REQ, OAM_FLUSH_REQ, ou OAM_ORG_REQ, a unidade OAM_FRAME_TRANSMIT gerará uma trama OAM de ECHO, FLUSH, ou ORG, respectivamente.

A unidade OAM_FRAME_RECEIVE implementa as funções necessárias para a recepção de tramas OAM, incluindo a formação de uma trama de ECHO e envio da mesma para a estação objectivo.

A máquina de estados OAM_FRAME_TRANSMIT retira tramas da fila de entrada Q_OAM_REQ, processa-as e coloca-as na fila Q_TX_CONTROL na saída.

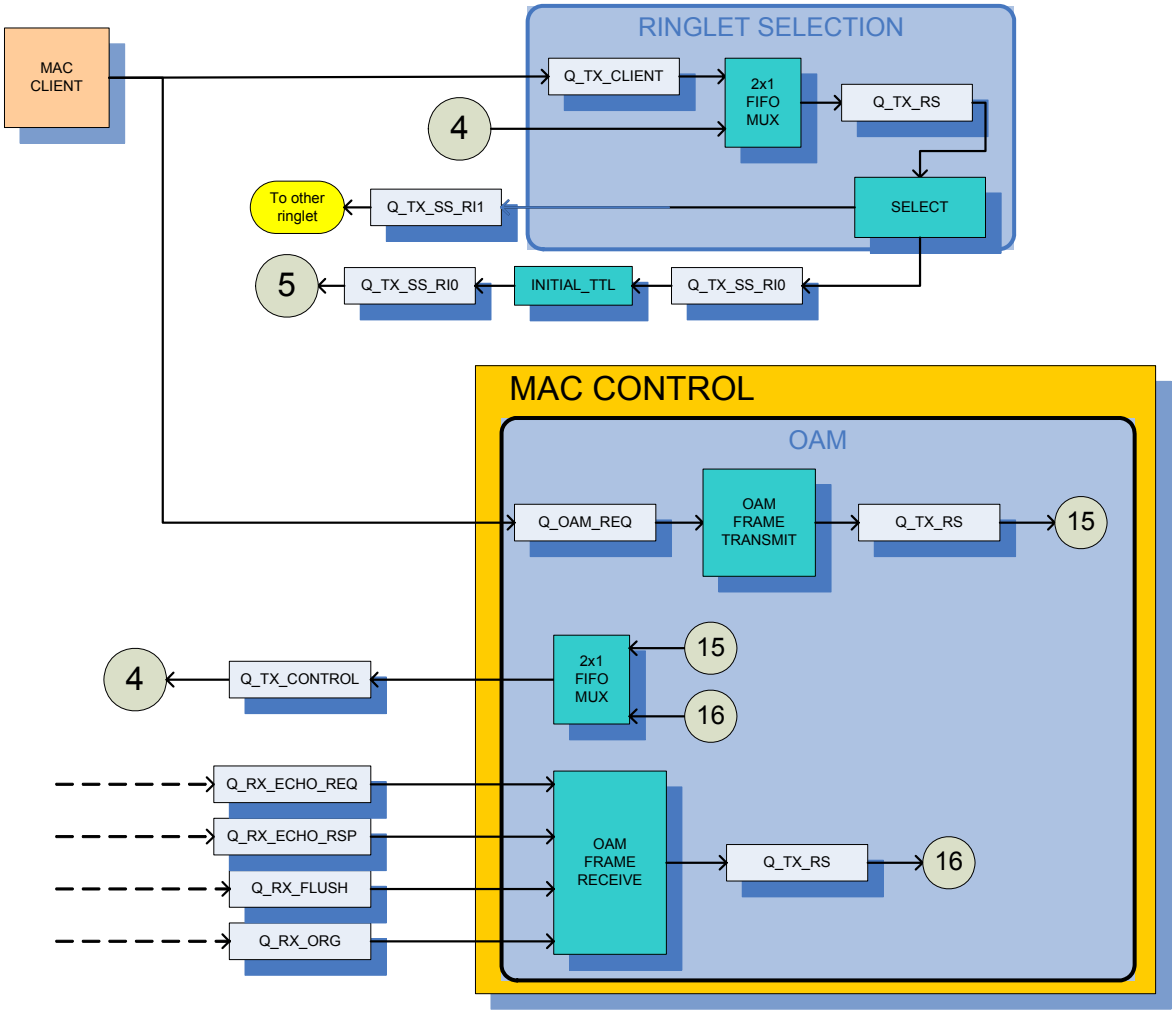


Figura 6.8 – Unidade de OAM integrada na unidade de MAC CONTROL

A máquina de estados OAM_FRAME_RECEIVE retira tramas das filas de entrada Q_RX_ECHO_REQ, ou de Q_RX_ECHO_RSP, ou de Q_RX_FLUSH, ou de Q_RX_ORG, e coloca-as na fila Q_TX_CONTROL na saída.

Por sua vez a máquina de estados 2X1_FIFO_MUX retira tramas, alternadamente, de ambas as filas de entrada Q_TX_CONTROL e coloca-as na fila Q_TX_CONTROL na saída.

6.5.5 Rate Control

A unidade de Rate Control é constituida pelos módulos *Idle Shaper*, *MAC_Control Shaper*, *Downstream Shaper*, *ClassA Shaper*, *ClassB Shaper*, *PreCongestion Shaper*, *PostCongestion Shaper*, *Source Shaper* e *FairnessEligibleSendIndication Shaper*. O algoritmo dos módulos da unidade de *Rate Control* está descrito no capítulo cinco e os circuitos estão ilustrados de uma forma simplificada na figura 6.9.

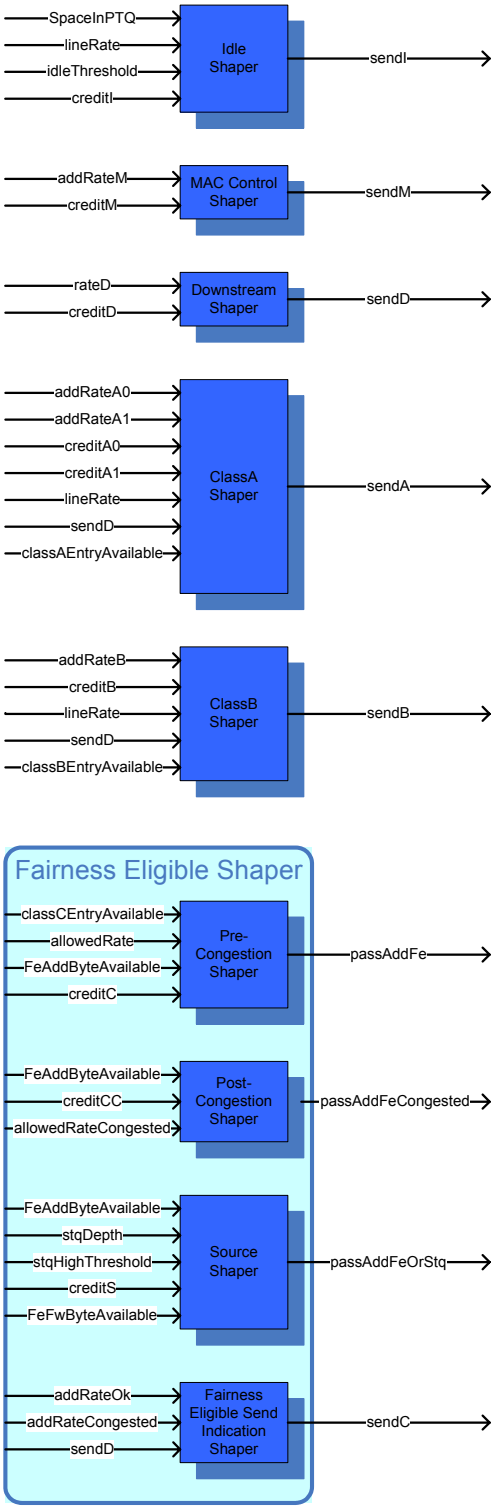


Figura 6.9 – Unidade de *Rate Control*

O *Idle Shaper* limita o tráfego de tramas *idle* fornecidas pelo MAC, aos limites reservados para este tipo de tráfego. As tramas *idle* são introduzidas no trajecto de dados de transmissão a uma taxa fixa equivalente a 500ppm da taxa da linha (*lineRate*). A taxa das tramas *idle* introduzidas é controlada pelo nível da PTQ. Enquanto a PTQ é enchida, o espaço livre torna-se menor do que *o valor* de *idleThreshold*. Neste caso, a taxa das tramas *idle* reduz-se para 250ppm da *lineRate,* o que aumenta a taxa de dados transmitidos para outros tipos de tramas, que por sua vez reduz eventualmente o espaço livre na fila de transmissão.

O MAC *Control Shaper* funciona em paralelo com os outros *shapers* através dos quais as tramas de controlo passam (isto é, os *shapers* subclasse-A0, classe-B e *fairness eligible*). Uma trama de controlo debita créditos de ambos os *shapers* através dos quais passa.

As taxas dos *shapers* subclasse-A0, classe-B e *fairness eligible*, são ajustadas de modo a incluir antecipadamente bastante largura de banda para o tráfego de controlo.

As acumulações de *creditM* ocorrem em todos os intervalos de actualização. O valor de *creditM* é originado das máquinas de estados de transmissão, e está sempre a ser decrementado de cada vez que é enviado um byte de uma trama de controlo. Esta máquina de estados gera o sinal de controlo *sendM*. O valor deste sinal está também em permanente actualização.

O *Downstream Shaper* monitoriza o tráfego adicionado e em trânsito para assegurar níveis suficientes e sustentáveis de tráfego *subclasse-A0* para estações a jusante.

Os *Shapers ClassA* e *ClassB* geram os sinais de controlo *sendA* e *sendB*, respectivamente.

O módulo de *Fairness Eligible Shaper* consiste nas seguintes unidades:

a) *Pre-Congestion Shaper*: ajusta os créditos para o tráfego adicionado *fairness eligible* e gera a indicação de *passAddFe,* que se trata de um valor booleano que indica se o tráfego elegível para ser processado pelo algoritmo de *Fairness*, pode ou não ser adicionado ao anel;

b) *Post-Congestion Shaper*: ajusta os créditos para o tráfego adicionado *fairness eligible* que flúi para além do ponto de congestionamento e gera a indicação de *passAddFeCongested*, que se trata de um valor booleano que indica se o tráfego elegível para ser processado pelo algoritmo de *Fairness,* limitado a um destino para além do ponto de congestionamento, pode ser adicionado ao anel;

c) *Source Shaper*: fornece o sinal de controlo *passAddFeOrStq* que é usado para regular entre o tráfego adicionado *fairness eligible* e o tráfego em trânsito na STQ;

d) *Fairness Eligible Send Indication*: fornece a indicação de *sendC*.

Esta unidade, *Fairness Eligible Send Indication*, descreve a indicação de controlo de fluxo gerada pelo MAC para controlar o tráfego *fairness eligible* (Classe-B-EIR e Classe-C) nos seus limites permitidos. O tráfego adicionado *fairness eligible* é controlado apenas pelas saídas do algoritmo de *fairness* e não por um método *token bucket* como usado pelos outros *shapers*.

O tráfego adicionado *fairness eligible* tem a sua taxa controlada a duas taxas, através da indicação de *sendC*. É sempre limitado ao valor de *allowedRate*. Adicionalmente, todo o tráfego fairness eligible que passar o ponto de congestionamento é também limitado ao valor de *allowedRateCongested.*

O valor de *sendC* não é um valor booleano, é um valor inteiro que fornece uma contagem máxima de extensões que as tramas *fairness eligible* podem atravessar. Isto permite que uma implementação estrangule selectivamente o tráfego *fairness eligible* baseado na contagem das extensões até ao seu objectivo. Uma indicação de *sendC* de valor 0 indica que o valor de *allowedRate* para a estação foi excedido, e consequentemente as tramas *fairness eligible* não podem ser emitidas (porque qualquer destino estaria mais afastado do que 0 extensões). Uma indicação de *sendC* superior a 0 e inferior ao tamanho do anel, indica que o valor de *allowedRateCongested* foi excedido através do congestionamento, e consequentemente as tramas *fairness eligible* podem ser emitidas até ao ponto de congestionamento (cuja distância de contagem das extensões é indicada pelo valor de *sendC*). Uma indicação de *sendC* (de pelo menos) do tamanho do anel indica que nenhuma taxa de *fairness* foi excedida, e

consequentemente as tramas *fairness eligible* podem ser emitidas até uma qualquer distância desejada.

6.6 Implementação e simulação do *Idle Shaper*

Abaixo está descrita, puramente como exemplo da filosofia seguida neste desenvolvimento do MAC do RPR, a implementação completa em SystemC da máquina de estados do *Idle Shaper*. A apresentação de todas as unidades seria fastidiosa e sem relevo especial.

6.7 Implementação

O ficheiro é composto pelo *header* e pelo *source*. No *header* estão definidos portos de entrada e saída, processos dependentes de eventos e sinais internos entre processos. No *source* é descrito o funcionamento da máquina de estados e respectivos processamentos de sinais dentro de cada estado.

```
// IDLE_SHAPER_FSM_D3_3_2.h: header file

#include <systemc.h>

SC_MODULE (IDLE_SHAPER_FSM_D3_3_2)
{
enum states {RESET=0X0, CALC_1=0X1, CALC_2=0X3, CALC_3=0X2};

sc_signal<states> curr_state, nxt_state;

void prc_comb_logic();
void prc_state();

   // ports
   sc_in<bool> Clock;
   sc_in<bool> Reset;
   sc_in<sc_uint<10> > TICK;
   sc_in<sc_uint<14> > SpaceInPTQ;
   sc_in<sc_uint<3> > lineRate;
   sc_in<sc_uint<14> > idleThreshold;
   sc_in<sc_uint<32> > currentTime;
   sc_in<sc_uint<6> > creditI;
   sc_in<bool> sendI_in;
   sc_in<sc_uint<6> > creditI_in;
   sc_in<sc_uint<32> > tickTime_in;
   sc_in<sc_uint<24> > addRateI_in;
   sc_out<bool> sendI_out;
   sc_out<bool> enSendI;
   sc_out<sc_uint<6> > creditI_out;
   sc_out<bool> enCreditI;
   sc_out<sc_uint<32> > tickTime_out;
   sc_out<bool> enTickTime;
   sc_out<sc_uint<24> > addRateI_out;
   sc_out<bool> enAddRateI;
   sc_out<sc_uint<3> > State_Out;

// default constructor
            SC_CTOR(IDLE_SHAPER_FSM_D3_3_2)
            {
                        // process declarations

                        SC_METHOD (prc_state);
                        sensitive_pos << Clock;
                        sensitive_pos << Reset;

                        SC_METHOD (prc_comb_logic);
                        sensitive << curr_state;
            }
}; // end module IDLE_SHAPER_FSM_D3_3_2
```

```
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
// IDLE_SHAPER_FSM_D3_3_2.cpp: source file

#include "IDLE_SHAPER_FSM_D3_3_2.h"
#include "/home/mosorio/ccss/SIRAC/sirac_v1/IEEE_802_17/rpr/mac/GLOBAL_VARIABLES.h"

//////////////////////////////////////////////////State Transitions Process/////////////////////////////////////////////////////////////////////////////////////////////////////////
void IDLE_SHAPER_FSM_D3_3_2::prc_state()
{
	if (Reset.read() == true)
	{
		curr_state = RESET;
	}
	else
	{
		curr_state = nxt_state.read();
	}
}
//////////////////////////////////////////Combinational Logic Transitions Process/////////////////////////////////////////////////////////////////////////////////////////////////////////
void IDLE_SHAPER_FSM_D3_3_2::prc_comb_logic()
{
	nxt_state = CALC_1;
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
	switch (curr_state.read())
	{
////////////////////////////////////////////////////////////////RESET/////////////////////////////////////////////////////////////////////////////////////////////////////////
		case RESET:

			State_Out = RESET;

			sendI_out = false;
			enSendI = false;

			creditI_out = 0X0;
			enCreditI = false;

			tickTime_out = 0X0;
			enTickTime = false;

			addRateI_out = 0X0;
			enAddRateI = false;

			nxt_state = CALC_1;

		break;
////////////////////////////////////////////////////////////////CALC_1/////////////////////////////////////////////////////////////////////////////////////////////////////////
case CALC_1:

State_Out = CALC_1;

addRateI_out = addRateI_in.read();
enAddRateI = false;

if ((currentTime.read() - tickTime_in.read()) >= TICK.read())
{
	sendI_out = false;
	enSendI = true;

	tickTime_out = currentTime.read();
	enTickTime = true;

	if ((IDLE_SIZE+IDLE_SIZE) > (creditI.read() + (addRateI_in.read()*(currentTime.read() - tickTime_in.read()))))
	//if (hiLimitI > (creditI.read() + (addRateI*(currentTime - tickTime)));
	{
		creditI_out = (creditI.read() + (addRateI_in.read()*(currentTime.read() - tickTime_in.read())));
		enCreditI = true;
	}
	else //if (hiLimitI <= (creditI.read() + (addRateI*(currentTime - tickTime)));
	{
		creditI_out = (IDLE_SIZE+IDLE_SIZE);//creditI_out = hiLimitI;
		enCreditI = true;
	}
```

```
		nxt_state = CALC_2;
}
else //if ((currentTime.read() - tickTime_in.read()) < TICK.read())
{
		sendI_out = (creditI_in.read() >= IDLE_SIZE);//sendI_tmp = (creditI_in >= loLimitI);
		enSendI = true;

		tickTime_out = tickTime_in.read();
		enTickTime = true;

		nxt_state = CALC_3;
}
break;
////////////////////////////////////////////////////////////////////////CALC_2////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
case CALC_2:

State_Out = CALC_2;

addRateI_out = 0X0;
enAddRateI = false;

if ((currentTime.read() - tickTime_in.read()) >= TICK.read())
{
		sendI_out = false;
		enSendI = true;

		tickTime_out = currentTime.read();
		enTickTime = true;

		if ((IDLE_SIZE+IDLE_SIZE) > (creditI.read() + (addRateI_in.read()*(currentTime.read() - tickTime_in.read()))))
		//if (hiLimitI > (creditI.read() + (addRateI*(currentTime - tickTime)));
		{
			creditI_out = (creditI.read() + (addRateI_in.read()*(currentTime.read() - tickTime_in.read())));
			enCreditI = true;
		}
		else //if (hiLimitI <= (creditI.read() + (addRateI*(currentTime - tickTime)));
		{
			creditI_out = (IDLE_SIZE+IDLE_SIZE);//creditI_out = hiLimitI;
			enCreditI = true;
		}

		nxt_state = CALC_1;
}
else //if ((currentTime.read() - tickTime_in.read()) < TICK.read())
{
		sendI_out = (creditI_in.read() >= IDLE_SIZE);//sendI_tmp = (creditI_in >= loLimitI);
		enSendI = true;

		tickTime_out = tickTime_in.read();
		enTickTime = true;

		nxt_state = CALC_3;
}
break;
////////////////////////////////////////////////////////////////////////CALC_3////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
case CALC_3:

State_Out = CALC_3;

enAddRateI = true;

sendI_out = (creditI_in.read() >= IDLE_SIZE);//sendI_tmp = (creditI_in >= loLimitI);
enSendI = true;

tickTime_out = tickTime_in.read();
enTickTime = false;

creditI_out = creditI_in.read();
enCreditI = false;

if (SpaceInPTQ.read() > idleThreshold.read())
{
		if (lineRate.read() == STM_1)
```

```
		{
			addRateI_out = ((LINE_RATE_STM_1/8)*0X1F4)/0XF4240;//(lineRate/8)*500/1000000 -> 500ppm
                                        }
			else if (lineRate.read() == STM_4)
                                        {
			addRateI_out = ((LINE_RATE_STM_4/8)*0X1F4)/0XF4240;//(lineRate/8)*500/1000000 -> 500ppm
		}
		else if (lineRate.read() == ETH_1G)
		{
			addRateI_out = ((LINE_RATE_1G_ETH/8)*0X1F4)/0XF4240;//(lineRate/8)*500/1000000 -> 500ppm
		}
		else if (lineRate.read() == STM_16)
		{
			addRateI_out = ((LINE_RATE_STM_16/8)*0X1F4)/0XF4240;//(lineRate/8)*500/1000000 -> 500ppm
		}
		else if (lineRate.read() == STM_64)
		{
			addRateI_out = ((LINE_RATE_STM_64/8)*0X1F4)/0XF4240;//(lineRate/8)*500/1000000 -> 500ppm
		}
		else //if (lineRate.read() == ETH_10G)
		{
			addRateI_out = ((LINE_RATE_10G_ETH/8)*0X1F4)/0XF4240;//(lineRate/8)*500/1000000 -> 500ppm
		}

		nxt_state = CALC_1;
}
else //if (SpaceInPTQ.read() <= idleThreshold.read())
{
		if (lineRate.read() == STM_1)
		{
			addRateI_out = ((LINE_RATE_STM_1/8)*0XFA)/0XF4240;//(lineRate/8)*250/1000000 -> 250ppm
		}
		else if (lineRate.read() == STM_4)
		{
			addRateI_out = ((LINE_RATE_STM_4/8)*0XFA)/0XF4240;//(lineRate/8)*250/1000000 -> 250ppm
		}
		else if (lineRate.read() == ETH_1G)
		{
			addRateI_out = ((LINE_RATE_1G_ETH/8)*0XFA)/0XF4240;//(lineRate/8)*250/1000000 -> 250ppm
		}
		else if (lineRate.read() == STM_16)
		{
			addRateI_out = ((LINE_RATE_STM_16/8)*0XFA)/0XF4240;//(lineRate/8)*250/1000000 -> 250ppm
		}
		else if (lineRate.read() == STM_64)
		{
			addRateI_out = ((LINE_RATE_STM_64/8)*0XFA)/0XF4240;//(lineRate/8)*250/1000000 -> 250ppm
		}
		else //if (lineRate.read() == ETH_10G)
		{
			addRateI_out = ((LINE_RATE_10G_ETH/8)*0XFA)/0XF4240;//(lineRate/8)*250/1000000 -> 250ppm
		}

		nxt_state = CALC_1;
}
break;
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
}
```

Este módulo de *Rate Control*, contém unidades com máquinas de estado simples. Estas unidades apenas fazem processamento e geram sinais, mediante eventos em sinais de entrada.

A filosofia de desenvolvimento de outras unidades constituintes do sistema, baseou-se essencialmente na elaboração de máquinas de estados interligadas por unidades de memória (FIFO) para fazer a comunicação de tramas entre essas máquinas de estado. Este sistema é constituído por diversas etapas nas quais as tramas que circulam pelo mesmo, são processadas consoante vão entrando e saindo de cada unidade. Por exemplo uma máquina de estados que receba uma trama, faz o processamento dessa trama e porventura envia-a para a máquina de

estados seguinte, ou armazena informação necessária para ser usada por outra máquina de estados algures no sistema.

Este sistema foi desenvolvido numa filosofia de *cut-through*, isto é, as tramas que entram são processadas, por cada unidade, assim que houver informação suficiente para que esse processamento possa ocorrer. Enquanto as tramas são processadas, as mesmas continuam a entrar nas unidades ou são mantidas em espera em filas lógicas à entrada de cada unidade. Posteriormente, são enviadas para uma fila lógica à saída e assim que se retirem completamente da máquina de estados, esta já estará preparada para receber outra trama que se encontre em espera na fila lógica de entrada.

Este tipo de filosofia permite que haja um funcionamento concorrente de todas as unidades que constituem este sistema, ao contrário de um sistema que processasse uma trama de cada vez, permanecendo tramas em espera à entrada do mesmo.

6.7.1 Modelo de teste

Na figura 6.10 está ilustrado o módulo de teste desenvolvido no CoCentricSystemStudio usado para simular a máquina de estados de *Idle Shaper*.

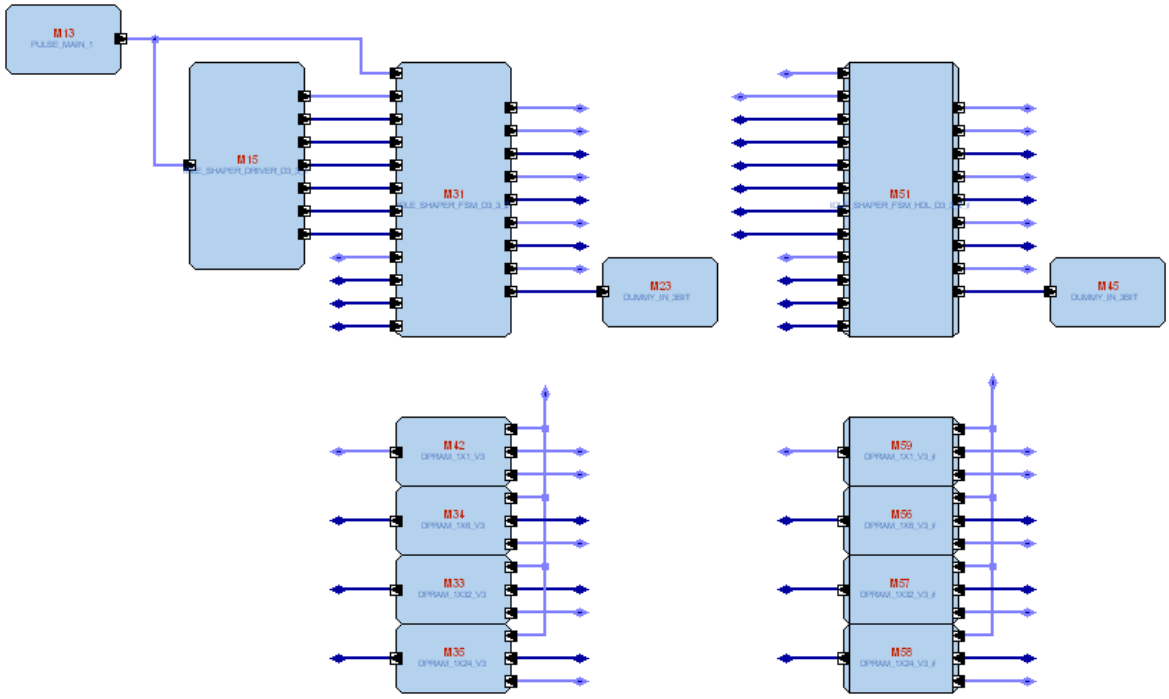


Figura 6.10 – Módulos de simulação e de co-simulação do *Idle Shaper*

Na figura 6.10 estão ilustrados os dois métodos de simulação elaborados para a unidade de *Idle Shaper*. O módulo M15 é o gerador de vectores descrito em SystemC e que vai servir para testar os módulos M31 e M51. O módulo M31 é a máquina de estados e respectivo processamento de dados do *Idle Shaper*, em SystemC. O módulo M51 é o correspondente ao módulo M31 mas compilado para Verilog. Para o bloco M31 são usados blocos de memória (M34, M33, M35, M42) em SystemC e os seus homólogos em Verilog são os blocos de memória M56, M57, M58 e M59, que servem para o bloco M51. É usado um bloco de geração de relógio em SystemC, M13, com um relógio de 33MHz. São usados dois blocos M23 e M45 como terminadores neutros que servem apenas para não gerar erros de linhas não ligadas, durante a simulação.

Abaixo está descrita a implementação completa em SystemC do módulo de geração de vectores de teste M15 (figura 6.10) para a máquina de estados do *Idle Shaper*.

```
// IDLE_SHAPER_DRIVER_D3_3_1.h: header file

#include <systemc.h>

SC_MODULE (IDLE_SHAPER_DRIVER_D3_3_1)
{

sc_uint<14> SpaceInPTQ_tmp;

sc_uint<32> currentTime_tmp;

void prc_IDLE_SHAPER_DRIVER();

sc_uint<32> k;
sc_uint<8> flag;

// ports
   sc_in<bool> Clock;
   sc_out<bool> Reset;
   sc_out<sc_uint<10> > TICK;
   sc_out<sc_uint<14> > SpaceInPTQ;
   sc_out<sc_uint<3> > lineRate;
   sc_out<sc_uint<14> > idleThreshold;
   sc_out<sc_uint<32> > currentTime;
   sc_out<sc_uint<6> > creditI;

// default constructor
            SC_CTOR(IDLE_SHAPER_DRIVER_D3_3_1)
            {
            // process declarations

                        SC_METHOD (prc_IDLE_SHAPER_DRIVER);
                        sensitive_pos << Clock;

            }

}; // end module IDLE_SHAPER_DRIVER_D3_3_1

// IDLE_SHAPER_DRIVER_D3_3_1.cpp: source file

#include "IDLE_SHAPER_DRIVER_D3_3_1.h"
#include "/home/mosorio/ccss/SIRAC/sirac_v1/IEEE_802_17/rpr/mac/GLOBAL_VARIABLES.h"

//////////////////State Transitions Process/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

void IDLE_SHAPER_DRIVER_D3_3_1::prc_IDLE_SHAPER_DRIVER()
{
            k = k+4;   //Counter increments by 4 bytes at each clock period

            currentTime_tmp = k;
            currentTime = currentTime_tmp;

            idleThreshold = SIZE_PTQ - JUMBO_MAX;
            lineRate = ETH_1G;

            SpaceInPTQ_tmp = SIZE_PTQ - 100*k;
            SpaceInPTQ = SpaceInPTQ_tmp;

            creditI = 2*IDLE_SIZE - k/4;

            TICK = 10*IDLE_SIZE;

            if (flag==5)            //Use of a dummy variable.
                                    //If this variable is not equal to 5, it ends the cycle and equals to 5.
                                    //Then the cycle starts from the begining.
            {
                        if (k == 12)
```

```
                {
                        Reset = false;
                }
                else if (k == 24)
                {
                        Reset = true;
                }
                else if (k == 36)
                {
                        Reset = false;
                }
                else if (k == 50000)
                {
                        k=0;
                }
        }
        else if (flag != 5)
        {
                k=0;
                flag=5;
        }
}
```

6.7.2 Resultados

Nas figuras 6.11-a e 6.11-b estão ilustrados os resultados obtidos nas simulação e co-simulação simultâneas, usando a ferramenta VirSim integrada no ambiente do CoCentricSystemStudio, tal como lustrado na figura 6.12. Em ambas as figuras, na coluna esquerda onde estão indicados os sinais correspondentes às ondas visualizadas, os sinais com relevo a negro correspondem aos da co-simulação SystemC/Verilog e os sinais com relevo a cinzento correspondem aos da simulação SystemC.

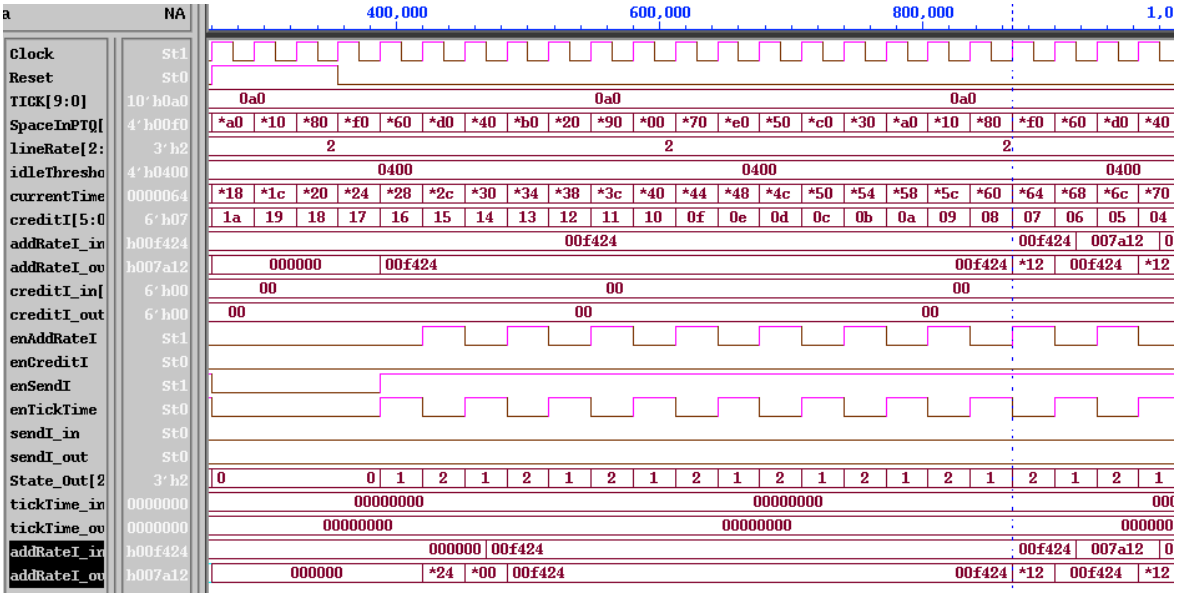


Figura 6.11-a – Resultados obtidos nas simulação e co-simulação simultâneas

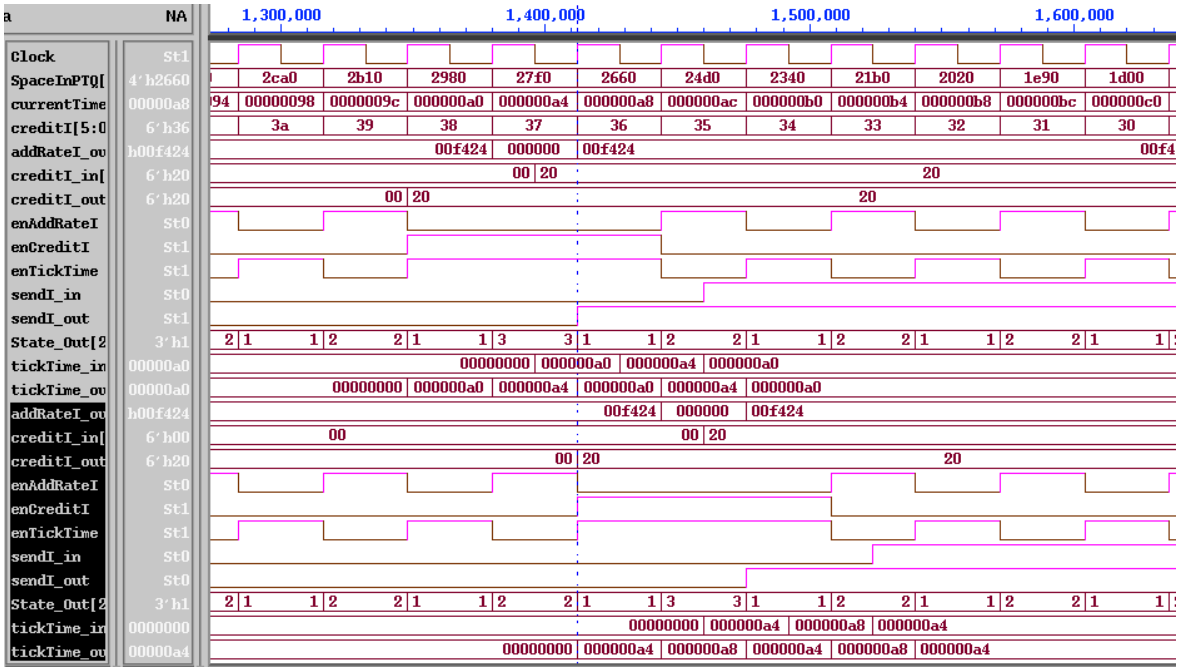


Figura 6.11-b – Resultados obtidos nas simulação e co-simulação simultâneas

Na figura 6.12 estão ilustrados os resultados da simulação do *Idle Shaper* implementado num FPGA, usando um analisador lógico ligado a uma placa de teste do FPGA.

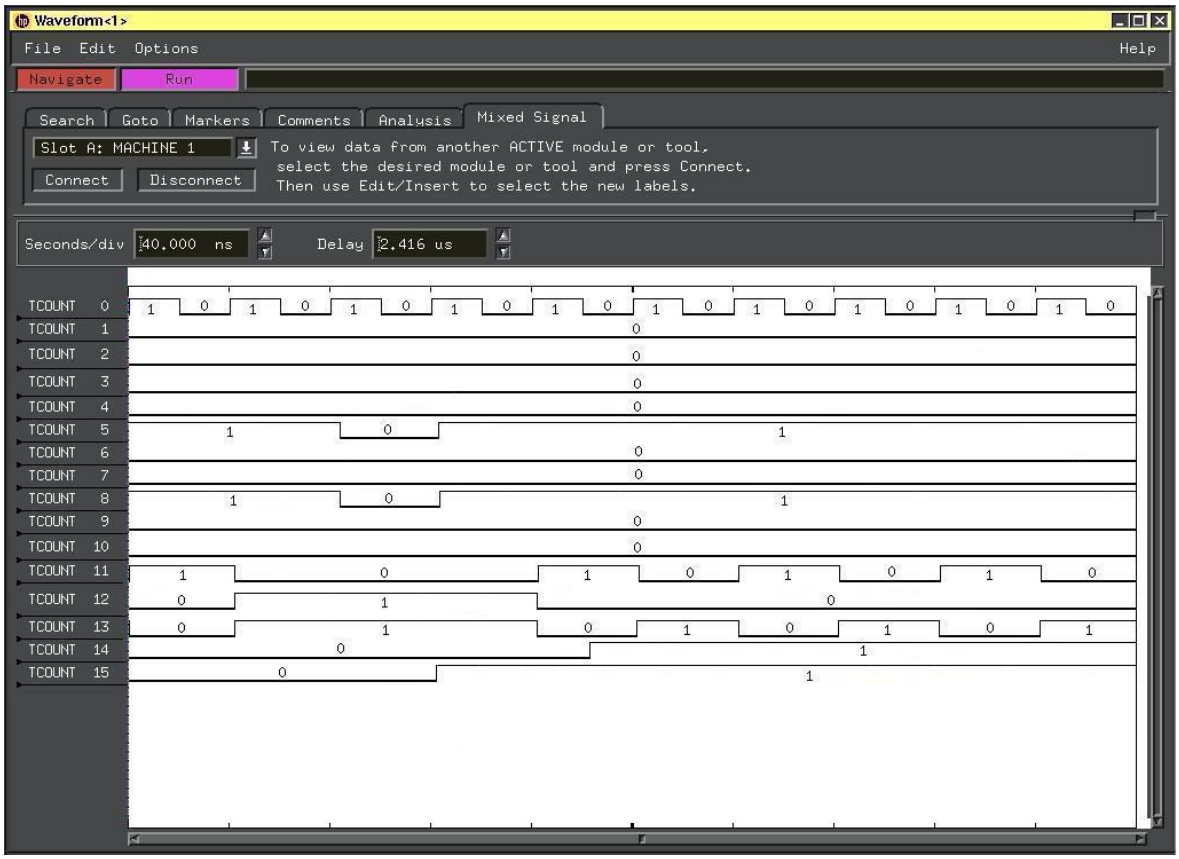


Figura 6.12 – Resultados obtidos na implementação em FPGA

Como se pode observar nas figuras 6.11-a e 6.11-b, as ondas não são exactamente iguais tanto para a simulação e co-simulação, no entanto os resultados são precisamente os mesmos. Isto é uma característica da ferramenta de simulação que usa dois simuladores distintos para cada tipo de linguagem. Isto resulta num desfasamento dos sinais quando estes passam de um ambiente Verilog para SystemC e vice-versa e também em diferentes valores nas saídas dos blocos, quando estes são inicializados.

Os resultados obtidos pelo analisador lógico na implementação em FPGA a uma frequência de 25MHz, estão ilustrados na figura 6.12. As correspondências de sinais entre as figuras 6.11-b e 6.13, são respectivamente:

Clock → TCOUNT 0;

AddRateI_out[9] → TCOUNT 1;

AddRateI_out[8] → TCOUNT 2;

AddRateI_out[7] → TCOUNT 3;

AddRateI_out[6] → TCOUNT 4;

AddRateI_out[5] → TCOUNT 5;

AddRateI_out[4] → TCOUNT 6;

AddRateI_out[3] → TCOUNT 7;

AddRateI_out[2] → TCOUNT 8;

AddRateI_out[1] → TCOUNT 9;

AddRateI_out[0] → TCOUNT 10;

enAddRateI → TCOUNT 11;

enCreditI → TCOUNT 12;

enTickTime → TCOUNT 13;

sendI_in → TCOUNT 14;

sendI_out → TCOUNT 15;

Como se pode observar nas figuras 6.11-b e 6.12, as ondas são exactamente iguais tanto para a simulação como para a implementação física num FPGA.

Isto permite concluir a exequibilidade deste método de desenvolvimento e implementação.

Os restantes blocos foram verificados de forma semelhante. A sua integração completa num sistema encontra-se a decorrer.

7 Conclusões

Nem o SONET/SDH nem a Ethernet foram projectados para as necessidades do ambiente da MAN. O SONET/SDH usa técnicas das camadas 1 e 2 para gestão da largura de banda e para protecção de serviço. Os comutadores Ethernet são baseados em Ethernet *bridging* ou em encaminhamento IP, para fazer a gestão da largura de banda, e para fornecer serviços de manutenção e protecção. Consequentemente, a rede é subaproveitada (no caso SONET/SDH) ou não determinística (no caso de simples comutação Ethernet). Nas MAN a especificidade da característica de comutação de circuitos do actual protocolo SONET/SDH, é excelente para o tráfego ATM. No entanto esta característica é também uma deficiência quando é necessário transportar tráfego em pacotes. Os produtos Ethernet para comutação e encaminhamento têm falta de robustez, escalabilidade e serviços determinísticos necessários para a infra-estrutura da rede pública de telecomunicações. A tecnologia RPR, investigada e normalizada pelo grupo 802.17 do IEEE, está preparada para obviar estes problemas.

A arquitectura RPR é baseada numa topologia em anel duplo, onde os nós de comutação de pacotes estão ligados a dois outros nós adjacentes através de um par de fibras. O MAC do RPR, assegura que os pacotes são eficientemente transportados num meio partilhado, suportado por estes dois anéis (*ringlet0* e *ringlet1*). A camada do MAC pode garantir a qualidade de serviço e uma gestão justa da largura de banda, controlando o acesso ao meio e decidindo a ordem do seu uso. Para além disso a camada MAC executa um mecanismo de protecção de serviço para proteger de falhas no anel, e um algoritmo para prevenir o congestionamento, o que permite ao sistema, operar perto do máximo da sua capacidade enquanto assegura a qualidade de serviço a todos os serviços configurados.

No cenário económico actual, é demasiado grande o esforço que os pequenos e médios fornecedores de serviços de telecomunicações, fazem para usar uma nova tecnologia sem saberem de antemão os seus benefícios efectivos. Portanto essas tecnologias devem ser económicas e fiáveis com intuito de alcançar tais expectativas.

Este documento abordou uma solução de desenvolvimento de baixo custo de um módulo RPR, para ser usado nos anéis SDH existentes e com interfaces locais para Fast e Gigabit Ethernet, com o intuito de explorar optimamente estas duas tecnologias.

A estrutura da MAN implementada com RPR usa essencialmente os anéis SONET/SDH, que são uma infra-estrutura legada que deve ser devida e optimamente reaproveitada – no melhor dos interesses dos operadores de telecomunicações. Por outro lado, a Ethernet está a tornar-se no principal meio para transferência de dados. O RPR apresenta uma evolução tecnológica emergente que vai permitir que a tecnologia base das redes metropolitanas de telecomunicações seja mantida por um prazo maior do que aquele previsto para a sua caducidade, acrescentando-lhe uma mais valia tecnológica.

Para alcançar os objectivos pretendidos, foram desenvolvidos os módulos constituintes do MAC do protocolo RPR, baseados num FPGA. Para lidar com a complexidade do sistema, foi feita uma abordagem de desenvolvimento usando uma linguagem *open-source* de descrição de sistemas, explorando toda a versatilidade existente numa metodologia de prototipagem em FPGA.

Esta metodologia de prototipagem em FPGA poderá ser mais vantajosa relativamente a uma solução desenvolvida totalmente com um *Network Processor*, pelo facto de se poder implementar em *hardware* as partes mais críticas em termos de velocidade e poder implementar em *firmware* funções menos críticas para o desempenho do sistema. Da mesma forma poderá ser vantajosa em termos económicos, dado que é possível baixar muito os custos de um FPGA que já esteja completamente prototipada e daí implementar um sistema com um Fast ASIC ou com um FPGA definitivamente programado.

Os resultados globais do método de desenvolvimento adoptado, permitiram fazer simulações e síntese ao nível dos módulos constituintes do sistema e consequentemente permitirão fazer a simulação e síntese do sistema completo, o que também fará decrescer o custo de todo o desenvolvimento e dos testes do mesmo.

Este trabalho de mestrado irá ter continuidade para serem concluídas a parte das interfaces com o cliente e o anel, e também a integração total do sistema num circuito lógico programável. Assim sendo, a curto prazo o trabalho desenvolvido irá ser integrado num protótipo o qual irá servir para construir uma rede teste de telecomunicações, num anel constituído por outras estações RPR iguais. Com uma pequena rede piloto constituída por alguns protótipos, será possível testar e demonstrar todas as capacidades e vantagens competitivas da tecnologia RPR face às tecnologias convencionais.

Para que seja apresentado um sistema compatível com as actuais redes de telecomunicações, este terá de cumprir com os requisitos específicos dos equipamentos de telecomunicações já estabelecidos no mercado, estabelecendo-se interfaces universais com os mesmos mas apresentando um novo modelo de abordagem para a gestão de tráfego de dados nas redes metropolitanas de telecomunicações, o MAC RPR.

8 Referências

[1] A. Kaufmann, “POS – Packet over SONET/SDH Pocket Guide”, Acterna Eningen Gmbh.

[2] IEEE Std 802-2002, IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture.

[3] IEEE Std 802.1D-2004, Information Technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Common specifications—Part 3: Media Access Control (MAC) Bridges.

[4] IEEE Std 802.1Q-1998, IEEE Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks.

[5] IEEE Std 802.3-2000, Information Technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements—Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications.

[6] IEEE Std 802.3ae-2002, IEEE Standard for Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements—Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications—Amendment: Media Access Control (MAC) Parameters, Physical Layers, and Management Parameters for 10 Gb/s Operation.

[7] IEEE Std 802.17/Draft 3.3, Information Technology—Telecommunications and information exchange between systems—Local and metropolitan area networks— Specific requirements—Part 17: Resilient Packet Ring (RPR) access method & physical layer specifications.

[8] IEEE Std 802.17a-2004, Information Technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Common specifications—Part 3: Media Access Control (MAC) Bridges—Supplement: 802.17 MAC revisions to IEEE Std 802.1D-2004.

[9] IETF RFC 1662: PPP in HDLC Like Framing, W. Simpson, July 1994.

[10] IETF RFC 2615: PPP over SONET/SDH, A. Malis, W. Simpson, June 1999.

[11-a] IETF RFC 791 Internet Protocol. J. Postel. Sep-01-1981.

[11-b] IETF RFC 2460 Internet Protocol, Version 6 (IPv6) Specification. S. Deering, R. Hinden. December 1998.

[12] IETF RFC 792 Internet Control Message Protocol, J. Postel, September 1981.

[13] ITU-T Recommendation X.210, Data Networks and Open Systems Communications.

[14] ITU-T Recommendation G.707, Network Node Interface for the Synchronous Digital Hierarchy (SDH).

[15] ITU-T Recommendation G.783, Characteristics of Synchronous Digital Hierarchy (SDH) Equipment Functional Blocks.

[16] ITU-T Recommendation G.7041, Generic Framing Procedure.

[17] ITU-T Recommendation X.85/Y.1321, IP over SDH using LAPS.

[18] J. Bhasker, “A SystemC Primer”, Star Galaxy, Allentown, PA, 2002, ISBN 0-9650391-8-8.

[19] OIF System Packet Interface Level 3 (SPI-3): OC48 System Interface for Physical and Link Layer Devices. Implementation agreement: OIF-SPI3-01.0, June 2004.

[20] OIF System Packet Interface Level 4 Phase 1 (SPI-4.1): OC192 System Interface for Physical and Link Layer Devices. Implementation agreement: OIF-SPI4-01.0, April 2001.

[21] OIF System Packet Interface Level 4 Phase 2 (SPI-4.2): OC192 System Interface for Physical and Link Layer Devices. Implementation agreement: OIF-SPI4-02.0, January 2001.

[22] Open SystemC Initiative. Functional Specification for SystemC™ 2.0. Update for SystemC 2.0.1, Version 2.0-Q April 2002 (//www.systemc.org/).

[23] P. J. Ashenden, “The designer’s guide to VHDL”, 2nd Ed., Morgan Kaufmann, 2002.

[24] Synopsys Inc., CoCentric™ System Studio, version 2003.12-SP2.

[25] Synopsys Inc. CoCentric™ SystemC Compiler – RTL User and Modeling Guide, version U-2003.06, June 2003.

[26] “SystemC User’s Guide”, Version 2.0.1, 2002.

[27] S. Palnitkar, “Verilog HDL – A guide to digital design and synthesis”, Prentice Hall, 1996.

[28] Synopsys Inc., “FPGA Compiler II / FPGA Express Verilog HDL Reference Manual”, Version 1999.05.

[29] Synopsys Inc., “Design Compiler Reference Manual”.

[30] Synopsys Inc., VCS-MX 7.1.1.

[31] Synopsys Inc. Describing Synthesizable RTL in SystemC™. Version 1.2. November 2002.

[32] Synplicity Inc., Synplify Pro® 7.6.1.

[33] Synopsys Inc., CoCentric™ SystemC Compiler, version U-2003.06.

[34] T.Grotker, S.Liao, G. Martin, Stuart Swan, “System Design with SystemC”, Kluwer Academic, 2002.

[35] Xilinx® Integrated Software Environment (ISE 6.2i).

[36] Ferreira da Rocha, Doutor José Rodrigues – “Redes Ópticas - redes da primeira geração – o SONET/SDH”, apontamentos da disciplina de Redes Ópticas, Universidade de Aveiro.

Anexo A - Formato das tramas RPR

Para sistemas de comunicação de dados que usam o MAC do RPR, são especificados os seguintes formatos de trama:

- trama de dados;
- trama de controlo;
- trama de fairness;
- trama de idle.

O formato da trama do RPR não inclui qualquer delimitação da camada 1 i.e. não há quaisquer sequências de início ou fim de trama.

A.1 – Tramas de dados

As tramas de dados são identificadas pelo campo de 2 bits, *ft* no byte *baseControl* (fig. A.1), igual a FT_DATA.

Quando os campos *daExtended* e *saExtended* são incluídos, a trama de dados é classificada de trama de dados estendida. Quando os campos de endereço estendido não estão incluídos, a trama de dados é classificada de trama de dados básica.

Para anéis onde tenham sido negociadas tramas regulares, o comprimento máximo de transferência (MTU) é definido por REGULAR_MAX; para tramas jumbo, o comprimento máximo de transferência (MTU) é definido por JUMBO_MAX. O tamanho máximo negociado para um anel é definido na variável, *mtuSize*.

Nome	Valor	Descrição
DATA_MIN	24	A mais pequena trama de dados consiste num cabeçalho e num *payload* de 2 bytes, consistindo apenas no campo *protocolType* e num *trailer*.
EXT_HDR_SIZE	12	Número de bytes adicionais no cabeçalho de uma trama de dados estendida, para além dos do cabeçalho de uma trama básica de dados. Os bytes extra consistem nos campos *daExtended* e *saExtended*.
REGULAR_MAX	1616	A maior trama regular de dados tem 1500 bytes no campo *serviceDataUnit*, e até 92 bytes reservados.
JUMBO_MAX	9216	A maior trama de dados tem 9100 bytes no campo *serviceDataUnit*, e até 92 bytes reservados.

Tabela A.1 – Condicionamentos ao tamanho da trama de dados

O número de bytes reservados depende se é usada a trama básica ou a trama estendida. A trama básica tem 92 bytes reservados. A trama estendida tem 80 bytes reservados. Os bytes reservados em REGULAR_MAX e JUMBO_MAX não estão incluídos na trama transmitida ou recebida. Existem apenas na definição do comprimento e estão projectados para permitirem evoluções futuras nos cabeçalhos. O campo *serviceDataUnit* não pode usar os bytes reservados, e não pode crescer para além dos 1500 bytes para uma trama regular ou 9100 bytes para uma trama jumbo.

a) formato da trama básica

1 byte	ttl	cabeçalho
1 byte	baseControl	
6 bytes	da	
6 bytes	sa	
1 byte	ttlBase	
1 byte	extendedControl	
2 bytes	hec	
2 bytes	protocolType	payload
N bytes	serviceDataUnit	
4 bytes	fcs	trailer

b) formato da trama estendida

1 byte	ttl	cabeçalho
1 byte	baseControl	
6 bytes	da	
6 bytes	sa	
1 byte	ttlBase	
1 byte	extendedControl	
2 bytes	hec	
6 bytes	daExtended	payload
6 bytes	saExtended	
2 bytes	protocolType	
N bytes	serviceDataUnit	
4 bytes	fcs	trailer

Figura A.1 – Formatos da trama de dados

ttl: Campo de 8 bits (*time to live*) que especifica o número máximo espectável de estações (*hops*) que a trama percorre antes de atingir o seu destino. Este campo providencia um mecanismo para assegurar que as tramas não circulem eternamente no anel;

baseControl: Campo de controlo básico de 8 bits;

da: Campo de 48 bits (*destination address*) que especifica a estação ou estações para as quais a trama é destinada. O campo *da* contém tanto o endereço MAC (48 bits) de um indivíduo ou de um grupo como definido no IEEE Std 802-2001;

sa: Campo de 48 bits (*source address*) que especifica a estação de origem da trama enviada. O campo *sa* contém o endereço MAC (48 bits) de um indivíduo como definido no IEEE Std 802-2001. O endereço contido neste campo é sempre o endereço MAC da estação local que transmite a trama para dentro do anel;

ttlBase: Campo de 8 bits que é configurado com o valor inicial do campo *ttl*, quando da transmissão de uma trama de dados;

extendedControl: Campo de 8 bits de controlo estendido;

hec: Campo de 16 bits (*header error check*) que é o *checksum* do cabeçalho. O *hec* é calculado sobre os campos *ttl*, *baseControl*, *da*, *sa*, *ttlBase* e *extendedControl*;

daExtended: Campo de 48 bits (*destination address extended*) que especifica a estação ou estações para as quais a trama é destinada e contém o mesmo valor que o campo *da*. O campo *daExtended* contém o endereço MAC (48 bits) de um indivíduo ou de um grupo como definido no IEEE Std 802-2001;

saExtended: Campo de 48 bits (*source address extended*) que especifica a estação de origem da trama enviada. O campo *sa* contém o endereço MAC (48 bits) de um indivíduo como definido no IEEE Std 802-2001. O endereço contido no campo *saExtended* é usualmente o endereço MAC de uma estação não local e está contido neste campo pois o valor do campo *sa* é sempre o endereço MAC da estação local que transmite a trama para dentro do anel. Os campos *daExtended* e *saExtended* estão presentes apenas quando o bit *ef* é 1.

protocolType: Campo de 16 bits contido dentro do *payload*. Quando o valor do *protocolType* é maior ou igual a 1536 (600_{16}) o campo de *protocolType* indica o tipo de protocolo do MAC do cliente, seleccionado de valores designados pelo IEEE Type Field Register. Quando este valor é menor do que 1536 (0_{16} – $5FF_{16}$), o *protocolType* é interpretado como sendo o comprimento da trama. O comprimento e interpretação de tipo deste campo são mutuamente exclusivos;

serviceDataUnit: Campo de comprimento variável que contém a SDU (Service Data Unit) providenciada pelo cliente;

fcs: Campo de 32 bits (*frame check sequence*) que é o CRC (Cyclic Redundancy Check) da trama. O *fcs* CRC32 da trama é calculado começando pelo byte a seguir ao *hec* até ao final do *payload*.

A.2 – Tramas de controlo

As tramas de controlo são identificadas pelo campo de 2 bits, *ft*, igual a FT_CONTROL. Uma trama de controlo pode ser uma trama *broadcast* ou *unicast*.

Nome	Valor	Descrição
CONTROL_MIN	24	A trama de controlo mais pequena consiste num cabeçalho, num *payload* de 2 bytes que inclui os *controlType*, *controlVersion*, um *ControlDataUnit* de 0 bytes e um *trailer*.
CONTROL_MAX	1616	A trama de controlo maior tem um campo de *controlDataUnit* de 1500 bytes e 92 bytes reservados.

Tabela A.2 – Condicionamentos ao tamanho da trama de controlo

Os 92 bytes reservados em CONTROL_MAX não estão incluídos na trama transmitida ou recebida. Existem apenas na definição do comprimento, e estão projectados para permitirem evoluções futuras nos cabeçalhos. O campo *controlDataUnit* não pode usar os bytes reservados, e não pode crescer para além dos 1500 bytes.

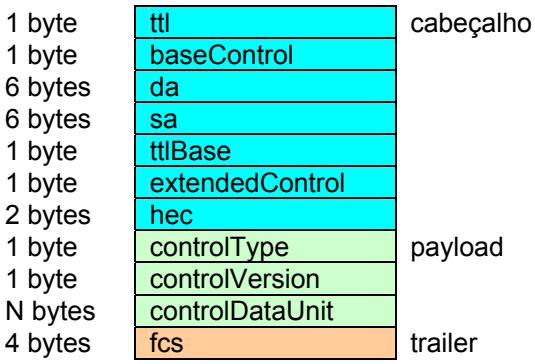


Figura A.2 – Formato da trama de controlo

ttl: Mesma definição que da trama de dados. O valor inicial deste campo é especificado posteriormente, nas descrições das tramas de controlo;

baseControl: Campo de 8 bits com opções para o processamento das tramas de controlo;

da: Campo de 48 bits. Adicionalmente, o campo *da* numa trama de controlo está condicionado a ser um endereço pertencente a uma estação local do anel, ou um endereço de *broadcast*;

sa: Campo de 48 bits com a mesma definição que da trama de dados;

ttlBase: Campo de 8 bits com a mesma definição que da trama de dados;

extendedControl: Campo de controlo estendido de 8 bits, definido à frente, e condicionado como especificado na tabela A.3;

Campo	Valor	Descrição
ef	0	São geradas apenas de estações locais do anel.
fi	FI_NONE	Não são *flooded* para os clientes
ps	0	São enviadas inicialmente sem terem já passado a sua origem
so	0	Não têm quaisquer requisitos de ordem estrita ou proibição de duplicação.

Tabela A.3 – Condicionamentos da trama de controlo para valores do sub campo *extendedControl*

hec: Campo de 16 bits com a mesma definição que da trama de dados;

controlType: Campo de 8 bits que identifica o tipo de trama de controlo. A tabela A.4 contém os tipos de tramas de controlo definidos actualmente;

Valor	Nome	Descrição
01_{16}	CT_STATION_ATD	Trama de descoberta de atributo da estação
02_{16}	CT_TOPO_PROT	Trama de protocolo de topologia e protecção
03_{16}	CT_TOPO_CHKSUM	Trama de checksum de topologia e protecção
04_{16}	CT_LRTT_REQ	Trama de pedido da medida do Link Round Trip Time
05_{16}	CT_LRTT_RSP	Trama de resposta à medida do Link Round Trip Time
06_{16}	CT_FDD	Trama de atraso diferencial de fairness
07_{16}	CT_OAM_ECHO_REQ	Trama de pedido de eco OAM
08_{16}	CT_OAM_ECHO_RSP	Trama de resposta ao eco OAM
09_{16}	CT_OAM_FLUSH	Trama de flush OAM
$0A_{16}$	CT_OAM_ORG	Trama de organização particular OAM
-	-	Reservado

Tabela A.4 – Valores do controlType

controlVersion: Campo de 8 bits que é o número da versão associada ao campo de *controlType*. O campo *controlVersion* providencia um meio de identificação de versões futuras das tramas de controlo. Inicialmente, todos os tipos de controlo são versão 0;

controlDataUnit: Campo de tamanho variável que está dependente do valor do campo *controlType*. O campo *controlDataUnit* é especificado nas descrições das tramas de controlo;

fcs: Campo de 32 bits (Frame Check Sequence) com o CRC da trama. O *fcs* CRC da trama de controlo é calculado começando pelo byte a seguir ao *hec* até ao final do *payload*.

A.3 – Tramas de *fairness*

As tramas de *fairness* são identificadas pelo campo de 2 bits, *ft*, igual a FT_FAIRNESS.

O tamanho de uma trama de *fairness* é definido por FAIRNESS_SIZE, e tem o valor de 16 bytes.

A trama de *fairness* é enviada para os MACs vizinhos para fornecer dados ao algoritmo de *fairness* desses MACs.

O formato da trama de *fairness* é diferente dos formatos das tramas de dados e de controlo. As tramas de *fairness* não são enviadas para nós de destino específicos, mas sim para a estação vizinha mais próxima ou divulgadas (*broadcast*) para todo o anel. Portanto, o endereço de destino não contém qualquer informação útil e é omitido. As tramas de *fairness* são mantidas o mais pequenas possível para reduzir o *jitter* noutras tramas, para reduzir o seu consumo efectivo de largura de banda e para minimizar os requisitos de memória para armazenar múltiplas tramas de *fairness* (especialmente quando são usados os algoritmos de *fairness multi choke*).

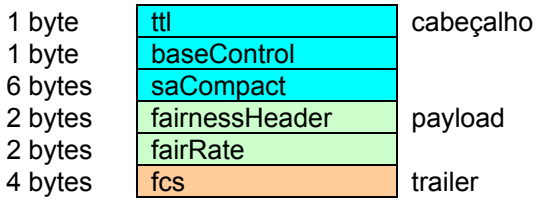


Figura A.3 – Formato da trama de fairness

ttl: Mesma definição que da trama de dados;

baseControl: Campo de 8 bits que afecta as opções de processamento da trama, como definido à frente e condicionado como especificado pela tabela A.5;

Campo	stationProtectionConfig	Valor do Campo	Descrição
fe	-	0	São excluídas dos protocolos de *fairness*
sc	-	CLASS_A0	Usam sempre largura de banda reservada
we	STEERING	0	Em sistemas *Steering* não são *wrap eligible*
	WRAPPING	1	Em sistemas *Wrapping* são *wrap eligible*

Tabela A.5 – Condicionamentos da trama de fairness para valores do sub campo baseControl

saCompact: Campo de 48 bits que contém um endereço MAC de 48 bits individual como definido no IEEE Std 802-2001. Especifica a estação que providenciou os valores contidos nos campos *fairnessHeader* e *fairRate*. Esta estação não é necessariamente a estação que gerou esta trama, que é sempre a estação vizinha a montante. O campo *saCompact* é nomeado diferentemente do campo *sa* por causa da sua diferente posição dentro da trama;

fairnessHeader: Um primeiro campo de 16 bits de *fairness-control*;

fairRate: Um segundo campo de 16 bits de *fairness-control*;

fcs: Campo de 32 bits que contém o CRC da trama de fairness e é calculado começando pelo byte seguido do campo *baseControl* até ao final do *payload*.

A.4 – Tramas de *idle*

As tramas de *idle* são identificadas pelo campo de 2 bits, *ft*, igual a FT_IDLE.

O tamanho de uma trama de *idle* é definido por IDLE_SIZE, e tem o valor de 16 bytes.

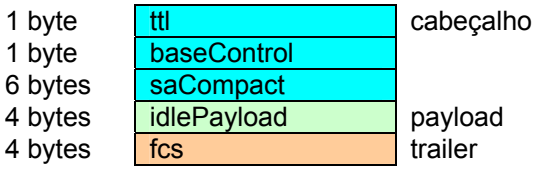


Figura A.4 – Idle frame format

A trama de *idle* é enviada para os MACs vizinhos para ajustar o ritmo de sincronização entre a estação e as estações vizinhas.

O formato de trama para tramas *idle* é diferente do formato de trama para tramas de dados e de outras tramas de controlo. As tramas *idle* não são enviadas para nós específicos de destino, mas são enviadas para a estação vizinha mais próxima. Portanto, o endereço de destino não contém qualquer informação útil e é omitido. As tramas de *idle* são mantidas a um tamanho fixo pequeno, para reduzir o *jitter* noutras tramas e para reduzir o seu consumo efectivo de largura de banda.

ttl: Mesma definição que da trama de dados. O valor inicial é 1;

baseControl: Campo de 8 bits que afecta as opções de processamento da trama, como definido à frente e condicionado como especificado pela tabela A.6;

saCompact: Campo que contém um endereço MAC de 48 bits individual como definido no IEEE Std 802-2001. Identifica a estação que gerou a trama, que é sempre a estação vizinha a montante;

idlePayload: Campo de 32 bits reservado para uso futuro. O campo de *idlePayload* será igualado a zero para implementações desta norma, e ignorado na recepção;

fcs: Campo de 32 bits que contém o CRC da trama de *idle* e é calculado começando pelo byte seguido do campo *baseControl* até ao final do *payload*.

O campo *saCompact* é nomeado diferentemente do campo *sa* por causa da sua diferente posição dentro da trama.

Campo	Valor	Descrição
fe	0	São excluídas dos protocolos de fairness
sc	CLASS_A0	Usam sempre largura de banda reservada

Tabela A.6 – Condicionamentos da trama de idle para valores do sub campo baseControl

A.5 – Campo *baseControl*

O campo *baseControl* de 8 bits consiste em múltiplos sub campos.

MSB					LSB
ri	fe	ft	sc	we	parity

Figura A.5 – Formato do campo *baseControl* para tramas de dados

ri: Bit (*ringlet identifier*) que identifica o *ringlet* no qual a trama foi originalmente transmitida. Os valores de *ri* são definidos na tabela A.7;

fe: Bit (*fairness eligible*) que assinala se a trama está sujeita ao algoritmo de *fairness*. O valor 0 indica que a trama não é *fairness eligible*, enquanto o valor 1 indica que a trama é *fairness eligible*. Os significados definidos pelos 2 bits do campo *sc* e do bit do campo *fe* combinados, estão especificados na tabela A.10. Os 2 bits do campo *sc* e o bit do campo *fe* combinados, são interpretados como definido na tabela A.11;

ft: Campo de 2 bits (*frame type*) que identifica o tipo de trama, definido na tabela A.8;

sc: Campo de 2 bits (*service class*) que identifica a classe de serviço da trama. Os valores de *sc* estão definidos na tabela A.9;

we: Bit (*wrap eligible*) que identifica se a trama é elegível para ser *wrapped* durante uma condição de *wrap*. O valor de 0 indica que a trama não é *wrap eligible*, enquanto o valor de 1 indica que a trama é *wrap eligible*;

parity: Bit que nas tramas de *fairness* e de *idle*, protege os campos de *ttl* e de *baseControl* (porque estas tramas não têm o campo *hec* para protecção do cabeçalho). O bit de *parity* é atribuído tal que o número total de bits 1 nos campos *ttl* e *baseControl*, incluindo o bit de *parity*, é ímpar. Em tramas de dados e de controlo, o bit de *parity* é reservado para uso futuro e será igualado a 0 por implementações desta norma, e ignorado por qualquer receptor.

Valor	Nome	Descrição
0	RINGLET_0	Transmitida no ringlet0
1	RINGLET_1	Transmitida no ringlet1

Tabela A.7 – Valores de *ri*

Valor	Nome	Descrição
00_2	FT_IDLE	Trama de Idle
01_2	FT_CONTROL	Trama de Controlo
10_2	FT_FAIRNESS	Trama de Fairness
11_2	FT_DATA	Trama de Dados

Tabela A.8 – Valores de *ft*

Valor	Nome	Descrição
00_2	CLASS_C	Classe C
01_2	CLASS_B	Classe B
10_2	CLASS_A1	Classe A, subclasse A1
11_2	CLASS_A0	Classe A, subclasse A0

Tabela A.9 – Valores de *sc*

sc	fe	Nome	Descrição
00_2	0	-	reservado
	1	CLASS_C	Classe C oportunistica
01_2	0	CLASS_B_CIR	Classe B – commited information rate
	1	CLASS_B_EIR	Classe B – excess information rate
10_2	0	CLASS_A1	Classe A reclamável
	1	-	reservado
11_2	0	CLASS_A0	Classe A reservada
	1	-	reservado

Tabela A.10 – Codificações das classes de serviço e *fairness eligible*

Valores dos campos		Interpretação		Descrição
sc	fe	Classe de Serviço	Fairness Eligible	
00_2	-	CLASS_C	Sim	Classe C oportunistica
01_2	0	CLASS_B_CIR	Não	Classe B – Commited Information Rate
	1	CLASS_B_EIR	Sim	Classe B – Excess Information Rate
10_2	-	CLASS_A1	Não	Classe A reclamável
11_2	-	CLASS_A0	Não	Classe A reservada

Tabela A.11 – Interpretações das classes de serviço e *fairness eligible*

O campo *we* é ignorado quando recebido por estações *steering*. O seu valor só tem significado quando recebido por estações *wrapping*. O valor atribuído para transmissão desde estações *steering* é igual a zero.

Enquanto uma trama que não é *wrappable* poderá ter sido *steerable* antes de ser transmitida, assim que tenha sido transmitida já não é mais *steerable*. Portanto, este campo não é para ser interpretado como tendo qualquer suporte em que à trama tenha sido providenciada protecção ou não.

A.6 – Campo extendedControl

O campo *extendedControl* (*extended ring control*) de 8 bits consiste em múltiplos sub campos (que não têm interdependência entre eles).

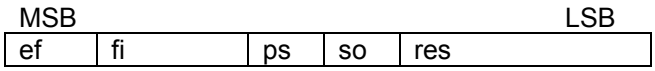


Figura A.6 – Formato do campo *extendedControl* para tramas de dados e de controlo

ef: Bit (*extended frame*) que indica se a trama é uma trama estendida. O MAC atribui o valor de 0 a *ef* na transmissão de tramas usando o formato básico da trama de dados, e atribui o valor de 1 a *ef* na transmissão de tramas usando o formato estendido da trama de dados;

fi: Campo de 2 bits (*flooding indication*) que indica se a trama é *flooded*, e se for, de que maneira. Os valores do campo *fi* estão definidos na tabela A.12;

ps: Bit (*passed source*) que é usado por sistemas *wrapping* (juntamente com outros campos) para prevenir o desordenamento e a duplicação ao prevenir a trama de ser *wrapped* mais do que duas vezes pela rede *wrapping*. É igualado a 0 quando a trama é inicialmente transmitida por uma estação e igualado a 1 quando a trama *wrapped* (i.e., uma trama a circular pelo *ringlet* secundário) passa a estação de origem;

so: Bit (*strict order*) que indica se a trama necessita de requisitos estritos de ordenamento. O valor de 1 indica que todos os MACs através dos quais a trama transita ou através do qual é recebida, devem providenciar ordenamento estrito para essa trama. O valor de 0 indica que os MACs não são requeridos de providenciar ordenamento estrito para essa trama.

Valor	Nome	Descrição
00_2	FI_NONE	Sem flood
01_2	FI_UNIDIR	flood unidireccional
10_2	FI_BIDIR	flood bidireccional
11_2	FI_RES	Tipo reservado de flood

Tabela A.12 – Valores de *fi*

A.7 - Formatos das tramas dos algoritmos de *Fairness*, Topologia e Protecção

O *payload* do formato da trama de *fairness* contém os valores de *fairnessHeader* e de *fairRate*.

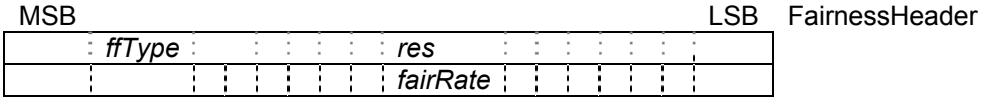


Figura A.7 — *Payload* da trama de *fairness*

ttl	MAX_STATIONS
sc	CLASS_A0
fe	FALSE
we	1
ri	(ver trama SCFF) (ver trama MCFF)

Tabela A.13 — Valores do *baseRingControl* da trama de *fairness*

O campo *ttl* é ajustado ao valor MAX_STATIONS por uma estação de origem. Cada estação subsequente num domínio de congestionamento ajusta o *ttl* decrementando-o de uma unidade

relativamente ao *ttl* da última SCFF recebida. Isto permite que uma estação de recepção calcule o número de *hops* para a estação de origem como sendo (MAX_STATIONS - *frame.ttl).*

Uma estação é a origem de uma trama de *fairness* se colocar *myMacAddress* no campo de frame.saCompact da trama transmitida.

ffType: Campo de 3-bit que identifica os tipos de tramas de f*airness;*

res: Campo de 13-bit (*reserved*) que é ignorado na recepção e ajustado a zero na transmissão;

fairRate: Campo de 16-bit que transporta uma taxa normalizada codificada como uma quantidade de 16-bit. Um valor de FULL_RATE ($FFFF_{16}$) indica a taxa máxima da linha.

000_2	SINGLE_CHOKE	Divulga o *fairRate* de uma estação para a estação vizinha a montante, uma vez em cada *advertisementInterval*
001_2	MULTI_CHOKE	Relata o *normLocalFairRate* de uma estação, a todas as outras estações no *ringlet*, uma vez em cada *reportingInterval*
010_2 - 111_2	reservado	-

Tabela A.14 — Valores do *ffType* (*fairness frame type*)

As tramas FDD (*fairness differential delay*) são identificadas pelo campo de 2-bit *ft*=FT_CONTROL e um campo de 8-bit *controlType*=CT_FDD.

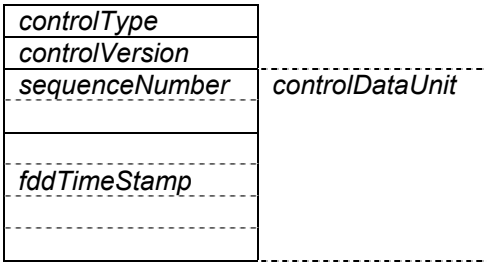


Figura A.8 — *Payload* FDD

Nome do campo	classeA FDD	classeC FDD
sc	CLASS_A0	CLASS_C
fe	FALSE	TRUE
we	1	
ttl	*NumberOfHops*(*frame.da*) ≤ *ttl* ≤ MAX_STATIONS	

Tabela A.15 — Valores do *baseRingControl* da trama FDD

O valor de *hopsToCongestion* é um valor inicial preferido para o campo do *ttl*.

sequenceNumber: Campo de 16-bit que identifica um par de tramas FDD, uma de ClasseA e outra de ClasseC, usado para calcular o FDD na estação de destino;

fddTimeStamp: Campo de 32-bit que contém um *timestamp* local gerado pelo remetente na altura de transmitir esta trama;

As tramas FDD são emitidas em pares, com uma trama no par que tem um valor de *sc* igual a CLASS_A0 e o valor de *fe* igual a FALSE, e o outro par tem um valor do campo *sc* igual a CLASS_C e um valor de *fe* igual a TRUE. Uma trama FDD é inválida se for recebida mas não for recebida a sua trama FDD associada com a outra classe de serviço.

A trama TP (*Topology & Protection*) de comprimento fixo é identificada pelo campo de *controlType*=CT_TOPO_PROT que é feito o *broadcast* para minimizar os atrasos da transmissão. Os índices da trama TP incluem a informação para sinalizar o estado de protecção da ligação, para a descoberta da topologia física do anel e para relatar a informação das preferências da estação.

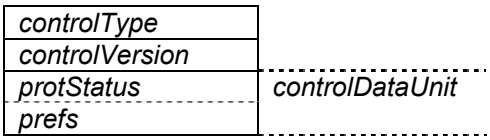


Figura A.9 — *Payload* TP

Campo	Sub-campo	Valor	Descrição
ttl	-	MAX_STATIONS	Permite que as tramas TP se propaguem através de todas as estações
baseControl	*sc*	CLASS_A0	Enviada como sub-classeA0
	fe	FALSE	Não elegível para *fairness*
	we	0	Estas tramas nunca são *wrapped*
da	-	FF-FF-FF-FF-FF-FF	Permite que as tramas TP sejam recebidas por qualquer estação

Tabela A.16 — Restrições aos valores dos campos do cabeçalho da trama TP

protStatus: Os sub-campos deste campo de 8 bits (*protection status*) são ilustrados na figura A.10.

esw: Bit (*edge state, west*) que indica se uma *edge* está presente na extensão oeste de uma estação.

ese: Bit (*edge state, east*) que indica se uma *edge* está presente na extensão este de uma estação.

psw: Campo de 3 bits (*protection state, west*) que indica o estado de protecção na extensão oeste da estação.

pse: Campo de 3 bits (*protection state, east*) que indica o estado de protecção na extensão este da estação.

Prefs: Os sub-campos do campo de 8-bit *prefs* (preferences) são ilustrados na figura A.11.

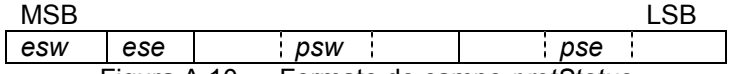


Figura A.10 — Formato do campo *protStatus*

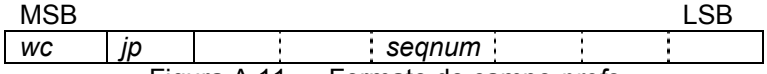


Figura A.11 — Formato do campo *prefs*

Valor	Nome	Descrição
000_2	IDLE	Não há pedido
001_2	WTR	Wait-to-Restore
010_2	MS	Manual Switch
011_2	SD	Signal Degrade
100_2	SF	Signal Fail
101_2	FS	Forced Switch
110_2	-	reservado
111_2	-	reservado

Tabela A.17 — Valores dos campos *psw* e *pse*

wc: Bit (*wrap protection configured*) que é ajustado baseado na configuração de protecção do operador e armazenado em *myTopoInfo.protConfig*.

jp: Bit (jumbo frame preferred) que é ajustado baseado na preferência *jumbo* do operador e é armazenado em *myTopoInfo.jumboPrefer*.

seqnum: Campo de 6 bits (*sequence number*) que é incrementado cada vez que outras porções do *controlDataUnit* tenham alterado.

As tramas TC (*Topology Checksum*), de comprimento fixo, são identificadas pelo campo *controlType*=CT_TOPO_CHKSUM. Estas tramas comunicam o *checksum* da topologia às estações vizinhas de modo a que as estações possam determinar se as parcelas relevantes das suas bases de dados da topologia coincidem, e consequentemente, se se podem retirar do conteúdo de contexto.

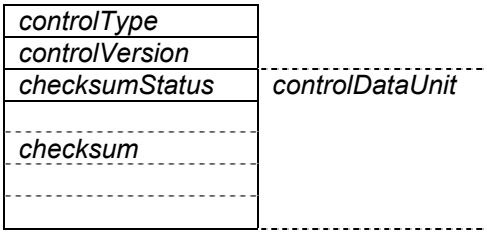


Figura A.12 — Formato da trama TC

checksumStatus: Campo de 8bits.

checksum: Campo de 32 bits. O valor deste campo é igual ao checksum calculado de um subconjunto de campos da base de dados da topologia.

Campo	Sub-campo	Valor	Descrição
ttl	-	1	Permite que as tramas TC não se propaguem para além da estação vizinha
baseControl	*sc*	CLASS_A0	Enviada como sub-classeA0
	fe	FALSE	Não elegível para *fairness*
	we	0	Estas tramas nunca são *wrapped*
da	-	FF-FF-FF-FF-FF-FF	Permite que as tramas TC sejam recebidas pela estação vizinha

Tabela A.18 — Restrições aos valores dos campos do cabeçalho da trama TC

checksumStatus:

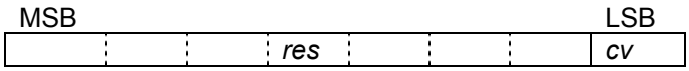


Figura A.13 — Formato do campo *checksumStatus*

res: Campo reservado de 7 bits.

cv: Bit (*checksum valid*) que indica que o valor no campo *checksum é válido*. Valor TRUE se o campo de *checksum* é válido.

As tramas de LRTT (*Loop Round Trip Time*) com comprimento fixo são identificadas pelo campo *controlType*=CT_LRTT_REQ. As estações que usam o ajuste de taxa conservador gerarão tramas de LRTT. Todas as estações executarão o processamento de tramas de pedido de LRTT.

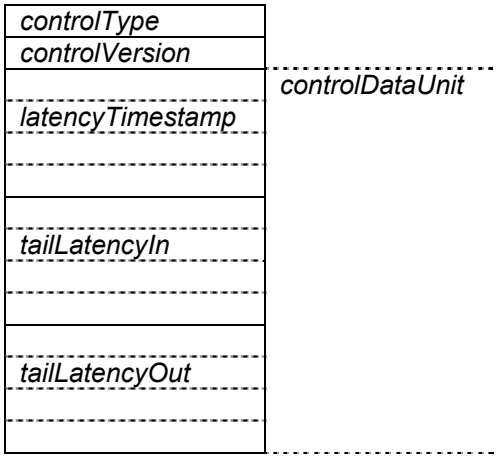


Figura A.14 — *Payload* da trama de pedido de LRTT

Campo	Sub-campo	Valor	Descrição
baseControl	*sc*	CLASS_A0	Enviada como sub-classeA0
	fe	FALSE	Não elegível para *fairness*
	we	0	Estas tramas nunca são *wrapped*
da	-	(relativo)	Endereço de *unicast* da estação na qual está a ser medido o LRTT

Tabela A.19 — Restrições aos valores dos campos do cabeçalho da trama de pedido de LRTT

latencyTimestamp: Campo de 32 bits que contém um valor temporal local gerado pelo requerente na altura da transmissão desta trama.

tailLatencyIn: Campo de 32 bits fornecido para quando a trama é reflectida como uma trama de resposta. O requerente ajustará este campo a zero.

tailLatencyOut: Campo de 32 bits fornecido para quando a trama é reflectida como uma trama de resposta. O requerente ajustará este campo a zero.

As tramas de LRTT (*Loop Round Trip Time*) com comprimento fixo são identificadas pelo campo *controlType*=CT_LRTT_RSP. Todas as estações executarão a geração da trama de resposta LRTT.

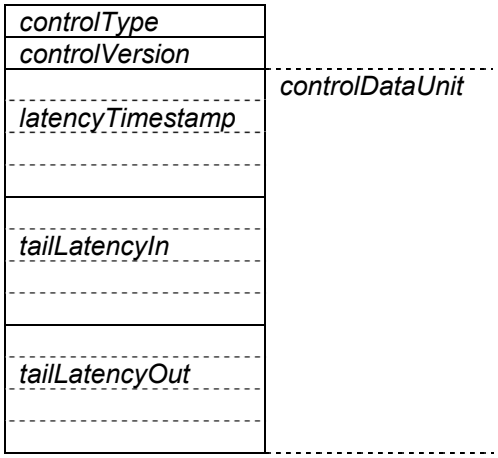


Figura A.15 — *Payload* da trama de resposta à LRTT

Campo	Sub-campo	Valor	Descrição
baseControl	*sc*	CLASS_A0	Enviada como sub-classeA0
	fe	FALSE	Não elegível para *fairness*
	we	0	Estas tramas nunca são *wrapped*
Da	-	Endereço *sa* da estação de pedido	Endereço de *unicast* da estação que mede o LRTT

Tabela A.20 — Restrições aos valores dos campos do cabeçalho da trama de pedido de LRTT

latencyTimestamp: Cópia do mesmo campo de 32 bits da última trama LRTT recebida.

tailLatencyIn: Campo de 32 bits fornecido para permitir que uma estação de resposta marque o tempo em que recebeu a trama de pedido de LRTT para a qual esta trama é uma resposta, com a finalidade de melhorar a estimativa da quantidade de tempo decorrido ao processar a trama na estação de resposta.

tailLatencyOut: Campo de 32-bit fornecido para permitir que uma estação de resposta marque o tempo em que transmitiu a trama de resposta LRTT, com a finalidade de melhorar a estimativa da quantidade de tempo decorrido ao processar a trama na estação de resposta.

As tramas de ATD (*Attribute Discovery*) com comprimento fixo são identificadas pelo campo *controlType*=CT_STATION_ATD.

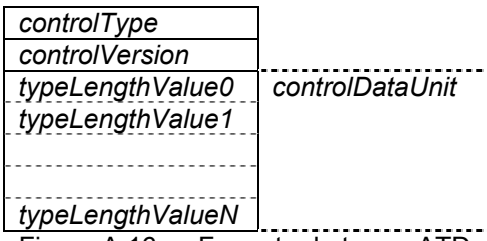


Figura A.16 — Formato da trama ATD

Campo	Sub-campo	Valor	Descrição
ttl	-	MAX_STATIONS	Permite que as tramas ATD se propaguem através de todas as estações
baseControl	*sc*	CLASS_A0	Enviada como sub-classeA0
	fe	FALSE	Não elegível para *fairness*
	we	0	Estas tramas nunca são *wrapped*
da	-	FF-FF-FF-FF-FF-FF	Permite que as tramas ATD sejam recebidas por qualquer estação

Tabela A.21 — Restrições aos valores dos campos do cabeçalho da trama ATD

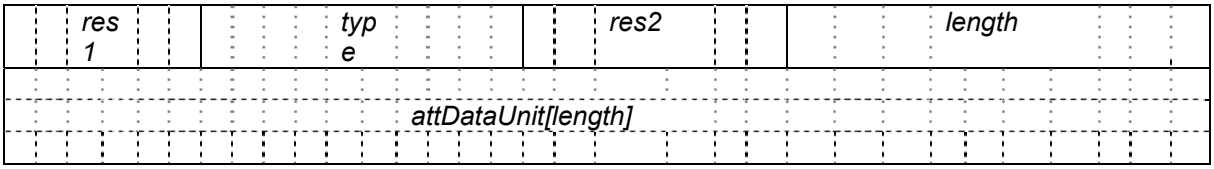


Figura A.17 — Formato do *type-length-value*

res1 e res2: Campos de 6 bits reservados.

type: Campo de 10 bits que identifica o tipo de ATT.

length: Campo de 10 bits com o comprimento, em bytes, do valor da informação *typeDependent* seguinte. O valor do comprimento será menor do que 1024.

attDataUnit: Dados cujo formato e função são especificados pelo campo *type* field precedente. O comprimento deste campo é menor do que 1024 bytes.

Valor	Nome	Tamanho	Descrição
0	-	-	Reservado
1	ATT_WEIGTH	2	Pesos *fairness* das estações
2	ATT_BANDWIDTH	4	Largura de banda reservada de classeA0
3	ATT_STATION_SET	1	Ajustes da estação
4	ATT_STATION_NAME	0-127	Nome da estação (ASCII)
5	ATT_MANAGE_ADDRESS	5 ou 17	Endereço IP de gestão
6	ATT_INTERFACE_INDEX	4	*ifIndex* da interface do anel
7	ATT_SECONDARY_MACS	12	Endereços secundários do MAC
8-1022	-	-	Reservado
1023	ATT_ORG_SPECIFIC	8-1023	Valores não normalizados (específicos de organizações)

Tabela A.22 — Codificações do campo *type* para a trama ATD

Anexo B - Base de dados da topologia

A base de dados da topologia contém informação ao nível do anel, informação da própria estação e informação das tramas TP, TC, ATD e LRTT recebidas de cada ringlet.

As diferentes parcelas da base de dados da topologia são referenciadas como se segue:

a) Informação ao nível do anel: *ringInfo.fieldname;*

b) Informação da própria estação: *myTopoInfo.fieldname;*

c) Informação das tramas TP, TC, ATD e LRTT por estação,

topoEntry[0][hops].fieldname, informação recebida desde o *ringlet0*

topoEntry[1][hops].fieldname, informação recebida desde o *ringlet1*

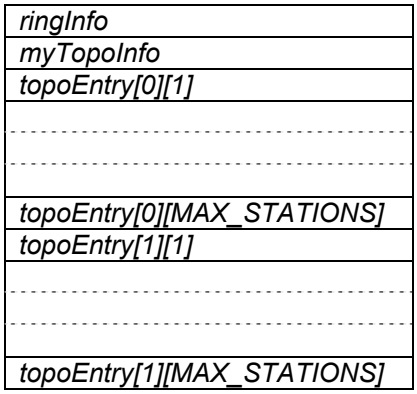


Figura B.1 – Estrutura da base de dados de topologia

A informação usada para configurar os valores na base de dados de topologia é originada de várias fontes:

a) as tramas TP são usadas para actualizar:
   - os endereços de MAC por estação no anel;
   - informação de contagem de extensões no anel;
   - estado das fronteiras (*edge*);
   - estados de protecção;
   - capacidades da estação;

b) as tramas TC actualizam o *checksum* da topologia;

c) as tramas LRTT actualizam o tempo de uma volta fechada em torno do anel;

d) as tramas ATD actualizam outros elementos na base de dados de topologia.

Descrição	Variável
Informação derivada das tramas TP	
Anel tipo *jumbo*	*jumboType*
Tamanho MTU	*mtuSize*
Numero de estações no anel	*numStations*
Total de *hops* de recepção no *ringlet0*	*totalHopsRx[0]*
Total de *hops* de transmissão no *ringlet1*	*totalHopsRx[1]*
Tipo de topologia do anel	*topoType*
Total de *hops* de transmissão no *ringlet0*	*totalHopsTx[0]*
Total de *hops* de transmissão no *ringlet1*	*totalHopsTx[1]*
Informação derivada das tramas ATD	
Estação que recebe tramas com um *fcs* errado	*badFcsUser*
Estação que usa um *fairness multi-choke*	*multichokeUser*
Largura de banda disponível no *ringlet0*	*unreservedRate[0]*
Largura de banda disponível no *ringlet1*	*unreservedRate[1]*

Tabela B.1 – Base de dados da topologia do nível do anel: *ringInfo*

Descrição	Variável
Informação derivada das tramas TP	
Endereço MAC da estação local	*macAddress*
Preferência *jumbo* da estação	*jumboPrefer*
Configuração de protecção da estação	*protConfig*
Estado de protecção da extensão oeste	*spanProtState*[0]
Estado de protecção da extensão este	*spanProtState*[1]
Estado do *edge* da extensão oeste	*spanEdge*[0]
Estado do *edge* da extensão este	*spanEdge*[1]
Numero de sequência	*sequenceNumber*
Ultimo endereço MAC conhecido da estação vizinha a oeste	*lastNeighborMac*[0]
Ultimo endereço MAC conhecido da estação vizinha a este	*lastNeighborMac*[1]
Checksum da topologia	*myChecksum.value*
Checksum da topologia válido	*myChecksum.valid*
Informação derivada das tramas TC	
Checksum da topologia da estação vizinha a oeste	*neighborCheck*[0].value
Checksum válido da topologia da estação vizinha a oeste	*neighborCheck*[0].valid
Checksum da topologia da estação vizinha a este	*neighborCheck*[1].value
Checksum válido da topologia da estação vizinha a este	*neighborCheck*[1].valid
Informação derivada das tramas ATD	
Nome da estação	*stationName*
Primeiro endereço MAC secundário	*secMac[0].address*
Estado do primeiro endereço MAC secundário	*secMac[0].state*
Segundo endereço MAC secundário	*secMac[1].address*
Estado do segundo endereço MAC secundário	*secMac[1].state*
Peso no *ringlet0*	*weight[0]*
Peso no *ringlet1*	*weight[1]*
Largura de banda reservada no *ringlet0*	*reservedRate[0]*
Largura de banda reservada no *ringlet1*	*reservedRate[1]*
Recepção de tramas com o fcs errado	*badFcsUser*
Tramas *fairness multi-choke* aguardadas	*multichokeUser*
Uso do modo conservador	*conservativeMode*
Tipo de endereço de gestão	*managementAddressType*
Endereço IP de gestão	*managementIpAddr*
Índice da interface	*interfaceIndex*

Tabela B.2 – Base de dados da topologia para a estação local: *myTopoInfo*

Variável
Informação derivada das tramas TP
macAddress
Valid
reachable
jumboPrefer
protConfig
spanProtState[0]
spanProtState[1]
spanEdge[0]
spanEdge[1]
sequenceNumber
Informação derivada de ATT
stationName
secMac[0].address
secMac[0].state
secMac[1].address
secMac[1].state
weight[0]
weight[1]
reservedRate[0]
reservedRate[1]
badFcsUser
multichokeUser
conservativeMode
managementAddressType
managementIpAddr
interfaceIndex
Informação derivada das tramas LRTT
Lrtt

Tabela B.3 – Base de dados da topologia para a um anel: *topoEntry[ringlet][hops]*

A base de dados da topologia é modificada na recepção de tramas TP, TC, LRTT ou ATD que resulta numa mudança na informação contida na base de dados, pela máquina de estados de protecção, ou baseada nas entradas oriundas desta máquina de estados que resulta em posteriores actualizações à base de dados. As mudanças nos parâmetros das tramas TC, LRTT ou ATD, ou pela máquina de estados de protecção, modificam os campos dentro da base de dados, mas não a posição aparente das estações na topologia.

As seguintes regras aplicam-se a actualizações da base de dados da topologia devido à recepção de ATTs diferentes de ATTs específicos de uma organização:

a) A informação na trama ATD é retida até que seja substituída pela informação de uma nova trama ATD, ou seja apagada pela ausência de um ATT dentro da trama ATD;

b) A informação da trama ATD é apagada sempre que a entrada associada seja nula, ou que o endereço MAC de uma entrada seja sobreposto por um novo valor;

c) Quando uma trama ATD é recebida, a informação contida nos campos ATT dentro desta trama, é usada para actualizar a base de dados de topologia;

d) Toda a informação ATT recebida de uma estação, é incluída na entrada da base de dados de topologia dessa estação.