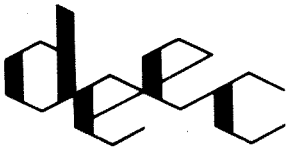


DEPARTAMENTO DE

ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES

FACULDADE DE ENGENHARIA

UNIVERSIDADE DO PORTO

Rua dos Bragas, 4099 Porto Codex – PORTUGAL

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Departamento de Engenharia Electrotécnica e de Computadores

Especificação e Desenvolvimento de um Protocolo de Comando e Controlo para Aplicações Multimédia Interactivas

Carlos Alberto Rodrigues e Silva

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Departamento de Engenharia Electrotécnica e de Computadores

Especificação e Desenvolvimento de um Protocolo de Comando e Controlo para Aplicações Multimédia Interactivas

Carlos Alberto Rodrigues e Silva

Licenciado em Engenharia Electrotécnica pela Faculdade de Engenharia da Universidade do Porto

Dissertação submetida para satisfação parcial dos

requisitos do grau de mestre

em

Engenharia Electrotécnica e de Computadores

(Área de especialização de Telecomunicações)

Dissertação realizada sob a supervisão de

Professor Doutor José António Ruela Simões Fernandes,
do Departamento de Engenharia Electrotécnica e de Computadores
da Faculdade de Engenharia da Universidade do Porto

Porto, Outubro de 1997

Resumo

Apesar das transformações profundas e aceleradas a que se assiste no domínio das telecomunicações, e de por esse motivo ser impossível prever com segurança o que nos reserva o futuro, um aspecto inquestionável é o da procura de redes cada vez mais rápidas e capazes de suportar serviços diversificados com complexidade crescente.

Com redes rápidas pode asssitir-se ao aparecimento de sistemas de natureza distribuída mais evoluídos no suporte de aplicações com níveis de exigência cada vez maiores.

O serviço de vídeo a pedido é um bom exemplo de como várias áreas de investigação se conjugam: compressão MPEG-2 de informação de áudio e vídeo, redes rápidas capazes de satisfazer exigentes requisitos de desempenho e de oferecer diferentes garantias de qualidade de serviço, e protocolos que asseguram o comando e o controlo indissociáveis da transferência de fluxos de informação de áudio e vídeo.

O projecto ATLANTIC do programa ACTS tem como objectivo integrar e demonstrar os blocos básicos de uma cadeia de distribuição de programas de televisão totalmente assente em sinais de áudio e de vídeo comprimidos, segundo as normas MPEG-2, e contribuir para a normalização das tecnologias associadas.

O foco deste trabalho situa-se ao nível da rede que suporta o estúdio ATLANTIC. São identificados os protocolos de comunicação adequados aos requisitos exigidos pelo estúdio, o TCP/IP e o ATM, sendo dedicada uma ênfase especial aos protocolos de comando e controlo, indispensáveis num ambiente como o previsto pelo ATLANTIC.

A especificação do protocolo de comando e controlo para o estúdio ATLANTIC, descrita neste trabalho, baseia-se no DSM-CC. O protocolo DSM-CC U-N assegura às camadas superiores o estabelecimento e controlo das ligações, neste caso ligações TCP/IP sobre ATM, e a gestão dos recurso atribuídos. Dada a natureza particular do problema abordado foram introduzidas algumas simplificações no protocolo DSM-CC U-N, que resultou na especificação de um protocolo U-N orientado concretamente aos requisitos do estúdio ATLANTIC. O protocolo especificado é implementado de acordo com o modelo arquitectónico adoptado para o estúdio; os blocos básicos constituintes são identificados, as interacções inerentes são convenientemente tratadas, abrindo caminho para o trabalho a efectuar na concepção do *software*.

Abstract

The world of telecommunications is nowadays characterized by deep and fast changes, that will certainly continue in the future. Nevertheless, the need for high speed communication networks able to support different services with increasing complexity is something one can forecast.

Such networks allow more sophisticated distributed systems able to support new applications with more demanding requirements.

The video on demand service shows how different research areas interact together in order to accomplish a service delivery: audio and video MPEG-2 compression techniques; high speed communication networks able to provide a variety of quality of service guarantees, and to satisfy highly demanding performance requirements; and command and control protocols associated with the transport of audio and video streams.

ATLANTIC is part of the ACTS programme. It aims at developing techniques and components which will provide the functionality for TV programme production and distribution based on a consistent MPEG-2 format throughout the broadcast chain. ATLANTIC will contribute to the definition and demonstration of standard architectures and interfaces for MPEG-2 compressed signals.

The main focus of this work is the ATLANTIC studio network. The needed communication protocols are identified - TCP/IP and ATM - according to the studio requirements. Moreover, the command and control protocols, vital to the ATLANTIC studio, are studied.

The command and control protocol specification, herein described, is based on DSM-CC. The DSM-CC U-N protocol provides to the upper layers the establishment of connections (in case TCP/IP over ATM connections) and the control and management of resources. Considering ATLANTIC's goals and requirements several simplifications on the DSM-CC U-N protocol were possible and carried out. The achieved U-N protocol specification was specifically built to the ATLANTIC studio. The U-N protocol implementation follows the models described in this work; the software building blocks are identified and the interactions between them are analyzed, thus easing the software development.

Résumé

Les changements dans le domaine des télécommunications sont difficile à prevoir et seul le futur nous apportera une solution. Une certitude cependant est que la demande pour des réseaux à grand débit, capable de supporter des services de plus en plus diversifiés et de plus grande complexité est des aujourd'hui une des plus importante question necessitant une reponse precise.

La mise au point de réseaux a grande capacité permet le développement de systèmes distribués et plus évolués qui seront le support des applications du futur.

Le service de vidéo sur demande est un bon example de la necessité d'associer différent domaines de recherche de manière a fournir un service: compression de l'information audio et vidéo selon la norme MPEG-2; réseaux rapides capables d'assurer les différents débits et qualités de service nécessaires; et protocoles pour contrôler la transference d'information audio et vidéo.

Le but du projet ATLANTIC, part du programme ACTS, est de développer des techniques et composantes qui peuvent fournit la functionalité nécessaire pour la production et distribution de programmes de télévision selon la norme MPEG-2, tout au long de la chaine de distribution. L'objectif est aussi de definir et démontrer différentes configurations et interfaces pour des signaux compressés via MPEG-2.

Dans ce projet, l'attention est concentrée au niveau du réseau qui supporte le studio ATLANTIC. Les protocoles de communication sont identifiés: TCP/IP et ATM. Les protocoles de commande et contrôle, vital pour le studio ATLANTIC sont également etudiés.

En particulier, le protocol DSM-CC de commande et contrôle est décrit dans ce document. Le protocole DSM-CC U-N fournit aux entités des niveaux supérieurs l'allocation, le contrôle et la gestion des resources en utilisant des liaisons TCP/IP sur ATM. Certaines simplifications et modifications ont été apportés au protocole original DSM-CC U-N de maniere à respecter la nature et besoins specifiques du studio ATLANTIC. Cela a conduit à la création d'un nouveau protocole U-N qui est conforme à l'architecture du modele adopté pour le studio ATLANTIC. Les blocs de base et leur interactions sont analysés et decrits, laissant la porte ouverte au développement du programme informatique de contrôle du studio ATLANTIC.

À minha mãe, Benilde,

à minha namorada, Maria João,

e aos meus irmãos, o Ricardo e o Marco.

Agradecimentos

Gostava em primeiro lugar de agradecer ao Professor José Ruela todo o trabalho e esforço a que o obriguei. Pela orientação sábia, pelos conselhos sempre prontos e valorosos, pela atenção e dedicação nunca negadas, o meu mais sentido agradecimento.

O meu obrigado a todas as pessoas que comigo trabalharam. Em particular, agradeço ao Isidro, o esclarecimento das dúvidas que vieram ao meu encontro, ao Nelson, ao Paulo e ao Illa, pela ajuda prestada. Tenho igualmente de agradecer ao Manuel Ricardo o forte incentivo que me deu. Apesar de afinal esta tese não incluir o que me sugeriu, todo o trabalho feito em colaboração com ele proporcionou-me uma experiência valiosa.

Agradeço ao Pedro, à Karine e ao Boris o auxílio.

Estou também grato à minha família e à minha namorada por toda a confiança e suporte que nunca hesitaram em me conceder.

Índice

Lista de Tabelas

Lista de Figuras

Lista de acrónimos

AAL	*ATM Adaptation Layer*
AALI	*ATM Adaptation Layer Interface*
ABR	*Available Bit Rate*
ACTS	*Advanced Communications Technologies and Services*
AMI	*ATM Management Interface*
API	*Application Programming Interface*
AREQUIPA	*Application REQUested IP over ATM*
ARP	*Address Resolution Protocol*
ATLANTIC	*Advanced Television at Low bit rates And Networked Transmission over Integrated Communications systems*
ATM	*Asynchronous Transfer Mode*
ATMARP	*ATM Address Resolution Protocol*
AUU	*ATM-layer-user-to-ATM-layer-user*
BBC	*British Broadasting Company*
BIOP	*Broadcast Interoperability Protocol*
BOA	*Basic Object Adapter*
BSD	*Berkeley Software Distribution*
BT	*Burst Tolerance*
CAC	*Connection Admission Control*
CBR	*Constant Bit Rate*
CDR	*Common Data Representation*
CDTV	*Cell Delay Transfer Variation*
CDV	*Cell Delay Variation*
CER	*Cell Error Rate*
CLIP	*Classical IP over ATM*
CLP	*Cell Loss Priority*
CLR	*Cell Loss Ratio*
CMR	*Cell Misinsertion Rate*
CORBA	*Common Object Request Broker Architecture*

CPCS-PDU	*Common Part Convergence Sublayer*
CPCS-PDU	*CPCS-Protocol Data Unit*
CPCS-UU	*CPCS User to User*
CPI	*Common Part Indicator*
CPR	*Constant Packet Rate*
CRC	*Cyclic Redundancy Check*
CS	*Convergence Sublayer*
CSELT	*Centro Studi e Laboratori Telecommunicazzioni*
CTD	*Cell Transfer Delay*
DAVIC	*Digital Audio-Visual Council*
DCT	*Discrete Cosine Transform*
DII	*Dynamic Invocation Language*
DSM-CC	*Digital Storage Media-Command and Control*
DSM-CC U-N	*DSM-CC User to Network*
DSM-CC U-U	*DSM-CC User to User*
DSS2	*Digital Subscriber Signalling System* #2
EDL	*Edit List*
EFCI	*Explicit Forward Congestion Indicator*
ENST	*École Nationale Superieure des Telecommunications*
EPD	*Early Packet Discard*
EPFL	*École Polytechnique Federale de Lausanne*
ESI	*End System Identifier*
ETSI	*European Telecommunications Standards Institute*
FhG	*Fraunhofer Gesellschaft*
FTP	*File Transfer Protocol*
FTTC	*Fiber To The Curb*
GFC	*Generic Flow Control*
GIOP	*General Inter-ORB Protocol*
GIT IE	*Generic Identifier Transport Information Element*

GPL	*General Public License*
HFC	*Hybrid Fiber Coax*
HIPPI	*High-Performance Parallel Interface*
HL	*Hierarchical Level*
IDL	*Interface Description Language*
IETF	*Internet Engineering Task Force*
IIOP	*Internet Inter-ORB Protocol*
ILMI	*Integrated Local Management Interface*
INESC	Instituto de Engenharia e Sistemas de Computadores
INET	*InterNET*
IOR	*Interoperable Object Reference*
IP	*Internet Protocol*
ISO/IEC	*International Organization for Standards / International Electrotechnical Comission*
ITU-T	*International Telecommunications Union - Telecommunications*
ITU-TSS	ITU-T *Standard Sector*
IWU	*Interworking Unit*
JPEG	*Joint Photographic Expert Group*
JWS	*Journalist Workstation*
LAN	*Local Area Network*
LANE	*LAN Emulation*
LIS	*Logical IP Subnetwork*
LLC/SNAP	*Logical Link Control / Sub-Network Attachment Point*
LRC	Laboratoire de Réseaux de Communication
MAC	*Medium Access Control*
MBS	*Maximum Burst Size*
MCR	*Minimum Cell Rate*
MIB	*Management Information Base*

MPEG	*Motion Picture Experts Group*
MPEG-2 PES	*MPEG-2 Packetised Elementary Stream*
MPEG-2 TS	*MPEG-2 Packetised Transport Stream*
MPTS	*Multiple Programme Transport Stream*
MTU	*Maximum Transmission Unit*
NHRP	*Next Hop Resolution Protocol*
NNI	*Network-to-Network Interface*
NPC	*Network Parameter Control*
NPT	*Normal Play Time*
NRM	*Network Resource Management*
nrt-VBR	*non real time* - VBR
NSAP	*Network Service Access Point*
NSC	*National Studio Centre*
OMA	*Object Management Architecture*
OMG	*Object Management Group*
ORB	*Object Request Broker*
OUI	*Organizationally Unique Identifier*
PC	*Personal Computer*
PCI	*Peripheral Component Interconnect*
PCR	*Peak Cell Rate*
PDU	*Protocol Data Unit*
PIC	*Physical Interface*
PID	*Protocol Identifier*
PPD	*Partial Packet Discard*
PPF	*Post Production Facility*
PTI	*Payload Type Identifier*
PVC	*Permanent Virtual Circuit*
QoS	*Quality of Service*
RAM	*Random Access Memory*

RDIS	Rede Digital Integrada de Serviços
RFC	*Request For Comments*
ROC	*Regional Output Centre*
RPC	*Remote Procedure Call*
RSVP	*ReSerVation Protocol*
RTP	Rádio Televisão Portuguesa
RTT	*Round Trip Time*
rt-VBR	*real time* - VBR
SAA API	*Service Aspects and Applications API*
SAAL	*Signalling AAL*
SAR	*Segmentation And Reassembly*
SCR	*Sustainable Cell Rate*
SDB CC	*Switched Digital Broadcast Channel Change*
SDU	*Segmentation Data Unit*
SECBR	*Severely Errored Cell Block Ratio*
SG	*Study Group*
SII	*Service Interoperability Interface*
SNMP	*Simple Network Management Protocol*
SPTS	*Single Programme Transport Stream*
SPVC	*Smart Permanent Virtual Circuit*
SRM	*Session and Resource Manager*
STB	*Set Top Box*
STP/UTP	*Shielded Twisted Pair / Unshielded Twisted Pair*
SVC	*Switched Virtual Circuit*
TCP	*Transmission Control Protocol*
UBR	*Unspecified Bit Rate*
UDP	*User Datagram Protocol*
U-N	*User to Network*
UNI	*User-to-Network Interface*
UNO	*Universal Networked Objects*

UPC	*Usage Parameter Control*
U-U	*User to User*
VBR	*Variable Bit Rate*
VC	*Virtual Channel*
VCC	*Virtual Channel Connection*
VCI	*Virtual Channel Identifier*
VLSI	*Very Large Scale Integration*
VPI	*Virtual Path Identifier*

1 Introdução

Assiste-se actualmente a um desenvolvimento permanente e generalizado em vários domínios das telecomunicações. A ubiquidade da Internet é talvez o mais palpável indício da emergência de novos serviços, cuja complexidade acrescida obriga a progressos tecnológicos a um ritmo acelerado.

As aplicações multimédia, entre as quais se contam serviços de teleconferência, jogos à distância e vídeo a pedido, colocam pesadas imposições de desempenho e de garantia de qualidade de serviço às redes de comunicação. Mesmo nas redes locais estes requisitos têm obrigado à procura de novas soluções (ex.: "LAN switching", *Fast Ethernet*, *Gigabit Ethernet*, etc.). A resposta para algumas questões parece estar na tecnologia ATM. Contudo, a maioria das aplicações e protocolos existentes não foram concebidos para tirar partido do ATM. Torna-se necessário antecipar cenários de transição que se não resultarem na solução ideal ao menos garantam um bom compromisso a curto prazo.

Os serviços multimédia de banda larga apresentam um enorme potencial em termos económicos. Com o objectivo de vários operadores abrangeram a totalidade do mercado que se adivinha, urge a necessidade de adoptar protocolos e interfaces abertos a vários níveis. O DSM-CC é um exemplo recente, no contexto dos protocolos de comando e controlo, para aplicações multimédia interactivas.

O DAVIC (*Digital Audio Video Council*) é uma associação sem fins lucrativos. Congrega 200 organizações de mais de 20 países e tem como objectivo encorajar a implementação e desenvolvimento de aplicações e serviços digitais multimédia de acordo com as especificações internacionais de interfaces e protocolos abertos disponíveis. O DAVIC é reponsável por uma das primeiras utilizações do DSM-CC em ambientes ATM, adoptando-o como o protocolo para controlo de sessões interactivas multimédia, dos recursos atribuídos às sessões e para interacções ao nível do serviço. O serviço de vídeo a pedido especificado pelo ATM Forum [1, 2] é também elucidativo da aplicação do DSM-CC.

O presente trabalho foi elaborado no contexto do projecto ATLANTIC, que tem como objectivos estudar e desenvolver tecnologias e componentes capazes de suportar a produção e distribuição de programas de televisão, usando compressão MPEG-2. Num estúdio, os débitos são tipicamente elevados e há a necessidade de garantir a qualidade de serviço, duas razões conducentes à adopção do ATM como a tecnologia base de comunicação. A transferência de fluxos MPEG-2 num estúdio requer elevada fiabilidade; neste trabalho equaciona-se uma

solução com recurso ao protocolo TCP. A transferência de fluxos desta natureza é igualmente indissociável de mecanismos de controlo. A especificação e implementação do protocolo que visa assegurar o estabelecimento e controlo das ligações de rede exigidas por estes mecanismos é precisamente o grande objectivo do presente trabalho.

O protocolo a especificar e implementar, designado *User-to-Network*, segue de perto a especificação do DSM-CC. No entanto, as considerações de compatibilidade total são preteridas relativamente aos requisitos do estúdio ATLANTIC. O contexto de integração deste protocolo obrigou ao estudo de interfaces com os protocolos com que necessita interactuar. Assim teve de ser assegurada uma interface com o protocolo AAL5 (protocolo de transporte das mensagens *User-to-Network*). A manipulação de SVCs implicou o estudo da interface com o protocolo de sinalização ATM Q.2931. Finalmente, a necessidade de integração com o DSM-CC *User-to-User*, obrigou à consideração do ambiente de execução desta componente do DSM-CC, o CORBA, e ao estudo subsequente de estratégias de coexistência.

O objectivo último deste trabalho é portanto a concepção de um bloco de *software* que implemente o protocolo especificado, assegurando a integração com as diversas componentes presentes no estúdio ATLANTIC.

O projecto ATLANTIC é descrito no capítulo 2, abrangendo os aspectos mais directamente relacionados com os requisitos de comunicações, do comando e do controlo. A opção pelo protocolo TCP/IP sobre a tecnologia ATM, como aqui se poderá verificar, é legitimada pelas imposições de fiabilidade identificadas e por vantagens ao nível das comunicações. O transporte de fluxos MPEG-2 entre vários sistemas introduz a necessidade de comando e controlo. Para o efeito, é estudado e proposto o DSM-CC.

Oportunamente, é efectuada a descrição do protótipo da rede local ATM no INESC bem como os trabalhos em curso inseridos nos objectivos do ATLANTIC.

No capítulo 3 são descritos os protocolos de comunicação que satisfazem os requisitos de comunicação e fiabilidade do estúdio ATLANTIC identificados no capítulo 2. A análise que incide sobre a teconlogia ATM tem a preocupação de focar os aspectos relevantes para a sua aplicação no ATLANTIC. A descrição do protocolo TCP é complementada com a análise da interacção deste com o ATM. Sucede-se a abordagem ao IP sobre ATM e a diversos aspectos subjacentes com potencial interesse para a compreensão das decisões tomadas no âmbito dos protocolos de comunicação seleccionados para o estúdio ATLANTIC.

A escolha do sistema operativo Linux é justificada no capítulo 4. Invoca-se o suporte do ATM neste ambiente, objecto de sucessivos trabalhos de actualização, disponíveis através da Internet. A integração do TCP/IP no *kernel* do sistema operativo e o acesso ao respectivo código fonte são igualmente apresentados como factores determinantes. Sucede-se neste capítulo uma abordagem às ferramentas disponibilizadas e usadas no decurso da implementação deste trabalho, o que permite compreender de forma mais clara a especificação e o trabalho experimental descritos nos capítulos posteriores.

No capítulo 5 é fundamentada a opção pelo DSM-CC. Este consta de duas componentes fundamentais: a parte *User-to-User* e a parte *User-to Network*. A primeira componente é descrita sucintamente, uma vez que não é o principal tema deste trabalho, com o propósito de constituir a base para a análise posterior da integração com o protocolo *User-to-Network*, aqui especificado e implementado. A componente *User-to-Network*, que tem por objectivo estabelecer e controlar as ligações a nível de rede (no caso presente ligações ATM) associadas à transferência de dados de áudio e de vídeo, é tratada com detalhe acrescido uma vez que o trabalho de implementação se relaciona de forma muito directa com esta área protocolar.

A descrição do serviço de vídeo a pedido é efectuada devido às semelhanças apresentadas com alguns cenários identificados no estúdio ATLANTIC. Ressalva-se, no entanto, o menor nível de exigência do ATLANTIC que evidencia a validade da opção por um modelo simplificado. Apesar de seguir de muito perto a especificação do DSM-CC a prioridade do trabalho é adequar-se às particularidades do ATLANTIC.

A especificação do protocolo *User-to-Network* termina este capítulo tendo como intuito constituir uma base sólida para o trabalho de implementação.

O capítulo 6 descreve o trabalho realizado na implementação em *software* do protocolo especificado. A abordagem ao CORBA antecede o estudo das interfaces identificadas como indispensáveis à fase de integração com todos os outros módulos de *software* em desenvolvimento, no âmbito do projecto ATLANTIC. Sucede-se por fim a descrição do modelo arquitectónico a que o trabalho experimental deve obedecer.

Finalmente no capítulo 7 são apresentadas as conclusões e discutidas perspectivas de trabalho futuro, em particular o que se espera concretizar com a integração e demonstração dos componentes que constituem a rede ATLANTIC.

2 O estúdio ATLANTIC – análise de requisitos

Neste capítulo, após a descrição sucinta do projecto ATLANTIC, abrangendo os aspectos mais directamente relacionados com os problemas levantados no plano das comunicações, do comando e do controlo, são identificados os respectivos requisitos que, como se verificará, fundamentam a opção pelo protocolo TCP/IP sobre a tecnologia ATM, por razões de fiabilidade e por vantagens que adiante se discutem ao nível das comunicações.

O transporte de fluxos MPEG-2 entre vários sistemas introduz a necessidade de comando e controlo, à qual responde o DSM-CC, estudado e proposto para o ATLANTIC.

Após uma abordagem à problemática inerente ao uso da pilha protocolar em questão, descreve-se neste capítulo o *testbed* criado no INESC, e os trabalhos em curso inseridos nos objectivos do ATLANTIC.

2.1 ATLANTIC

O projecto ATLANTIC congrega um conjunto de instituições internacionais – INESC, BBC, CSELT, Snell & Wilcox, FhG, ENST e EPFL – com o objectivo de estudar e desenvolver tecnologias e componentes capazes de suportar a produção e distribuição de programas de televisão, usando compressão MPEG-2. O ATLANTIC faz parte do programa ACTS (*Advanced Communications Technologies and Services*) suportado pela União Europeia através do "Fourth Framework Programme" (1994-1998) acordado no Tratado de Maastricht. O ACTS representa o maior esforço da União Europeia para suportar investigação e desenvolvimento tecnológico no domínio das telecomunicações.

Um factor importante na contenção dos custos e na manutenção da qualidade no domínio da produção de televisão no futuro, contempla obrigatoriamente a capacidade de manipular e processar os sinais de áudio e vídeo na sua forma comprimida. O projecto ATLANTIC tem por objectivo o desenvolvimento de tecnologia que permita ao longo de toda a cadeia, desde a captação até à difusão, a permanência do sinal no formato comprimido MPEG-2.

Os objectivos principais do ATLANTIC são portanto integrar e demonstrar os blocos básicos de uma cadeia de distribuição totalmente assente em sinais áudio e vídeo comprimidos e contribuir para a normalização das tecnologias demonstradas no projecto. Este último objectivo vai ao encontro da decisão do ATLANTIC adoptar soluções *hardware* e *software* compatíveis com as normas internacionais – ITU-T, ATM Forum, IETF, etc.

Pretende-se ainda a definição de interfaces normalizadas para fluxos comprimidos, aplicáveis a vários débitos e compatíveis com as redes de computadores e de telecomunicações.

A repercurssão económica aguardada, num cenário futuro que se prevê de crescimento acelerado de canais de televisão, traduzir-se-á na redução dos custos inerentes às tecnologias correntes. O desenvolvimento de componentes VLSI de interesse estratégico constitui-se como imperativo para assegurar a edição de fluxos MPEG-2 a baixo custo, pretendida pelo ATLANTIC.

2.1.1 Cadeia MPEG-2

O grande objectivo do ATLANTIC é dominar o uso das técnicas de compressão desde a aquisição de informação até à entrega desta ao utente destinatário.

As técnicas de compressão digital, pelo desenvolvimento que actualmente apresentam, prometem dominar a breve termo o mercado de difusão de televisão. Antevê-se já que a aquisição de programas, a sua produção, arquivo e distribuição, podem usufruir da compressão digital.

Existem fortes razões na base desta orientação. Na aquisição de programas, a compressão digital proporciona métodos robustos que utilizam a largura de banda de forma eficiente. No domínio da produção, as vantagens traduzem-se em métodos eficientes para o arquivo e edição de programas. Na distribuição, as técnicas digitais permitem oferecer um maior número de canais, relativamente às actuais técnicas analógicas.

A compressão deve ser feita de forma a que a perda de informação inerente não prejudique a visualização e audição dos programas finais. Cada estágio de compressão conduz a uma perda de qualidade, agravado por um efeito de degradação em cascata que tem de ser considerado. Na sua tentativa de utilizar técnicas de compressão desde o processo de aquisição de informação até ao passo final de difusão, o ATLANTIC aborda também esta problemática.

Como actualmente não estão completamente normalizadas as arquitecturas de redes que suportam fluxos MPEG-2, outro objectivo do ATLANTIC é o de fornecer auxílio na normalização da forma de interligar equipamento conforme com a norma MPEG-2. Para isso, pretende criar um estúdio de televisão, que envolve a recepção proveniente de diversas fontes e a entrega de vários programas a múltiplos emissores.

Com o propósito de demonstrar a viabilidade dos seus objectivos, o projecto ATLANTIC pretende desenvolver e demonstrar protótipos das componentes fundamentais numa cadeia de distribuição de televisão.

2.1.1.1 Alguns conceitos de MPEG-2

Apesar de para o vídeo se poder realizar compressão segundo as normas JPEG, da "ETSI", baseadas em *wavelets*, ou baseadas na DCT sem compensação de movimento, a norma MPEG (MPEG-2 ou MPEG-1) encontra o consenso geral na área da distribuição. Há, no entanto, ainda alguma discussão quanto à norma de compressão a usar nas restantes áreas envolvidas.

A norma *MPEG-2 Systems Specification* [3] descreve a forma como fluxos de dados áudio e vídeo comprimidos podem ser multiplexados para formar um único fluxo.

No âmbito deste trabalho interessa abordar a terminologia MPEG-2 [4], uma vez que no contexto do estúdio ATLANTIC se procede à manipulação dos dados comprimidos desde a captação da informação até à fase de distribuição.

Assim, programa (*programme*) é entendido pelo MPEG como um canal de difusão (ex.: RTP 2). Um fluxo elementar (*elementary stream*) é entendido, por sua vez, como um componente básico de um programa. Um serviço de difusão contém tipicamente três fluxos elementares correspondentes a vídeo, áudio e teletexto. Cada fluxo elementar pode ser estruturado em pacotes PES (*Packetised Elementary Stream*), que consistem em duas partes, uma de dados e outra destinada ao cabeçalho.

Os pacotes TS (*Transport Stream*) consistem em pacotes de comprimento fixo, com 188 octetos, permitem o suporte de aplicações multi-programa. Na realidade, um pacote TS tem a capacidade de acomodar vários programas independentes – MPTS (*Mlti Programme Transport Stream*), ou apenas um – SPTS (*Single Programme Transport Stream*).

Todos os pacotes PES que se destinam a ser multiplexados em conjunto são convertidos em pacotes TS, constituindo estes um fluxo MPEG-2 TS.

2.1.1.2 Modelos de referência ATLANTIC e blocos funcionais

A arquitectura do sistema recorre a uma série de modelos de referência hierárquicos, com o propósito de cada modelo definir com objectividade a funcionalidade e interfaces de cada bloco funcional.

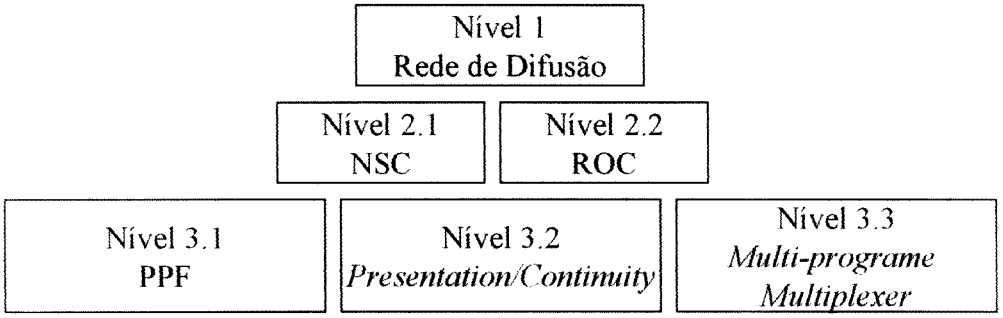


Figura 1 - Modelo hierárquico de nível 1.

Um modelo hierárquico de interesse para a compreensão do enquadramento do presente trabalho está ilustrado na figura 1. Um detalhe acrescido é alcançado à medida que se progride no modelo esquematizado.

A rede de difusão é composta, na perspectiva do ATLANTIC, por dois grandes blocos de nível hierárquico inferior. A figura 2 ilustra a forma como estes blocos se interligam.

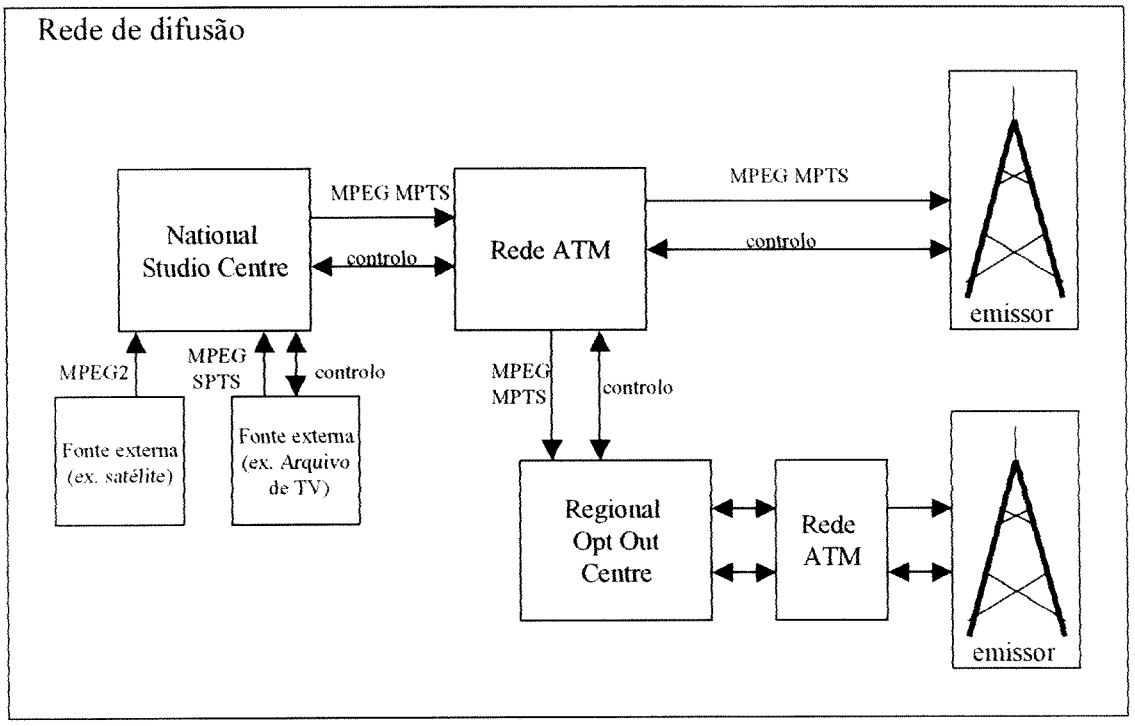


Figura 2 -Modelo de referência para a cadeia de distribuição.

A distribuição de um fluxo MPEG-2 MPTS, a partir de um estúdio nacional (NSC), é feita para uma rede terrestre de emissores. O estúdio regional (ROC) tem a possibilidade de a nível local extrair um canal MPEG-2 do fluxo MPEG-2 MPTS e substituí-lo por um outro de interesse regional.

Progredindo para o nível hierárquico 3, verifica-se que consta de três blocos funcionais, o bloco PPF (*Post Production Facility*), o bloco *Presentation/Continuity* e o bloco

Multi-programme Multiplexer. O objectivo do designado bloco de pós-produção (PPF) é o de proporcionar uma infra-estrutura capaz de suportar a produção eficiente de programas usando em simultâneo a compressão de áudio e vídeo. O modelo ATLANTIC assume dados na forma MPEG-2 à entrada da rede do bloco PPF, tendo à sua saída os programas finais no formato MPEG-2 TS. A saída do bloco PPF consiste desta forma na colecção de programas na sua versão final, que são armazenados no servidor apropriado. Os programas, uma vez editados e armazenados, podem ser visionados sob o controlo do bloco *Presentation/Continuity*. Neste são recebidos vários fluxos de áudio e de vídeo que podem ser seleccionados/misturados de forma a produzir na saída um fluxo correspondente a uma emissão de um canal televisivo ("single programme"). O *Multi-programme Multiplexer* assegura a combinação de um conjunto de fluxos SPTS num fluxo MPTS, proporcionando ainda um mecanismo que atribui a cada fluxo (*programme stream*) uma porção da largura de banda disponível.

2.1.1.3 Estúdio ATLANTIC

O foco do presente trabalho incide muito em particular sobre os aspectos da rede que se suporta o bloco funcional PPF.

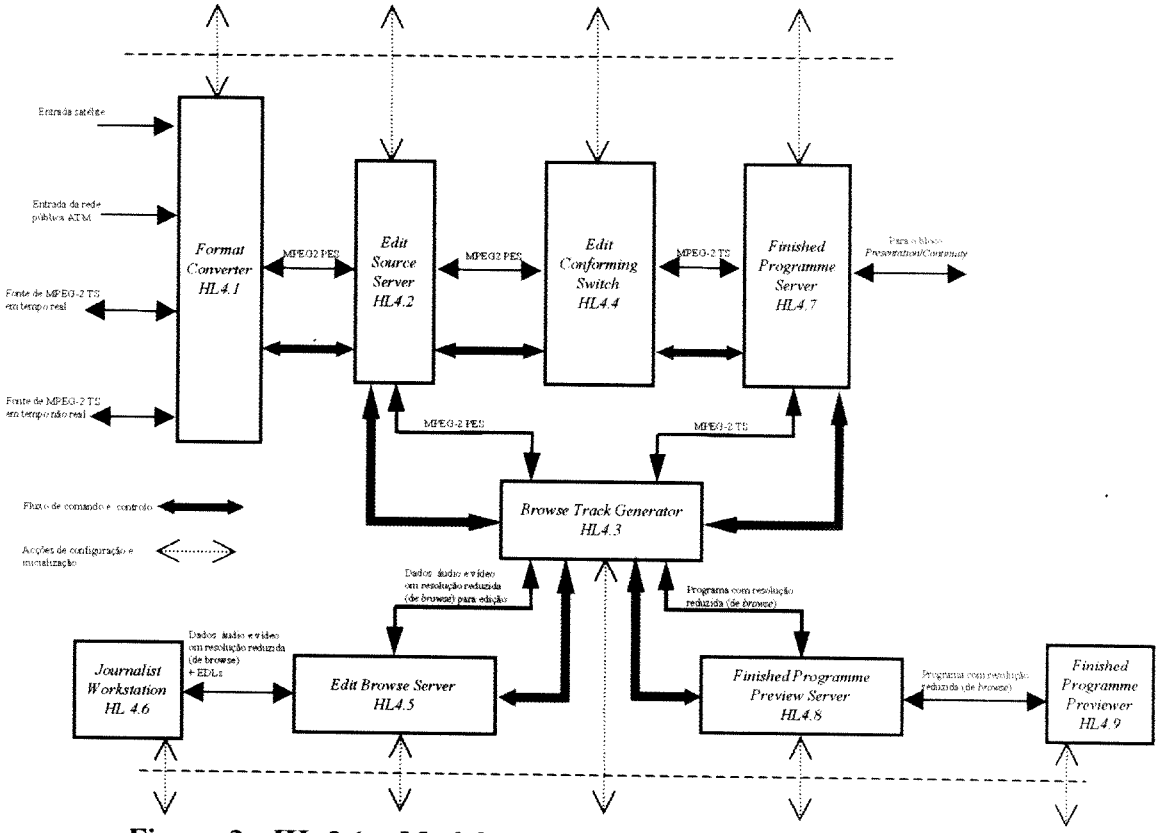


Figura 3 - HL 3.1 – Modelo do bloco de pós-produção.

Como mostra a figura 3, um conversor de formatos recebe os dados à entrada do bloco de pós-produção, convertendo-os em *streams* elementares MPEG-2 PES, a armazenar no servidor de edição (HL4.2). Em simultâneo, é gerada uma versão do mesmo material de entrada, com menor resolução, e armazenada no servidor de edição HL4.5. O formato desta versão permite edição não linear e procura, em diferido (*off-line*).

Neste cenário pretende-se o suporte de várias sessões de edição em simultâneo, a operar sobre os dados armazenados com resolução reduzida. Da edição, executada a partir da estação de jornalista (JWS), resulta uma lista de edição (EDL), usada posteriormente para controlar um comutador de edição (HL4.4) – um dispositivo que opera em tempo real sobre os dados armazenados com a resolução máxima. À sua saída o comutador de edição gera um fluxo MPEG-2 TS, o qual é finalmente guardado no servidor que armazena os programas na sua versão final (HL4.7). Mais uma vez, em paralelo, uma versão de menor resolução do programa final é armazenada no servidor HL4.8, usado para visualizar as versões finais dos programas, por meio de uma interface semelhante à disponível num comum leitor de vídeo.

É feita uma clara distinção entre dois tipos de controlo, como ilustra a figura 3. O fluxo de controlo está associado às acções de comando e controlo a que as diversas entidades lógicas presentes no bloco PPF necessitam recorrer para regular as transferências dos fluxos MPEG-2. As acções de configuração e inicialização estão por seu lado relacionadas com a acções sobre as máquinas que suportam a funcionalidade de cada entidade lógica.

2.1.2 Identificação de requisitos

À luz da descrição do estúdio ATLANTIC é agora possível proceder à identificação das exigências impostas no domínio das comunicações e do comando e controlo.

A modelização da rede destinada a interligar as entidades funcionais presentes no bloco de pós produção exige um prévio levantamento dos requisitos de comunicação.

As aplicações que no estúdio ATLANTIC recorrem aos serviços disponibilizados pela rede terão igualmente de dispor de mecanismos capazes de lhes permitir o controlo, a configuração e a gestão dos recursos da rede. A manipulação de fluxos MPEG-2 entre as diversas entidades funcionais presentes no estúdio ATLANTIC impõe deste modo requisitos de controlo muito específicos, típicos de aplicações multimédia de banda larga.

2.1.2.1 Requisitos de comunicação

A interligação dos elementos funcionais presentes no estúdio ATLANTIC impõe requisitos de conectividade, expressa em termos do número e tipo de ligações e do número de

interfaces físicas, a que a tecnologia de rede tem de corresponder. No processo de troca de dados, que pode ser efectuado de modo contínuo (ex.: fluxo de dados do *Format Converter* para o *Edit Source Server*) ou de forma interactiva (ex.: fluxo de dados solicitado pela *Journalist Workstation* ao *Edit Browse Server*), a rede tem de possuir meios para assegurar ligações ponto a ponto e ponto a multi-ponto, entre as diversas máquinas. O suporte de várias ligações entre duas quaisquer entidades terá igualmente de ser suportado.

Do ponto de vista do desempenho, os aspectos críticos a considerar no estúdio ATLANTIC são o *throughput* e a latência, aos quais são impostos apertados limites devido à presença de aplicações em tempo real. Na verdade, a troca em tempo real de informação áudio e vídeo não tolera atrasos significativos. Adicionalmente, cada fluxo de informação vídeo é exigente em termos de largura de banda.

Para além dos aspectos considerados, a natureza dos fluxos de dados no estúdio ATLANTIC exige que as ligações respeitem determinados limites relativamente a alguns parâmetros.

As garantias de desempenho e de qualidade de serviço exigidas pelo ATLANTIC podem ser proporcionadas por uma rede ATM. Pela análise dos requisitos que aqui se identificam, a tecnologia ATM é de facto a que de forma mais adequada vai ao encontro das exigências do estúdio ATLANTIC. O ATM é uma tecnologia de comunicação orientada às ligações que assenta na comutação de células de comprimento fixo. Permite o estabelecimento de ligações estáticas, por procedimentos de gestão, ou dinâmicas, por meio de sinalização, às quais oferece garantia de qualidade de serviço (largura de banda, atraso, etc.) por ligação, com débitos que podem ir até 622 Mbit/s.

As categorias de serviço suportadas pelo ATM permitem precisamente adequar as ligações a requisitos de tempo real ao permitirem a reserva de largura de banda de forma eficiente e a garantia de atrasos e débitos. Os protocolos de sinalização e a facilidade de gestão de uma rede ATM possibilitam também a conectividade entre as máquinas do estúdio, quer as ligações sejam dinâmicas ou estáticas. Apesar de por meio de ligações estáticas ser possível assegurar todo o funcionamento do estúdio, tal solução está longe de ser óptima devido à natureza dinâmica de grande parte das ligações envolvidas. Mediante este factor torna-se obrigatório o suporte de ligações dinâmicas no ATLANTIC, e como tal a sinalização ATM é imperativa.

A opção pelo ATM é ainda apoiada por um conjunto adicional de vantagens oferecido. Um grande trunfo da tecnologia ATM é a sua flexibilidade; pode acomodar tráfego com

diferentes características e com diferentes requisitos de qualidade de serviço. A escalabilidade é também um factor a favor da sua aplicação no ATLANTIC. Se uma ligação entre dois comutadores for utilizada intensivamente a ponto de a partir de determinado momento não poder satisfazer as solicitações de que é alvo, para aumentar a largura de banda do *backbone* basta apenas aumentar o número de ligações físicas, aumentar a capacidade da ligação física ou aumentar o número de comutadores, sem que tal implique a reconfiguração das máquinas. Em termos económicos, o custo de uma rede ATM prova também ser competitivo. A grande popularidade de que disfruta actualmente induz uma pressão acrescida na evolução de componentes e especificações, despoletada pela competição, que tem como resultado inevitável custos mais acessíveis.

Na comparação com outras tecnologias de rede disponíveis que permitam satisfazer as exigências de débitos elevados, o ATM apresenta na globalidade o conjunto de características que melhor se adequa aos requisitos do ATLANTIC.

A tecnologia HIPPI é orientada às ligações, comutada ponto a ponto, com um canal físico especificado para 800 Mbit/s e 25 metros. Usa pacotes de tamanho variável até 65535 octetos e usa um esquema de controlo de fluxo baseado em créditos. Os pacotes são encaminhados pela fonte, e quando um pacote é transferido obtém toda a largura de banda do canal durante a transmissão. Apesar de com o HIPPI se dispor de uma largura de banda superior relativamente ao ATM, não se garante o atraso, não sendo possível garantir da parte das ligações o respeito por compromissos, que como se concluiu é um requisito fundamental do ATLANTIC. A limitação em termos de distâncias e custos elevados associados à tecnologia HIPPI constituem desvantagens adicionais sérias relativamente ao ATM.

Uma rede FDDI opera sobre fibra óptica ou UTP, apresenta uma topologia em anel, oferecendo débitos até 100 Mbit/s. As ligações podem ter comprimentos até 100 km e o número de máquinas que se podem interligar vai até 500. Como o meio de transmissão é partilhado, a latência da rede limita o máximo *throughput*. Apesar de ser possível atribuir largura de banda para tráfego síncrono, a sensibilidade do tráfego áudio e vídeo a variações nos valores do atraso inviabilizam o recurso a esta tecnologia.

Outra alternativa ao ATM poderia ser a tecnologia *Ethernet* a 100 Mbit/s. Esta tecnologia surgiu na sequência da necessidade crescente de largura de banda nas redes *Ethernet* comuns. A evolução é feita de forma mais simples relativamente a outras tecnologias de elevado débito. No entanto, apesar da largura de banda elevada, não são oferecidas garantias de qualidade de serviço.

2.1.2.2 Fiabilidade

A análise anterior permitiu concluir que a tecnologia ATM satisfaz os requisitos de desempenho impostos pelo estúdio ATLANTIC. Há todavia, a imposição de ao nível das aplicações a comunicação ser fiável, uma vez que o tráfego MPEG-2 não tolera perdas. Consequentemente, ao mecanismo de transporte tem de ser confiada a tarefa de recuperar das perdas que eventualmente ocorram nas camadas protocolares inferiores, sem lesar todavia as garantias de desempenho.

O protocolo TCP proporciona um serviço fiável, assegurando nomeadamente a entrega ordenada das unidades de dados. A fiabilidade aliada à universalidade do TCP constituem-se como determinantes na adopção do TCP como protocolo de transporte na rede ATLANTIC.

A coexistência do TCP e do ATM tem, no entanto, de ser analisada, uma vez que os mecanismos de controlo de congestionamento do TCP comprometem à partida aspectos tão fundamentais como a latência da comunicação. Por exemplo, o mecanismo de controlo de congestionamento do TCP terá de ser adaptado, pois em ambientes ATM é nocivo. Como foi concebido para operar em redes *best-effort*, o mecanismo de controlo de congestionamento torna-se dispensável se o TCP operar sobre ATM, que oferece ligações com garantias de qualidade de serviço. O mecanismo de controlo de fluxo TCP pode todavia revelar-se útil, daí que seja necessário estudar os parâmetros de controlo de tráfego do TCP. As múltiplas características do TCP poderão assim ser capitalizadas desde que considerem o ambiente ATM, as aplicações a suportar e os requisitos de desempenho.

No estúdio ATLANTIC, os mecanismos de controlo de tráfego do ATM relegam as perdas para problemas de transmissão. No entanto, a ocorrência de perdas é interpretada pelo TCP como indício de congestionamento da rede. Torna-se assim necessário a modificação do algoritmo de controlo de congestionamento do TCP, nomeadamente ao nível do mecanismo de *slow start*, do limiar de congestionamento e da actualização da janela de congestionamento, descritos adiante. Justifica-se mesmo a adopção de um mecanismo de recuperação rápida (*fast recovery*), pois no ATLANTIC as perdas não são derivadas de situações de congestionamento, dada a forma como são negociados os recursos.

Para garantir um contrato de tráfego, o ATM usa um mecanismo de controlo de admissão de ligações com base na qualidade de serviço. O TCP pode ser deste modo usado apenas para controlo de erros e eventualmente controlo de fluxo devendo permitir, como se

concluiu, a recuperação rápida de situações de perdas. Todavia, o controlo de fluxo do TCP poderá por exemplo ser utilizado na suavização do tráfego VBR, especialmente quando se lhe depara a multiplexagem de vários fluxos TCP.

O ATM tem também por seu lado de considerar o facto de fornecer serviço ao TCP. Ao nível da camada ATM as perdas devem ser minimizadas pois tal obrigaria o TCP a proceder às correspondentes retransmissões, implicando significativas degradações no desempenho (a perda de uma única célula implica a retransmissão do pacote completo).

2.1.2.3 Requisitos de comando e controlo

A relação entre as diversas entidades lógicas presentes no estúdio ATLANTIC exige funcionalidades de comando e controlo. A troca de informação de áudio e de vídeo requer mecanismos que permitam, de forma adequada, a manipulação da informação no seu formato comprimido, quer com resolução máxima quer com resolução reduzida.

Se por um lado existe a necessidade de controlar os fluxos MPEG-2, por outro emerge a necessidade de gerir e controlar os recursos da rede. O DSM-CC cobre estes dois aspectos. É constituído por duas grandes componentes: a parte U-U e a parte U-N. A primeira proporciona ao utilizador a possibilidade de controlar o acesso a um serviço. Neste caso é necessária uma interface que permita ao utilizador controlar a troca de fluxos MPEG-2. A componente U-N está relacionada com o estabelecimento e gestão de ligações de rede, a pedido da parte U-U.

As flagrantes semelhanças de alguns cenários presentes no estúdio ATLANTIC com o modelo especificado para o serviço de vídeo a pedido [1, 2], descrito mais adiante, tornam clara a opção pelo DSM-CC.

É possível no estúdio ATLANTIC identificar vários cenários que se mapeiam directamente no paradigma cliente-servidor do DSM-CC [5].

A decisão de recorrer a uma rede ATM para interligar todos os elementos presentes no estúdio digital estreita o conjunto de exigências levantado ao protocolo de controlo da rede, no caso o DSM-CC. Adicionalmente, tratando-se simplesmente de um estúdio, o desafio imposto ao DSM-CC não é de modo nenhum tão complexo como por exemplo o levantado pelo serviço de vídeo a pedido, especialmente no que respeita ao número de entidades envolvidas. São portanto de esperar simplificações de vária ordem no protocolo a implementar.

O DSM-CC compreendendo as suas duas componentes, U-U e U-N, tem a capacidade de intervir como protocolo de controlo nos cenários ilustrados na figura 3.

Cada cenário cliente-servidor implica um determinado conjunto de acções, de acordo com a respectiva funcionalidade associada. Na perspectiva do DSM-CC U-N, a funcionalidade condiciona os requisitos exigidos por cada um destes cenários.

O cenário que envolve a estação de jornalista e o servidor de edição com resolução reduzida, será o ponto de partida para a especificação e implementação do protocolo U-N do ATLANTIC.

2.2 Protótipo da rede local ATM

O estudo e implementação da rede ATLANTIC, obriga à existência de um ambiente de desenvolvimento e teste, capaz de suportar os protocolos escolhidos bem como proporcionar um ambiente semelhante ao da futura rede ATLANTIC.

	Rede *Ethernet*		Rede ATM	
PC	IP	MAC	CLIP	MAC/ESI
Bunny	194.117.24.122	00:00:E8:D3:2A:27	192.168.26.122	00.20.48.08.9A:92
Skyrack	194.117.24.66	00:C0:0C:50:04:46	192.168.26.66	00.20.48.08.99.44
Samurai	194.117.24.146	00:AA:00:2B:93:BA	192.168.26.146	00.20.48.08.9A.A2
Osteolito	194.117.24.81	00:4F:4C:01:81:06	192.168.26.81	00.20.48.08.9A.43

Tabela I- Endereços das máquinas no *testbed* do INESC.

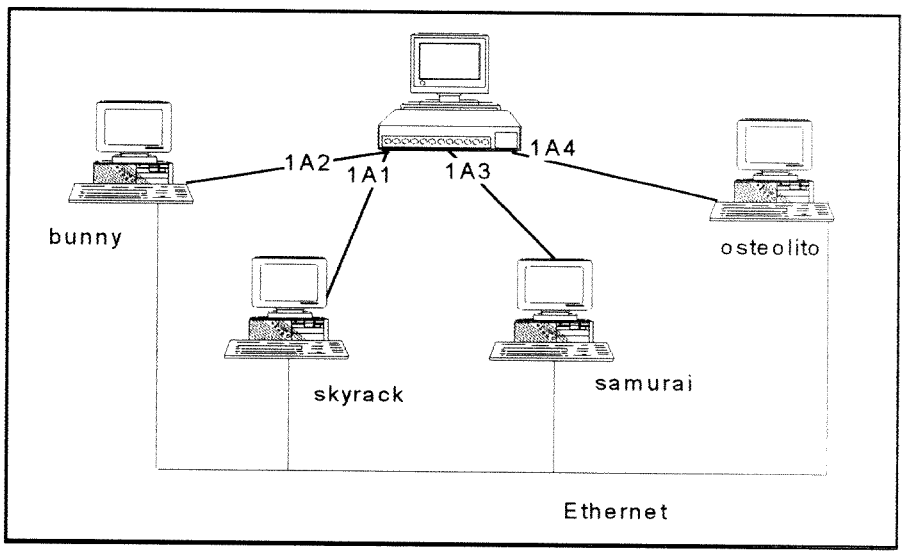


Figura 4 - Redes ATM e *Ethernet* do *testbed* do INESC.

Na fase inicial de desenvolvimento e teste a rede local ATM instalada no INESC é constituída pelos seguintes equipamentos: ASX-200WG, um comutador ATM da Fore, usado

na ligação das quatro máquinas presentes na rede; PC-Skyrack, um PC Pentium a 133 MHz, com barramento PCI, 40 Mbytes de memória RAM, equipado com uma carta PCA-200PC ATM da Fore (155 Mbit/s); PC-Samurai, PC-Bunny e PC-Osteolito, todos PC Pentium a 133 MHz, com barramento PCI, 32 Mbytes de memória RAM, equipados com uma carta PCA-200PC ATM da Fore (155 Mbit/s).

O comutador ATM *ForeRunner* ASX-200WG apresenta as seguintes características:

- arquitectura não-bloqueante a 2.5 Gbit/s;
- máximo de 12 a 24 portas (conforme a interface física);
- interfaces LAN UTP/fibra a 25/100/155/622 Mbit/s;
- conforme com a especificação UNI 3.0;
- suporte de SVCs, PVCs e SPVCs, IP sobre ATM, emulação de LANs, das categorias de serviço UBR, CBR e VBR, e de *multicast* e *broadcast*;
- suporte de gestão da largura de banda proprietária *ForeThought*
  - CAC;
  - *Leaky bucket* duplo;
  - *Smart buffers*;
  - Até 13312 células / *buffer* de porta;
  - Fila de espera por VC;
  - 3 níveis de serviço / 127 sub-níveis;
  - EPD (*Early Packet Discard*) e PPD (*Partial Packet Discard*);
  - Controlo de fluxo EFCI.

A interligação dos vários equipamentos na rede local ATM é assegurada por meio de fibra óptica multi-modo. Cada PC está também equipado com uma carta *Ethernet*, o que significa que cada computador tem dois números IP, um que o identifica na rede *Ethernet* e outro destinado a ser usado com o CLIP (*Classical IP over ATM*) na rede ATM, como ilustrado na tabela I e na figura 4.

As máquinas estão dotadas com o sistema operativo Linux, com a versão do *kernel* 2.0.25, o pacote de *software* ATM versão 0.31 e o *device driver* para ATM em Linux descrito em [6, 7]. Este destina-se a assegurar a interface da pilha protocolar do Linux com a carta PCA200E ATM da Fore (155 Mbit/s), cujo suporte não era ainda garantido pelo Linux, como se mostra na tabela III do capítulo 4. O *device driver* desenvolvido no INESC sobre *firmware* da *Fore Systems* oferece uma interface para as camadas superiores de acordo com a especificação da EPFL [8].

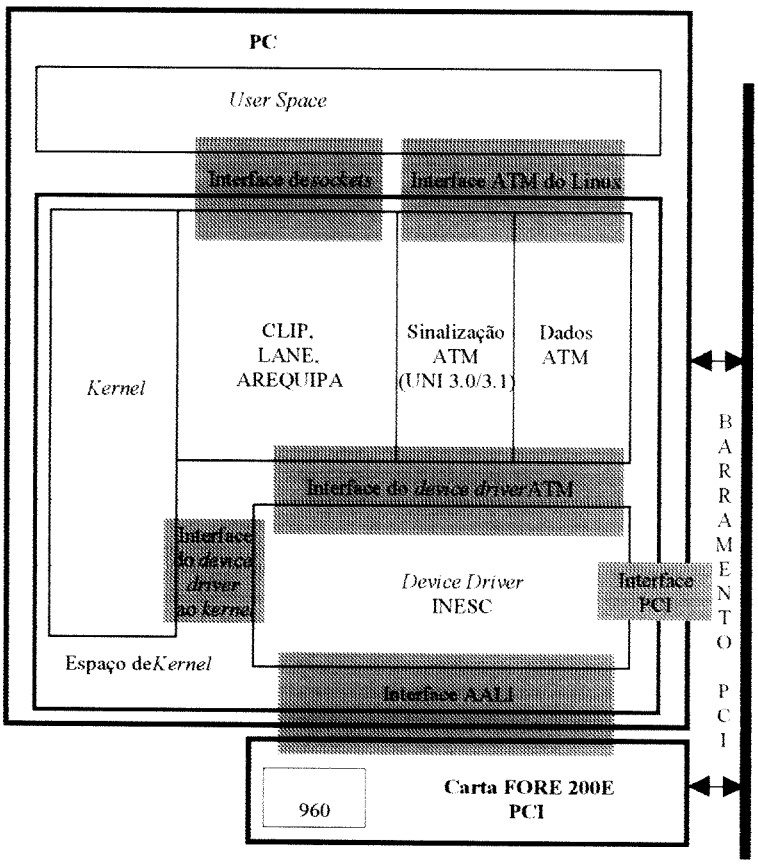


Figura 5 - Interfaces do *Device Driver* INESC.

O contexto onde se insere o *device driver* do INESC obrigou à concepção de quatro interfaces, como mostra a figura 5. A interface PCI consiste em código responsável por processar as primitivas PCI de forma a providenciar a comunicação com a carta física. A interface AALI (*ATM Adaptation Layer Interface*) da carta da Fore, define como é efectuada a comunicação com o PC, através da memória que a carta possui. O barramento PCI cria um espaço de endereços para esta memória que é posteriormente mapeado em espaço de *kernel*. Perante a necessidade de descarregar o *firmware* para a carta, acrescida do facto de o *firmware* se localizar no sistema de ficheiros, a solução encontrada consiste num programa em *user space* que, via *driver*, implementa esta funcionalidade. O método usado mapeia memória da carta em *user space* e escreve o *firmware* directamente nessa área. A interface principal do *device driver* é a interface do *device driver* ATM, pois quaisquer dados ou acções de sinalização terão de passar por esta interface.

Apesar de se encontrar em desenvolvimento, o *device driver* do INESC assegura já o funcionamento das várias ferramentas contidas no pacote de *software* ATM.

Como descrito no capítulo 4, a pilha protocolar implementada no *kernel* do Linux suporta o protocolo TCP/IP e o ATM, nomeadamente a sinalização UNI 3.0 e UNI 3.1.

Adicionalmente, providencia ferramentas de suporte do TCP sobre o ATM, que implementam o "Classical IP over ATM", a emulação de LANs (LANE) e o ATMARP.

2.3 Conclusão

A descrição do projecto ATLANTIC, com especial incidência sobre os requisitos impostos à rede destinada a assegurar a interligação dos elementos constitutivos do estúdio, conduziu à selecção do TCP sobre a tecnologia ATM para assegurar os requisitos de fiabilidade e desempenho e à adopção do DSM-CC para o comando e controlo do transporte dos fluxos MPEG-2.

Por fim, é efectuada a descrição do *testbed* no INESC, que demonstra servir os propósitos deste trabalho.

3 Protocolos de comunicação – ATM e TCP/IP

Neste capítulo são descritos os protocolos de comunicação que satisfazem os requisitos de comunicação e fiabilidade do estúdio ATLANTIC.

A abordagem ao ATM cobre os aspectos desta tecnologia com relevo para a aplicação no ATLANTIC. Após a descrição do protocolo de transporte TCP, é efectuada a análise da interacção deste com o ATM. A análise do IP sobre ATM é também realizada.

Por fim, são listadas as opções a que o ATLANTIC obriga no âmbito dos protocolos de comunicação.

3.1 ATM

O ATM é uma tecnologia de transmissão e comutação orientada às ligações e é caracterizada pelo uso de pacotes de tamanho fixo (células) de 53 octetos: 48 octetos destinados a transportar dados e os restantes 5 octetos constituindo o cabeçalho para realização de funções protocolares.

Na base da sua concepção esteve a ambição de suportar numa mesma plataforma o transporte de tráfego de voz, vídeo e dados.

O ATM tornou-se o método de transporte escolhido pelo ITU-T para suportar a RDIS de Banda Larga. Mas as potencialidades da aplicação do ATM estendem-se também às redes locais. Neste contexto, manipula-se sobretudo tráfego *best-effort*, entendido na perspectiva do ATM como tráfego sem grandes exigências de tempo real. Porém, a evolução no domínio das redes é permanente, e com o advento das aplicações multimédia na generalidade das redes, mesmo o tráfego em redes locais poderá ter grandes requisitos em termos de garantia de atrasos e débitos, como o problema tratado e detalhado mais à frente neste capítulo permite inferir.

3.1.1 Características do ATM

A concepção da tecnologia ATM assentou em três grandes directrizes: estrutura hierárquica, orientação às ligações e utilização de fibra óptica. Tal como as actuais redes telefónicas a rede ATM está estruturada hierarquicamente. O serviço fornecido é orientado às ligações: o processo de transferência de informação só será iniciado após completado todo um processo de negociação e estabelecimento da ligação correspondente. Por último, a esmagadora maioria das redes ATM recorre à fibra óptica como suporte físico, caracterizado

por elevada largura de banda disponível e por reduzidíssima probabilidade de erro no processo de transmissão.

O equipamento do utilizador é ligado à rede por meio de uma interface UNI, cujo propósito é o de não só proporcionar uma interface ao utilizador que suporte multiplexagem, mas também o de proteger o fornecedor de serviço ATM de problemas que surjam do lado de um utilizador. As ligações entre os comutadores realizam-se através interfaces NNI o que possibilita igualmente a interligação entre redes ATM independentes.

A figura 6 mostra a estrutura de uma rede ATM. Evidencia-se uma clara distinção entre os sistemas terminais (*end systems*) e a rede. Presente está também a distinção entre rede privada, que existe em empresas ou *campus* universitários, e rede pública, análoga à oferecida pelas companhias telefónicas. A ligação do utilizador à rede ATM é efectuada por meio de uma interface normalizada, a UNI, actualmente na versão 4.0 [9]. A UNI define vários suportes físicos (fibra multi-modo, UTP-5, etc.), múltiplos débitos (de alguns Mbit/s até 155 Mbit/s), códigos de linha, protocolos de configuração e protocolos de sinalização.

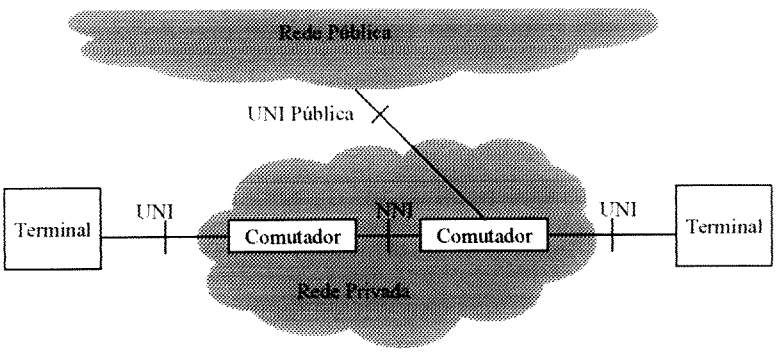


Figura 6 - Estrutura de uma rede ATM.

As ligações ATM podem ser de natureza estática, PVC (Permanent Virtual Circuit), estabelecidos mediante procedimentos de gestão, ou dinâmica, SVC (Switched Virtual Circuit), usando-se para tal procedimentos de sinalização. A sinalização ATM é baseada no protocolo DSS2 [10], para *unicast*, e no protocolo Q.2971 [11] para *multicast*. O transporte das mensagens de sinalização é assegurado pelo protocolo SAAL [12, 13, 14].

A figura 8 ilustra os procedimentos de sinalização ATM necessários ao estabelecimento de uma ligação comutada, num cenário como o ilustrado na figura 7. A entidade que inicia a ligação envia uma mensagem de SETUP para a entidade a contactar. Esta mensagem é processada por todos os comutadores até chegar ao destino. A mensagem

CALL_PROCEEDING, gerada pela rede, indica que o estabelecimento da ligação foi iniciado e não será aceite mais informação relativa ao seu estabelecimento.

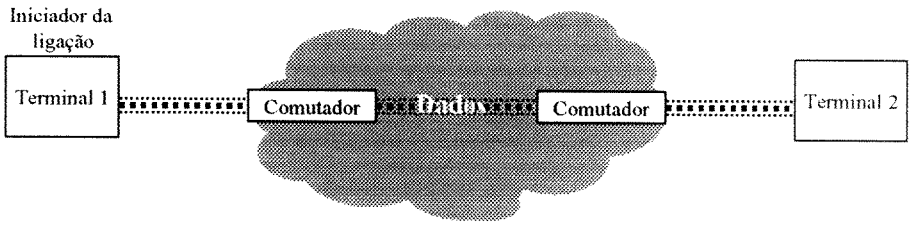


Figura 7 – Exemplo dos elementos envolvidos numa rede.

Se a entidade destinatária aceitar o estabelecimento da ligação, envia então uma mensagem de CONNECT, que transporta o VPI/VCI correspondente no elemento de informação *Connection Identifier*, cujo propósito é o de identificar os recursos da ligação ATM localmente. Esta mensagem é também interpretada por todos os comutadores por que vai passando até chegar ao terminal que iniciou a ligação. Este notifica então a entidade chamada através da mensagem CONNECT_ACKNOWLEDGE. O processo de estabelecimento da ligação termina e pode proceder-se à troca de dados.

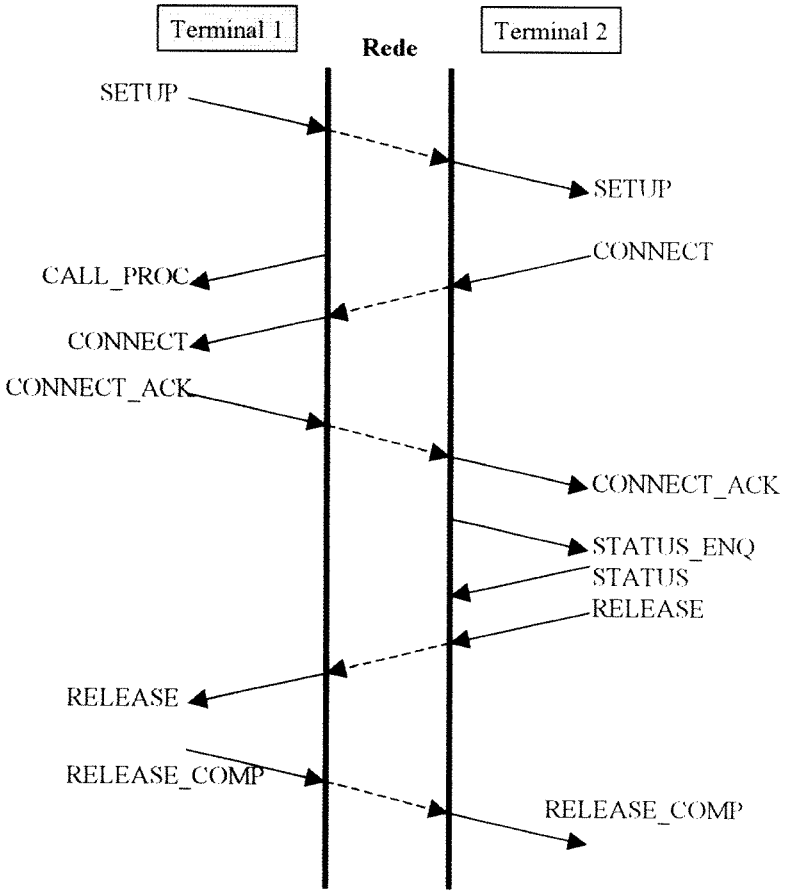


Figura 8 - Exemplo de fluxo de mensagens de sinalização.

O estado da ligação pode ser conhecido por parte de qualquer entidade envolvida, bastando para tal recorrer à mensagem STATUS_ENQUIRY, que pode ser emitida por um terminal ou pela rede em qualquer altura. Em resposta é obrigatório o envio da mensagem STATUS, descritiva do estado da ligação, por parte da entidade de nível 3 contactada.

A ligação pode ser terminada por iniciativa de qualquer um dos terminais. O procedimento consiste no envio da mensagem de RELEASE que contém um campo destinado a transportar um valor relativo à causa invocada para que a ligação seja terminada.

Numa rede que comporte tráfegos de proveniências diversas, de aplicações vídeo, de dados e de voz por exemplo, o tamanho reduzido da célula acarreta vantagens. O atraso de serialização é minimizado, ou seja, um pacote correspondente a tráfego de voz não é penalizado por tráfego de dados; numa rede de pacotes convencional ficaria obrigado a esperar que a transmissão do pacote de dados terminasse, enquanto que numa rede ATM os pacotes de dados são fraccionados em células permitindo o envio da voz antes de terminado o envio dos dados. Adicionalmente, o atraso de envio é também diminuído. De facto, a necessidade de guardar todas as células de um datagrama antes do seu envio torna-se obsoleta, podendo estas ser enviadas à medida que vão estando disponíveis no comutador.

3.1.2 Formato da célula ATM

O cabeçalho da célula ATM tem dois formatos esquematizadas na figura 9: um para as células entregues a uma interface UNI e outro para células que atravessam interfaces NNI.

Enquanto o cabeçalho da célula UNI inclui o campo GFC, o cabeçalho da célula NNI reserva o mesmo espaço para o indentificador de *virtual path*.

O campo GFC foi previsto com o propósito de possibilitar a negociação da multiplexagem dos recursos partilhados da rede, por entre as várias ligações ATM, mas não está ainda completamente definido, sendo como tal posto a zero.

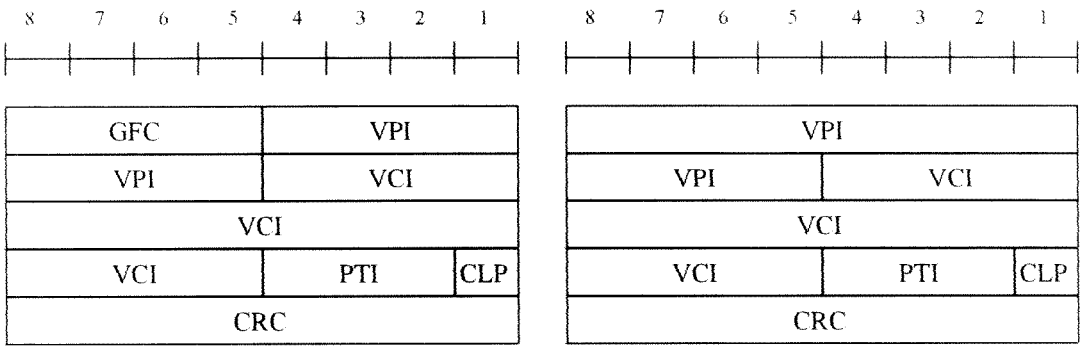


Figura 9 - Formatos dos cabeçalhos das células UNI e NNI.

Cada canal virtual ATM é identificado por um valor inteiro de 24 bits (28 bits na interface NNI). Este identificador é utilizado na multiplexagem e no encaminhamento de células e substitui os endereços de partida e destino, comuns em tantos protocolos (ex.: IP). O identificador de circuito divide-se em VPI (*Virtual Path Identifier*) e VCI (*Virtual Channel Identifier*), de 8 e 16 bits respectivamente. À passagem por um comutador, o identificador de canal nas células muda, uma vez que é feito o encaminhamento para outro comutador, por meio de um conjunto de portas diferente (o VPI/VCI tem significado local).

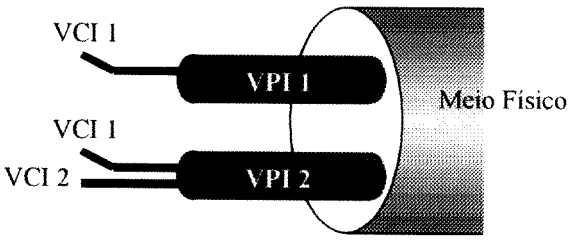


Figura 10 - Ligações ATM.

Emerge pois a necessidade de, no transporte de pacotes (ex.: datagramas IP), proceder ao seu acondicionamento nas ditas células. Há uma diferença deste procedimento relativamente à fragmentação, que ocorre por exemplo nas redes convencionais de pacotes, uma vez que o cabeçalho do datagrama não é repetido em cada célula.

O campo PTI (*Payload Type Identifier*) permite distinguir entre tráfego do utilizador e tráfego de administração e gestão (OAM).

O campo CLP (*Cell Loss Priority*) consiste num único bit, que legitima os comutadores a eliminar as células em que está activado, em caso de congestionamento ou violação do contrato negociado.

O campo CRC (*Cyclic Redundancy Check*) é calculado apenas relativamente aos cinco octetos do cabeçalho. O seu cálculo e verificação tem de ser feito em todos os sistemas que terminam a camada ATM.

3.1.3 Estabelecimento de uma ligação

No estabelecimento de uma ligação ATM o utilizador especifica a qualidade de serviço pretendida.

Cada utilizador tem necessidades específicas em relação à qualidade de serviço exigida. Esta deve ser garantida pela rede e daí a necessidade de mecanismos de controlo de tráfego especificados pelo ATM Forum em [15]. O papel destes mecanismos é o de não só prevenir o congestionamento como também o de optimizar a utilização dos recursos da rede.

Com vista a assegurar a gestão de tráfego o ATM implementa o seguinte conjunto de funções:

- *Connection Admission Control* (CAC): mecanismo que decide se uma ligação deve ser aceite ou rejeitada. A decisão respeita a qualidade de serviço das ligações presentes e verifica se estão disponíveis recursos que permitam respeitar o contrato associado à ligação em negociação.
- *Feedback controls*: acções da responsabilidade da rede e dos terminais no sentido de regular o tráfego nas ligações ATM.
- *Usage Parameter Control* (UPC), *Network Parameter Control* (NPC): monitorização e controlo do tráfego por parte da rede com vista a assegurar que os contratos negociados no estabelecimento das ligações são respeitados. Caso tal não suceda a rede pode eliminar ou marcar células (usando por exemplo o algoritmo "Leaky Bucket").
- *Cell Loss Priority Control*: os terminais podem para algumas categorias de serviço proceder ao envio de células com o campo CLP activado. A rede pode interpretar este campo no sentido de eliminar estas células, caso seja necessário, para respeitar a qualidade de serviço de tráfego prioritário.
- *Traffic Shaping*: mecanismos que visam modificar as características do tráfego, de acordo com valores de parâmetros de tráfego, espaçando adequadamente as células de um fluxo.
- *Network Resource Management* (NRM): a gestão de recursos da rede pode contemplar a separação lógica de ligações de acordo com as características do serviço. Técnicas de escalonamento de células podem, por exemplo, ser usadas nesta gestão.
- *Selective Cell Discard and Frame Discard*: para maior eficiência, uma rede congestionada pode eliminar tramas de nível superior ao invés de proceder à eliminação de células, isto é, elimina selectivamente as células de um pacote/trama.
- *ABR Flow Control*: o protocolo de controlo de fluxo ABR pode de forma adaptativa partilhar a largura de banda disponível entre vários utilizadores que usam a categoria ABR.

3.1.4 Camada de adaptação ATM AAL5

O modelo protocolar de referência ATM consiste essencialmente em três camadas:

1. camada física, que assegura a adaptação das células ao suporte de transmissão;
2. camada ATM, responsável pela multiplexagem e comutação das células; esta camada é totalmente independente da camada física;
3. camada de adaptação (AAL), que adapta o fluxo de informação à estrutura da célula e fornece um serviço adequado aos requisitos das aplicações.

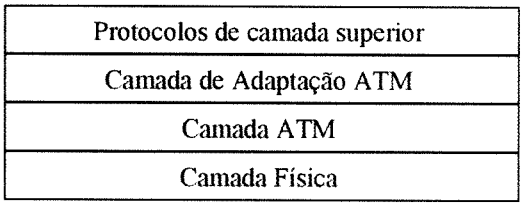


Figura 11 – Modelo protocolar de referência do ATM.

A camada de adaptação divide-se duas sub-camadas: a sub-camada de convergência (CS, *Convergence Sublayer*) e a sub-camada de segmentação e reassemblagem (SAR, *Segmentation and Reassembly Sublayer*). A primeira assegura o encapsulamento dos dados do utilizador antes de estes serem passados à sub-camada SAR, que segmenta o resultado em blocos de 48 octetos.

A camada de adaptação ATM tem portanto a seu cargo, segmentar as unidades de dados da aplicação (ex.: pacotes) em células, passar as células à rede para transmissão e, finalmente no destino, voltar a associar as células em unidades de dados.

Para responder aos requisitos identificados foram definidos até ao momento quatro protocolos de adaptação, de acordo com [16, 17].

Apesar das categorias de serviço permitirem, de forma mais adequada, negociar os parâmetros de qualidade de serviço e a atribuição de recursos relativamente a uma ligação, a classificação em classes de serviço proposta pelo ITU-T teve importância na definição dos protocolos da camada de adaptação ATM.

	Classe A	Classe B	Classe C	Classe D
Relação temporal entre a fonte e o destino	Requerida		Não requerida	
Débito	Constante	Variável		
Modo de ligação	Orientado às ligações			*Connectionless*

Figura 12 - Classes de serviço ATM.

Em [17], é especificada a camada AAL1 para serviços com débito constante; para serviços de débito variável com requisitos de tempo real foi previsto o tipo AAL2. Estes

protocolos vieram ao encontro dos requisitos levantados pelas classes de serviço A e B, respectivamente.

Os serviços de classe C e D, requeridos por vários protocolos de comunicação baseados na transmissão de pacotes ou tramas de informação, de comprimento variável e associados a tráfego *bursty*, conduziu à especificação de camadas de adaptação adequadas a este tipo de tráfego – AAL3/4 e AAL5. O AAL3/4 suporta tráfego de classes C e D, tal como o AAL5, embora se considere que este está particularmente vocacionado para serviços orientados às ligações (*connection oriented*). Para além disso o AAL5 tem também sido considerado para suporte de serviços de classe A e B (ex.: vídeo com débito constante ou variável). A vantagem do AAL5 sobre o AAL3/4 é sobretudo a simplificação introduzida. O cabeçalho é menor relegando, no entanto, a tarefa de multiplexagem para uma camada superior. A detecção de erros é também simplificada, a pré-atribuição de *buffers* de reassemblagem deixa de ser necessária e apenas um tipo de CPCS_PDU é suportado. O encapsulamento ao nível da sub-camada SAR é removido, permitindo-se deste modo transportar 48 octetos de CPCS_PDU em cada célula. O tipo de SAR_PDU é indicado pelo parâmetro AUU (*ATM-layer-user-to-ATM-layer-user*) do campo PTI do cabeçalho da célula ATM [18].

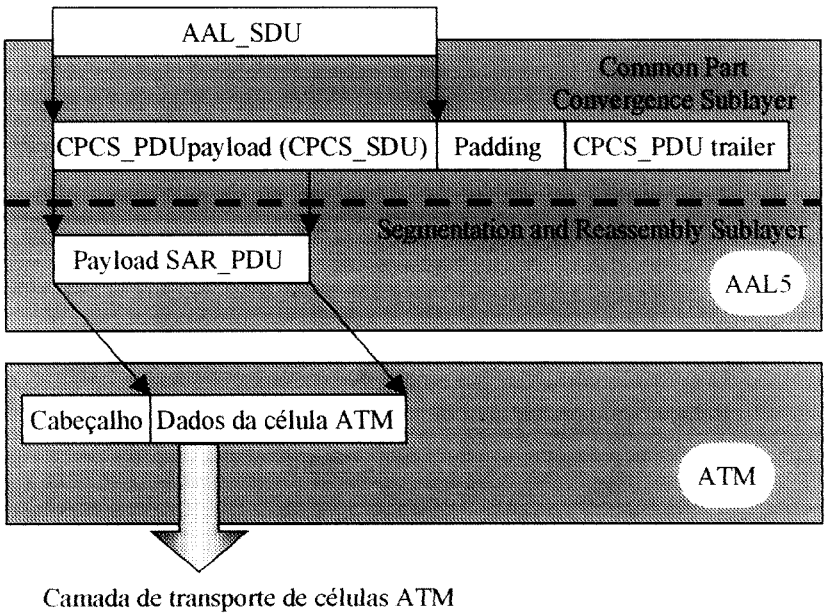
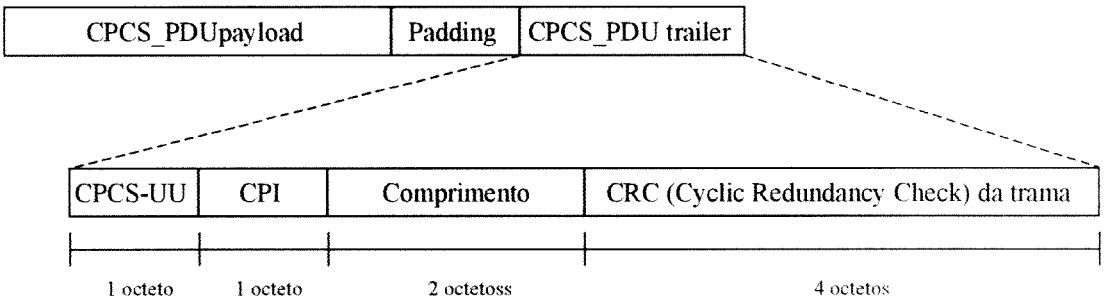


Figura 13 - Transmissão de uma AAL_SDU através da camada AAL5.

A figura 13 mostra a transmissão de uma AAL_SDU recorrendo à camada AAL5.

Apenas dois tipos de SAR_PDU são usados neste contexto. Tendo o valor zero, o parâmetro AUU indica o princípio ou a continuação de SAR_PDUs. Quando activado, indica a presença do último SAR_PDU de um CPCS_PDU.



CPCS-UU: CPCS User to User Indication
CPI: Common Part Indicator

Figura 14 - O CPCS_PDU (Common Part Convergence Sublayer Protocol Data Unit).

O CPCS_PDU é mostrado na figura 14. O campo de enchimento (*padding*), de comprimento compreendido entre 0 e 47 octetos, assegura que o *trailer* ocupa sempre os últimos oito octetos do último SAR_PDU e que o alinhamento do CPCS_PDU é feito em múltiplos de 48. A detecção de erros está a cargo do campo CRC, que estende o seu alcance a todo o CPCS_PDU. O campo contíguo indica o comprimento em octetos dos dados transportados no CPCS_PDU (CPCS_PDU *payload*). O campo CPI (*Common Part Indicator*) não tem ainda um objectivo definido tendo por ora de ser inicializado com o valor zero. O campo CPCS-UU (CPCS *User-to-User*) permite o transporte transparente de um octeto entre duas entidades CPCS (é transportado sem ser interpretado).

Como a camada AAL5 não suporta a multiplexagem numa ligação virtual simples, todos os SAR_PDUs correspondem ao CPCS_PDU actual ou ao próximo. Assim o primeiro SAR_PDU a chegar após a recepção de um SAR_PDU com o campo AUU activado é interpretado como o princípio de um novo CPCS_PDU.

Em síntese, a camada AAL5 encapsula uma unidade de dados da camada superior. Estas são de comprimento variável, entre 1 e 65535 octetos. No caso do protocolo IP, o ATM empacota deste modo um pacote IP, num "pacote" AAL5, segmenta-o posteriormente em células e activa o parâmetro AUU na última célula do "pacote" para sinalizar o fim do datagrama. Apesar da camada AAL5 aceitar datagramas com tamanho até 65535 octetos, o protocolo TCP/IP limita tipicamente este MTU a datagramas com 9180 octetos [19]. Quer isto

dizer que o protocolo IP fragmentará todos os datagramas que excedam este limite antes de os passar à camada AAL.

3.1.5 Parâmetros de qualidade de serviço e parâmetros de tráfego

São vários os parâmetros negociados entre o utilizador e a rede no momento do estabelecimento de uma chamada. A qualidade de serviço, as características de fluxo das células e as regras para a verificação da conformidade dos parâmetros de tráfego, são acordados antes da chamada ser efectivamente estabelecida.

Os parâmetros de qualidade de serviço são os seguintes:

- *Cell Loss Ratio*: CLR = células perdidas / total de células transmitidas;
- *Cell Error Ratio*: CER = células erradas / (células correctamente transferidas + células erradas);
- *Severely Errored Cell Block Ratio*: SECBR = número de blocos severamente corrompidos[1] / número total de blocos;
- *Cell Misinsertion Rate*: CMR = células incorrectamente encaminhadas / intervalo de tempo;
- *Cell Transfer Delay*, CTD: atraso sofrido por uma célula desde que é emitida até ser recebida (resulta do atraso de propagação, do atraso nos comutadores e de atrasos de codificação, descodificação, segmentação e reassemblagem);
- *Cell Delay Variation*, CDV: a diferença entre o CTD máximo e mínimo medidos durante a ligação; resulta da variabilidade da fonte e dos efeitos cumulativos de variabilidade introduzidos pela rede.

De entre esta lista de parâmetros os mais importantes são o CLR, o CTD e o CDV, que dependem das características da rede e determinam as categorias de serviço ATM.

Existe ainda um outro grupo de parâmetros que recebem a designação parâmetros de tráfego. Uma fonte especifica, no processo de negociação de uma ligação, um ou mais parâmetros conforme a categoria de serviço solicitada. A rede garante a qualidade de serviço apenas às células que os não violem.

- *Peak Cell Rate*, PCR: inverso do tempo mínimo entre células geradas pela camada ATM (e não no fluxo multiplexado); do ponto de vista prático pode ser entendido como o débito médio máximo durante um *burst*;

[1] Bloco severamente corrompido – um bloco no qual mais que M células sobre N (que constituem o bloco) estão erradas, perdidas ou incorrectamente encaminhadas.

- *Sustainable Cell Rate*, SCR: o débito médio de transmissão de células de que o utilizador pode usufruir;
- *Minimum Cell Rate*, MCR: débito mínimo garantido (no caso de tráfego ABR);
- *Maximum Burst Size*, MBS: número máximo de células que podem ser enviadas ao débito PCR, respeitando o SCR;
- *Cell Delay Variation Tolerance*, CDVT: a quantidade de *jitter* tolerado pela rede, ou seja, a tolerância para a variação do tempo de chegada entre células à rede;
- *Burst Tolerance*: BT = (MBS - 1)*(1 / SCR – 1 / PCR); a tolerância relativamente à variação do tempo entre células, por forma a respeitar-se o SCR.

Os parâmetros de tráfego são controlados e monitorizados pela rede de forma a assegurar que as características de tráfego negociadas para cada ligação são respeitadas. O algoritmo "Leaky Bucket", por exemplo, permite monitorizar as células que vão entrando no comutador e verificar se os contratos de tráfego UPC, negociados no estabelecimento das ligações correspondentes, estão a ser respeitados.

3.1.6 Categorias de serviço ATM

Os operadores ATM propuseram uma classificação de serviços em categorias com o intuito de possibilitar a utilização eficiente dos recursos da rede e garantir a qualidade de serviço que cada utilizador pede no estabelecimento de uma ligação. O relacionamento das características do tráfego e dos requisitos de qualidade de serviço com o comportamento da rede fica deste modo assegurado relegando para segundo plano a classificação do serviço em classes, proposta pelo ITU-T.

De acordo com ATM Forum [15] consideram-se 5 categorias de serviço, definidas com base em parâmetros de tráfego e de qualidade de serviço:

- CBR (*Constant Bit Rate*), a utilizar por aplicações em tempo real que têm requisitos de atraso e de perdas exigentes; uma quantidade fixa de largura de banda é atribuída, com base no PCR negociado, e reservada durante o tempo de duração da ligação, podendo ou não ser utilizada na totalidade pela fonte (ex.: serviços de voz e de vídeo, emulação de circuitos), mas não é disponibilizada para outro tráfego.
- rt-VBR (*Real-Time Variable Bit Rate*): as fontes de tráfego enviam as suas células com débitos variáveis (as fontes podem ser descritas como *bursty*), negociando o PCR, o SCR e o MBS; pode ser suportada a multiplexagem estatística devendo a rede oferecer garantias quanto ao atraso e variação do atraso.

- nrt-VBR (*Non-Real-Time Variable Bit Rate*): esta classe de serviço é idêntica à anterior negociando igualmente o PCR, o SCR e o MBS, com a excepção de não oferecer garantias quanto ao atraso máximo das células na rede.
- UBR (*Unspecified Bit Rate*): nenhuma garantia de serviço é negociada sendo portanto fornecido um serviço do tipo *best-effort* (ex.: transferência de ficheiros e serviço de correio electrónico).
- ABR (*Available Bit Rate*): esta classe de serviço garante apenas taxa de perdas pequena se a fonte adaptar o seu débito de acordo com a informação de controlo gerada pela rede – para o efeito, é especificado um mecanismo de controlo que suporta vários tipos de realimentação para o controlo do débito da fonte em resposta a alterações na rede; no estabelecimento da ligação é negociado um débito mínimo garantido, o MCR (que pode ser zero), e o PCR (largura de banda máxima requerida); a largura de banda disponibilizada pela rede pode variar não podendo, no entanto, descer abaixo de MCR.

3.1.7 Controlo de congestionamento em ATM

O controlo de congestionamento, no caso particular das redes locais, é tradicionalmente implementado pelo protocolo MAC que consiste em atribuir à máquina que ganha o acesso à rede, por exemplo por meio de um *token* ou após a resolução de colisões, toda a largura de banda disponível. Todas as outras máquinas terão de aguardar a sua vez. Nas redes ATM o protocolo MAC é removido, podendo cada dispositivo usar toda a largura de banda no acesso à rede e potencialmente provocar o seu congestionamento. Se porventura uma ligação ficar congestionada e os *buffers* do comutador virem excedida a sua capacidade, o procedimento resume-se muito simplesmente à eliminação das células em excesso. A tecnologia ATM não prevê, no entanto, a transmissão parcial de pacotes de níveis superiores e todo o pacote terá de ser retransmitido na eventualidade da perda de células.

A evolução do ATM, exigida nomeadamente pelo suporte do TCP, apelou para a implementação de mecanismos de controlo de congestionamento sendo os dois mais comuns o PPD e o EPD [20]. Isto aplica-se tipicamente a tráfego de dados suportado em AAL5, usando o mecanismo AUU descrito anteriormente.

No mecanismo PPD as células relativas a um pacote são eliminadas assim que uma seja perdida, no caso de se exceder a capacidade dos *buffers*; a eliminação das células é feita

até que se encontre o parâmetro AUU o que implica que a multiplexagem de pacotes num mesmo VC está impossibilitada.

O esquema EPD, introduzido por [20], consiste na eliminação por parte do comutador de pacotes completos antes de atingido o limite dos *buffers* respeitando-se um valor de limiar pré-estabelecido.

3.2 TCP

O protocolo TCP, graças ao advento da Internet, tornou-se no mais usado protocolo de Transporte suplantando o proposto no modelo OSI e tornando-se o protocolo *de facto*.

Este protocolo é usado como complemento do mecanismo de endereçamento proporcionado pelo protocolo IP (permitindo multiplexar/desmultiplexar fluxos TCP sobre IP), que manipula datagramas. O TCP proporciona um serviço de transporte fiável, extremo a extremo, orientado às ligações, que para além de garantir a integridade e o ordenamento dos dados assegura também o controlo de fluxo às conexões. Tal é conseguido recorrendo a uma complexa interacção entre vários algoritmos e estruturas de dados [21].

Como protocolo fiável, o TCP assenta na técnica *positive acknowledgement with retransmission*, segundo a qual o receptor notifica a fonte (através de uma mensagem de *ACKNOWLEGE*) que recebeu um ou mais pacotes em sequência. Para tal, a fonte guarda um registo de cada pacote que envia, por forma a aguardar a respectiva mensagem de notificação. Simultaneamente, é iniciado um temporizador, designado de retransmissão, por cada pacote enviado, que terá de ser retransmitido se o temporizador expirar (ex.: mensagem de notificação perdida, pacote perdido ou atraso excessivo devido a congestionamento na rede).

A quantidade de dados que o TCP pode enviar através de uma determinada ligação está limitada pela janela anunciada pelo receptor e por uma janela designada de congestionamento, controlada pelo emissor, e que varia de acordo com o estado de congestionamento da rede.

O TCP guarda o registo dos dados transferidos em octetos apesar de os enviar em pacotes. Cada pacote consiste num cabeçalho seguido de dados. O cabeçalho contém informação de controlo e de identificação das máquinas (endereço IP e número da porta TCP). Do cabeçalho consta ainda informação relativa ao número do primeiro octeto do próximo pacote que a fonte espera receber (relativamente ao tráfego em sentido oposto). A janela anunciada pelo receptor é transportada no cabeçalho das mensagens dirigidas à fonte, indicando a quantidade de dados que está disposto a aceitar.

Alguns pacotes transportam apenas informação de controlo, para pedido de estabelecimento ou terminação de ligação.

Os mecanismos iniciais de controlo de janela e de congestionamento sofreram entretanto novas evoluções, nomeadamente em resposta ao advento de aplicações multimédia e do tráfego WWW.

As comunicações baseadas em fibra óptica, de banda larga e com elevados requisitos de fiabilidade, obrigaram a extensões no TCP, como são os mecanismos de *Window Scaling* [22] e o *Fast Retransmit/Fast Recovery* [23]. Tudo isto não foi ainda, porém, ao encontro de algumas particularidades do TCP lesivas nestes novos cenários. O mecanismo de *slow start* em determinadas situações não se justifica [22], sendo desejável um mecanismo que permita recuperar rapidamente de situações resultantes da expiração de temporizadores de retransmissão ou de congestionamento da rede, para melhor aproveitamento da largura de banda disponível. Em [24] chama-se, no entanto, a atenção para o perigo de afectar a estabilidade da rede com um algoritmo mais agressivo no caso de ter havido de facto congestionamento da rede.

O facto de o protocolo TCP ter sido concebido para funcionar sobre qualquer tecnologia de suporte tem constituído um importante factor de inércia na sua adaptação a tecnologias como o ATM que, na eventualidade de perder uma única célula na transmissão, pode potencialmente desencadear o mecanismo de *slow start*.

Os mecanismos de controlo de fluxo e de erros foram concebidos para redes que fornecem um serviço *best-effort*. Assim, não há reserva de recursos para as ligações TCP o que significa potenciais situações de congestionamento e perdas consequentes de pacotes. A recuperação das perdas realiza-se por meio de retransmissão e durante a recuperação devem ser usadas técnicas de controlo de congestionamento com o objectivo de prevenir situações graves que poderiam resultar de retransmissões sucessivas (em cadeia). As redes ATM proporcionam controlo de tráfego e mecanismos de atribuição prévia de recursos com várias classes de tráfego, sub-aproveitados com o uso do TCP.

3.2.1 MSS

No protocolo TCP a informação é enviada em pacotes que, neste contexto, recebem a designação de segmentos. O valor máximo do tamanho dos segmentos – *Maximum Segment Size* (MSS) - é previamente acordado entre os pares envolvidos na conexão. Os segmentos enviados através de uma conexão não têm necessariamente o mesmo tamanho.

O dimensionamento do MSS assume-se relevante pelas repercussões que tem no desempenho do TCP. Se sub-dimensionado, a utilização efectiva da rede é reduzida (só o cabeçalho introduzido pelo TCP/IP é de 40 octetos), se por outro lado, o valor do MSS for demasiado elevado, pode conduzir a fenómenos de fragmentação de pacotes ao longo das sub-redes atravessadas e o processo de reconstituir a informação na recepção pode ser seriamente atrasado pois a simples perda de um fragmento implica a retransmissão do pacote na sua totalidade. Para além disso, em redes com velocidades de transmissão elevadas, o processo de reconstituição de informação pode tornar-se o factor limitativo do desempenho.

3.2.2 Mecanismos de controlo de fluxo

O controlo de fluxo no TCP recorre a um mecanismo de janela deslizante que permite a transmissão de pacotes enquanto se aguarda a notificação relativa a outros já transmitidos. O número de octetos transmitidos nestas condições é limitado pelo tamanho da janela anunciada pelo receptor que, dispõe assim de um mecanismo de controlo de fluxo ao anunciar a capacidade, em termos de *buffers*, disponível para a conexão TCP. A fonte também impõe controlo de fluxo através da janela de congestionamento que reflecte a percepção da fonte relativamente ao estado de congestionamento da rede.

Os dados são enviados considerando a janela de congestionamento, de que a fonte guarda o registo, e a janela anunciada pelo receptor. Para efeitos de transmissão é usado o menor destes valores.

Em certos pontos da rede, a carga pode eventualmente exceder a largura de banda disponível conduzindo a perdas por congestionamento. Os atrasos aumentam implicando a expiração dos temporizadores correspondentes nas fontes e têm lugar retransmissões. Estas, por sua vez, podem aumentar ainda mais o congestionamento – *congestion collapse*. Para prevenir este último fenómeno, quando ocorrem perdas ou atrasos, o TCP recorre às estratégias de *slow start* e *congestion avoidance*.

A estratégia de *congestion avoidance* assenta no pressuposto de que as perdas ocorrem sobretudo devido a congestionamento. Mediante a perda de um segmento, a janela de congestionamento é reduzida para metade (até um mínimo de um segmento). Para os segmentos que permanecem na janela, o temporizador de retransmissão aumenta exponencialmente.

A estratégia de *slow start* permite a recuperação de situações de congestionamento e evita que novas conexões introduzam congestionamento. Sempre que se aumenta o tráfego

após um período de congestionamento ou se inicia uma ligação, a janela de congestionamento é inicializada com o valor um e aumentada um segmento por cada notificação recebida pela fonte. A fonte começa assim por emitir apenas um segmento. Após a recepção da notificação, a janela de congestionamento é incrementada de um para dois segmentos. O envio de dois segmentos implica a recepção de duas notificações implicando, cada uma, o incremento da janela de congestionamento que ficará com o valor quatro. A abertura da janela de congestionamento é desta forma exponencial.

Para impedir que a janela aumente muito rapidamente, conduzindo eventualmente a novo congestionamento, assim que a janela atinge metade do tamanho que possuía no instante em que se verificou a situação de congestionamento (*slow start threshold size*), a velocidade de crescimento da janela é reduzida, aumentando apenas um segmento se para todos os segmentos na janela for recebida uma notificação (fase de *congestion avoidance* do TCP).

3.2.3 *Fast retransmit* e *fast recovery*

Uma vez que o TCP não consegue aperceber-se se a recepção de várias mensagens de notificação para o mesmo segmento é causada pela perda desse segmento ou apenas devido ao reordenamento dos segmentos, o algoritmo *fast retransmit* assume que um segmento se perdeu se receber três ou mais mensagens de notificação para o mesmo segmento (*duplicated ACKNOWLEDGE*). Sem esperar que o temporizador de retransmissão expire, é retransmitido o segmento completo que supostamente se perdeu.

O algoritmo *fast recovery* possibilita *throughput* elevado sob condições de congestionamento moderado, especialmente com valores de janelas elevados. Após o *fast retransmit* enviar o segmento supostamente perdido, o algoritmo de *congestion avoidance* é executado mas o de *slow start* não, ou seja, com o *fast retransmit* a janela de congestionamento é reduzida para metade do valor que possuía quando ocorreu a suposta perda e o incremento de um segmento é feito apenas quando, para todos os segmentos da janela, for recebida uma mensagem de notificação.

Sem estes mecanismos que, tipicamente, são implementados em conjunto qualquer perda de pacotes obrigaria ao escoamento dos dados no interior da rede seguido de um *slow start*. Ao invés disso, a perda de um pacote por janela pode ser recuperada, de forma eficiente, com estes algoritmos.

3.2.4 Desempenho

Um parâmetro de grande importância no TCP é o produto largura de banda*atraso na medida em que, corresponde aos dados em trânsito no interior da rede (*data in flight*). O valor do atraso corresponde ao RTT (*Round Trip Time*) da rede.

O valor deste produto condiciona o dimensionamento dos *buffers* na fonte e na recepção pois, para optimizar o *throughput* da conexão TCP, tem de se considerar a quantidade de dados sem notificação a manipular para que a rede permaneça ocupada. O dimensionamento dos *buffers* deve ser criterioso. Feito de forma inadequada pode resultar na perda de pacotes devido simplesmente a picos de tráfego (*burstiness*) bastante abaixo, eventualmente, do congestionamento da rede.

Para se atingir a utilização máxima o tamanho da janela deve pelo menos corresponder ao produto largura de banda*atraso. Este produto é elevado para redes com atrasos significativos (ex.: ligações satélite) ou com largura de banda elevada (ex.: numa rede ATM com 600 Km a 155 Mbit/s o produto atinge um valor próximo de 1 Mbit[2]). Surgiu então, o mecanismo *Window Scaling* [25], para permitir tamanhos de janela superiores ao máximo original de 65535 octetos. Todavia, o recurso a janelas muito grandes tem o efeito nefasto de aumentar da probabilidade de perda de mais que um pacote por janela. Tal implica um drástico abaixamento no *throughput* pois, enquanto o temporizador de retransmissão não expira, a ligação não é utilizada. Esta é uma das causas principais na base do mau desempenho que o TCP apresenta em redes como o ATM.

3.2.5 Análise do TCP sobre ATM

O confronto das características do TCP com as do ATM deixa antever um conjunto de obstáculos ao desempenho adequado, do TCP sobre ATM.

As implementações TCP abrangem débitos que podem ir de 100 bit/s até 10 Mbit/s, com atrasos entre 10^{-3} e 10^{2} segundos. O recurso a fibras ópticas nas redes ATM permite velocidades de transmissão mais elevadas, acima destes valores.

A camada de adaptação AAL recebe os segmentos TCP/IP, fragmenta-os em células que envia de forma ordenada através de um circuito previamente estabelecido, garantindo portanto a entrega ordenada das células no destino. O processo de reconstituição dos segmentos não levanta problemas.

[2] L.B.*RTT = $155*10^{6}$ (bit/s)*$2*5*10^{-6}$(s/km)*600(km)=930000 bits

As redes ATM são caracterizadas por uma largura de banda disponível elevada. Como tal, o produto largura de banda*atraso tem um valor elevado. Para que a utilização da rede seja optimizada a janela de congestionamento, usada pelo TCP, terá de ter pelo menos este valor conduzindo ao aumento da probabilidade de perda de mais que um pacote para uma mesma janela.

Os parâmetros de uma ligação TCP - tamanho de janela, MSS, RTT – não têm correspondência com os parâmetros de qualidade de serviço negociados para uma ligação ATM. Para além disso, o tráfego TCP varia significativamente implicando uma variação dinâmica dos parâmetros associados enquanto que os parâmetros negociados para uma ligação ATM se mantêm fixos até ao fim da ligação.

O tráfego TCP tende a variar significativamente (*burstiness*), ou seja, o TCP transmite os pacotes de uma janela tão rápido quanto possível (ex.: 65535 octetos podem ser transmitidos em menos de 10 ms). O tráfego TCP pode, no entanto, ser suavizado por mecanismos implementados na fonte (*traffic shaping*). A variabilidade do tráfego TCP aliada à granularidade insuficiente para assegurar um bom desempenho em presença de perdas (se o comutador ficar sob congestionamento, o tempo que se aguarda para a retransmissão ao nível do TCP pode ir de 200 a 500 ms), podem conduzir a situações de fraco desempenho.

O TCP foi originalmente concebido para operar sobre redes *best-effort* que comutassem pacotes. O facto do ATM comutar células constitui um grande entrave ao desempenho do TCP sobre ATM. A comparação do desempenho do TCP sobre redes ATM (sem o controlo de congestionamento efectuado pelo ATM) é comparado em [20] com o desempenho do TCP sobre redes de pacotes. A primeira conclusão é de que o desempenho pode ser drasticamente comprometido pela perda de células sob congestionamento no comutador ATM. O baixo *throughput* obtido deve-se sobretudo à largura de banda desperdiçada na transmissão de células relativas a pacotes já corrompidos (pacotes relativamente aos quais o comutador já perdeu pelo menos uma célula). Os *buffers* são também utilizados de forma ineficiente. Em [20, 26] demonstra-se que a adopção do TCP em ambientes ATM pode desperdiçar metade da largura de banda disponível. Em [27] sugere-se igualmente, com base em observações detalhadas, que o mau desempenho do TCP sobre o ATM se fica a dever ao facto de nos comutadores ATM se verificar a tendência de eliminar células pertencentes a vários pacotes. Note-se que o desempenho do TCP seria bom ainda que se perdesse um pacote completo quando a janela aumenta. O que acontece porém, é a perda

de células pertencentes a vários pacotes seguida da transmissão das células pertencentes a estes pacotes já corrompidos e condenados portanto a serem retransmitidos.

Numa situação em que várias máquinas de uma mesma rede se encontram a transmitir, havendo congestionamento, múltiplos pacotes são corrompidos. O despoletamento do mecanismo de *slow start* pode ocorrer de forma sincronizada nas várias máquinas conduzindo a situações de oscilação na rede; todas as máquinas transmitem congestionando a rede após o que todas iniciam o *slow start* ficando a rede sub-utilizada e assim sucessivamente.

O TCP poderá melhorar o seu desempenho se for dotado da capacidade de recuperar rapidamente da perda de múltiplos pacotes. Todavia, a alternativa de efectuar o controlo de fluxo (*rate control*) ao nível ATM, nomeadamente através do mecanismo EPD, apresenta-se mais promissora. O trabalho descrito em [28] demonstra que se podem obter ganhos significativos no desempenho de tráfego TCP sobre redes ATM se for usado o controlo de fluxo ao nível do ATM. Segundo as medidas efectuadas, a eficiência é quase perfeita, pois é assim prevenida a perda de células por congestionamento, e como resultado o TCP não provoca atrasos devidos a prolongados períodos de inacção enquanto se aguarda que os temporizadores de retransmissão expirem. Também as experiências conduzidas no âmbito do projecto MAGIC [26] permitem concluir que o controlo de fluxo feito estritamente ao nível da ligação ATM evita o congestionamento e a indução de perda de células, proporcionando simultaneamente um excelente *throughput* do TCP nas redes locais. São analisadas em [20] duas estratégias de controlo de congestionamento ao nível do ATM: o PPD e o EPD. A primeira técnica, embora melhore o *throughput* não conduz à sua optimização. O mecanismo EPD, por seu lado, não espera que se exceda a capacidade dos *buffers* e como tal elimina unidades completas de nível superior (tramas AAL5 no caso) mal se atinja um valor limiar. O resultado é a optimização do *throughput* evitando a fragmentação. Apesar de constituir uma violação do princípio de estruturação em camadas, este mecanismo integra-se na tendência descrita em [29], "layering may not be the most effective modularity for implementation".

A diminuição do *throughput*, quando o TCP é suportado sobre o ATM deve-se ainda ao uso de *buffers* pequenos no comutador (que demasiado grandes são susceptiveis de provocar atrasos), janelas TCP grandes, pacotes TCP grandes e ao número elevado de ligações TCP. Os últimos três factores introduzem congestionamento e agravam o efeito de perda de células de diferentes pacotes.

Em conclusão, face às razões apontadas, o *throughput* baixa necessariamente pois há a entrega de células inactivas (pertencentes a pacotes corrompidos), há a retransmissão de

pacotes já recebidos e a ligação poderá ficar parada à espera que expire o temporizador associado ao TCP.

3.3 IP sobre ATM

O protocolo de rede IP surge quase invariavelmente associado ao protocolo de transporte TCP, ambos popularizados com a implantação e expansão da Internet. O IP possibilita a interligação de várias sub-redes com diferentes tecnologias. Com o TCP/IP, o TCP vê apenas uma única rede pois o IP apresenta uma interface homogénea.

A introdução do ATM não pode seguir uma estratégia de ruptura com as tecnologias existentes. Para que as aplicações já existentes, desenvolvidas para correr sobre TCP/IP, não sofram alterações, o que é um importante objectivo enquanto não se criam aplicações que tirem partido directamente do ATM, a integração do ATM pode seguir uma de duas alternativas: a emulação de LANs (LANE) proposta pelo ATM Forum e o IP sobre ATM proposto pelo IETF. Enquanto a primeira solução suporta qualquer protocolo de rede a segunda prevê apenas tráfego IP.

No documento que descreve o IP sobre ATM [30] são propostos vários métodos para que os datagramas IP sejam mapeados em ligações ATM. Questões como o encapsulamento e a multiplexagem, o mapeamento de endereços IP em endereços ATM, a interligação de redes e o tamanho do MTU, são abordadas.

O CLIP foi definido pelo IETF [31] e descreve como os protocolos IP e ARP podem correr sobre PVCs e SVCs. O modelo introduz o conceito de LIS (*Logical IP Subnetwork*) e permite a multiplexagem de vários protocolos, quer baseada em VCs quer em encapsulamento LLC. A multiplexagem de vários protocolos sobre AAL5 é definida em [32]. A questão do tamanho do MTU é abordada e respondida em [19]. Por omissão, o MTU para uma rede IP sobre AAL5/ATM é 9180 octetos. Se forem utilizados SVCs a especificação dos campos a incluir nas mensagens de sinalização ATM encontra-se em [33].

3.3.1 LIS

Uma LIS é uma sub-rede IP lógica. Cada LIS opera e comunica de forma independente ainda que localizada na mesma rede ATM que outras sub-redes lógicas. No interior da LIS as máquinas encontram-se ligadas por ATM e comunicam de forma directa (*single hop*). Os dados são enviados directamente através de um VCC (*Virtual Channel Connection*), extremo a extremo, sem encaminhamento, mesmo que os elementos do par da

ligação estejam adstritos a comutadores ATM distintos. Se porventura a comunicação implicar máquinas de sub-redes lógicas diferentes então terá de passar por um *router* IP mesmo que fosse possível estabelecer um VC entre os dois membros IP sobre a rede ATM.

Para que os membros IP pertençam a uma LIS têm de ter o mesmo identificador de sub-rede e estarem directamente ligados por ATM. Os membros fora da LIS têm de estar acessíveis por intermédio de um *router*. Para além disso, os elementos da sub-rede têm de possuir os mecanismos de resolução de endereços ATMARP e InATMARP ou apenas InATMARP, se assentarem sobre SVCs ou sobre PVCs, respectivamente.

Duas redes LIS estão assim separadas por um *router* que termina os VCCs. Uma máquina pode pertencer a mais que uma LIS necessitando para tal de tantos endereços IP quantas as LIS a que pertence.

Se forem usados PVCs, os endereços IP são mapeados em circuitos virtuais de forma estática. Com SVCs tem de ser implementada a sinalização ATM e o que está especificado em [33].

3.3.2 Resolução de endereços

Para suportar IP sobre ATM é necessário o suporte dos protocolos ATMARP e InATMARP, extensões dos protocolos ARP e InARP.

Se a rede assentar sobre PVCs, todas as máquinas IP que suportem PVCs têm de utilizar o protocolo InATMARP. Quando uma máquina recebe resposta a um pedido InATMARP completa a tabela ATMARP. O mapeamento entre IP e ATM pode, no entanto, ser feito manualmente sem recorrer a quaisquer protocolos (através de uma tabela configurada manualmente).

Se a rede assentar sobre SVCs é necessário recorrer a um servidor ATMARP, dotado de uma base de dados, capaz de responder a qualquer pedido ATMARP de qualquer membro IP da sub-rede lógica que serve. O estabelecimento de uma ligação ATM implica portanto uma comunicação prévia com o servidor.

Numa LIS, um cliente tem de registar os seus endereços ATM e IP no servidor ATMARP tendo de possuir *a priori* o endereço ATM do servidor. Se um cliente desejar estabelecer uma ligação com qualquer outro membro da LIS basta-lhe conhecer apenas o endereço IP do destinatário para ser capaz de formular o pedido de resolução do endereço IP ao servidor ATMARP. Este retornará o endereço ATM correspondente possibilitando o estabelecimento da ligação ATM.

3.3.3 Multiplexagem

Vários protocolos podem ser multiplexados sobre AAL5 ou por encapsulamento LLC ou com base em VCs. O primeiro método necessita apenas de um VC para suporte dos vários protocolos. A contrapartida é a utilização de parte do espaço destinado a dados (o campo de dados dos CPCS-PDUs terá de transportar também informação relativa ao protocolo subjacente) e a introdução de um acréscimo no tempo de processamento. O encapsulamento com base em VCs requer mensagens de sinalização para, na fase de estabelecimento das ligações, estabelecer a correspondência entre cada protocolo suportado e os VCs.

No encapsulamento LLC/SNAP, os PDUs são encapsulados no campo de dados dos CPCS-PDUs do protocolo AAL5. No início do PDU, um cabeçalho LLC e um cabeçalho SNAP identificam o protocolo do PDU. Para encaminhar PDUs ISO, o cabeçalho LLC transporta a informação relativa ao protocolo (o SNAP não é usado). Para protocolos como o IP (não ISO), a identificação é feita com base nos cabeçalhos LLC e SNAP (este é constituído por dois campos: OUI – *Organizationally Unique Identifier* e o PID – *Protocolo Identifier*, com os valores 0x000000 e 0x0800, respectivamente, no caso do IP).

A multiplexagem baseada em VCs, também designada por encapsulamento nulo (*null encapsulation*), dispensa a adição de quaisquer informações ao campo de dados dos CPCS-PDUs do protocolo AAL5 pois por cada protocolo suportado é estabelecido um VC. Este tipo de encapsulamento adequa-se a redes que permitam a criação dinâmica de um número elevado de VCs, de forma rápida e económica, característico das redes privadas.

Duas outras formas de encapsulamento, no âmbito do IP sobre ATM, cuja filosofia assenta sobretudo na redução ou mesmo eliminação do cabeçalho IP são o TULIP (*TCP and UDP over Lightweight IP*) e o TUNIC (*TCP and UDP over Nonexisting IP Connection*). Ambos exigem que as entidades IP posssam comunicar entre si através de ligações *single hop*. Após a resolução dos nomes, da resolução dos endereços e da sinalização ATM, é estabelecida uma associação entre os pares envolvidos na ligação. Os campos transportados no cabeçalho IP, como os endereços fonte e destinatário, tornam-se desta forma redundantes.

Com o TULIP apenas o campo *PROTOCOL* (que identifica tratar-se do protocolo IP), do cabeçalho IP, é transportado em cada pacote sendo toda a restante informação trocada no estabelecimento da ligação. Tal é possível pois uma vez estabelecida a associação da ligação ao par envolvido, não existem problemas de encaminhamento, o AAL5 indica o tamanho de cada pacote, não ocorre fragmentação e qualquer eventual condição de erro é tratada pela

sinalização ATM. Para o suporte do TULIP é necessária uma extensão simples ao processo de negociação do encapsulamento, descrito em [33]. O encapsulamento TULIP ocorrerá na última fase dos procedimentos de sinalização.

Com a utilização do TUNIC não há transporte de qualquer informação da camada de rede nos pacotes transferidos, optando-se antes por trocar toda essa informação no momento do estabelecimento da ligação. A associação implícita é feita entre duas aplicações sobre TCP ou UDP directamente sobre AAL5 num VC dedicado. Este encapsulamento requer, no entanto, extensões significativas à negociação do encapsulamento, descrito em [33].

Encapsulamento	Dados na mensagem de SETUP	Dados para a desmultiplexagem
LLC/SNAP		Endereços fonte e destino Família protocolar Protocolo Portas
Baseada em VCs	Família protocolar	Endereços fonte e destino Protocolo Portas
TULIP	Endereços fonte e destino Família protocolar	Protocolo Portas
TUNIC-A	Endereços fonte e destino Família protocolar Protocolo	Portas
TUNIC-B	Endereços fonte e destino Família protocolar Protocolo Portas	

Tabela II - Encapsulamento

A tabela II sintetiza as opções disponíveis para o encapsulamento de protocolos sobre AAL5, bem como os dados que cada um exige no estabelecimento da ligação e os que servem de base à desmultiplexagem. Com base nestas alternativas, no CLIP, a multiplexagem pode ser feita a vários níveis:

1. nível rede – podem existir tantos VCCs quantos os protocolos suportados. Na recepção os dados são desmultiplexados por: endereços da fonte e do destino, protocolo (TCP ou UDP) e porta. A associação é feita no estabelecimento da ligação.
2. nível rede extremo a extremo (TULIP)– podem existir tantos VCCs quantos os protocolos suportados. Na recepção os dados recebidos no VCC de tráfego IP são desmultiplexados por: protocolo (TCP ou UDP) e porta. A associação é feita no estabelecimento da ligação.

3. nível de transporte (TUNIC-A) – implica uma ligação extremo a extremo. Os pacotes não transportam informação de rede. Podem existir tantos VCCs quantos os protocolos a suportar entre duas máquinas de uma LAN ATM. Na recepção, os dados no VCC de tráfego IP são desmultiplexados por porta. A associação entre VCC e protocolo de transporte é feito no estabelecimento da ligação.
4. nível da aplicação (TUNIC-B) – pode existir um VCC por cada fluxo de transporte entre duas aplicações. A associação entre VCCs e os pares de portas é feita no estabelecimento de cada ligação.

O último método é o mais eficiente pois o cabeçalho introduzido na comunicação é menor e o tempo de processamento nas máquinas é reduzido. Para além disso, a separação de tráfego por VCCs, com requisitos de qualidade de serviço eventualmente diferentes, garante que no caso de haver congestionamento num VCC os outros não são afectados. Apesar de tudo, a primeira solução é a que se encontra mais bem definida enquanto as restantes estão em evolução. Para além disso se forem usados SVCs, em [33] tem de ser suportado o encapsulamento LLC.

3.4 Arquitectura de comunicação do ATLANTIC

A análise feita relativamente aos requisitos impostos pelo estúdio ATLANTIC e o estudo das características de cada protocolo no sentido de fundamentar as opções relativas a cada questão foram ambos conjugados de forma a aproveitar sinergias.

Ao nível físico a interligação dos elementos constituintes do estúdio ATLANTIC realiza-se com recurso a uma rede ATM local. Esta consta de um comutador cujas portas se ligam numa topologia em estrela. Topologias mais complexas (ex.: "two-tier" ou em malha) poderão ser consideradas sem se alterarem os pressupostos relativamente às opções arquitectónicas. Uma interface física a 155 Mbit/s, sobre fibra multi-modo (embora se possa recorrer a uma solução alternativa usando STP/UTP categoria 5), cobre as necessidades do ATLANTIC uma vez que os fluxos MPEG-2 previstos não excedem este débito.

Pelo facto de haver necessidade de estabelecer ligações ATM com o exterior (ex.: recepção de fluxos MPEG de uma rede ATM terrestre) sem que os aspectos de segurança e a necessidade de terminar protocolos internos e externos sejam descurados, a interligação com o exterior pode ser efectuada através de uma unidade designada "Channel Adapter", incorporada no *Format Converter*.

A categoria de serviço ATM deve ser tal que evite perdas devido a congestionamento, pois obrigaria o protocolo TCP a efectuar as correspondentes retransmissões implicando uma significativa degradação no desempenho. Como a grande prioridade do ATLANTIC consiste na satisfação dos requisitos de tempo real exigidos por algumas aplicações, as alternativas reduzem-se às categorias de serviço rt-VBR e CBR, se bem que no ATLANTIC algumas situações se adequem à categoria de tráfego ABR (sem requisitos de tempo real).

Um contrato CBR assegura largura de banda reservada por canal virtual (com a reserva de recursos feita com base no valor PCR) e como tal para as ligações TCP associadas. Deste modo, em princípio, apenas os erros de transmissão causam perdas de pacotes TCP. A diferença entre os contratos CBR e rt-VBR reside sobretudo no facto de o primeiro ser menos eficiente no aproveitamento dos recursos. A diferença é todavia atenuada pelo facto de se manipularem fluxos MPEG-2 SPTS, que em muitas circunstâncias não variam significativamente.

A categoria de serviço ATM a seleccionar deve ser CBR pois para além de ser adequada para tráfego CBR (ex.: fluxos MPEG-2 com débito constante) é também apropriada para tráfego VBR em que se pretendam garantias absolutas (dito "quasi CBR"). Neste último caso, sempre que a fonte transmitir com débito inferior ao valor de PCR negociado, os recursos são desperdiçados não sendo reutilizados por outras categorias de serviço. Portanto, quanto mais *bursty* for o tráfego menor será a eficiência. No entanto, a adopção da categoria de serviço CBR, para tráfego MPEG-2 de débito variável mas em que a variabilidade é pequena, pode ser um bom compromisso uma vez que a flutuação de débito pouco significativa torna compensatória a opção por uma solução simples e com garantias. No ATLANTIC, actualmente, o facto do tráfego MPEG-2 apresentar débito praticamente constante corrobora a opção pela categoria de tráfego CBR. Tal não implica porém, que a fonte tenha de ser "rigorosamente" CBR. Num cenário onde o fluxo MPEG-2 SPTS assentasse directamente sobre o ATM, como especificado em [1, 2] relativo ao serviço de vídeo a pedido, o contrato teria necessariamente de negociar a classe de tráfego CBR uma vez que não existe controlo de fluxo (e se supõe da parte do servidor um serviço CPR). No caso de se exceder a capacidade dos *buffers* resultaria a perda de informação ou, para evitá-la, um elevado esforço de bufferização na recepção.

A camada AAL5, pela sua simplicidade, apresenta-se como a mais adequada aos propósitos do estúdio ATLANTIC, tendo sido oportunamente descrita neste capítulo.

No estúdio ATLANTIC é vantajoso estruturar as máquinas pertencentes ao estúdio numa rede lógica. Assim, apesar de não se recorrer ao encaminhamento IP, pois todas as unidades interligadas são contactadas por meio de uma ligação ATM, pode organizar-se as máquinas numa LIS que para além de proporcionar um mecanismo de segurança, delimita do ponto de vista lógico a rede. A contrapartida é a sobreposição do protocolo IP ao serviço orientado às ligações ATM que conduz à adição redundante do cabeçalho IP aos pacotes TCP.

Como se referiu, as duas soluções mais correntes para o suporte de TCP/IP sobre ATM são a emulação de LANs, proposta pelo ATM Forum, e o CLIP, proposto pelo IETF. A primeira alternativa como suporta qualquer protocolo introduz um cabeçalho maior. O CLIP, tendo sido concebido especialmente para o suporte de TCP/IP sobre ATM, revela-se como a solução óbvia, considerando que no ATLANTIC se adoptou o TCP pelas razões discutidas. No entanto, o CLIP não negoceia recursos pelo que um processo de associar uma ligação ATM a uma ligação TCP/IP tem de ser encontrado. No capítulo seguinte descreve-se o AREQUIPA a propósito deste requisito.

Em presença de uma rede privada ATM, caracterizada tipicamente pelo estabelecimento rápido e económico de ligações, com a garantia de que imediatamente acima do ATM apenas o protocolo IP será suportado, justifica-se a multiplexagem baseada em VCs [32] também conhecida por encapsulamento nulo pois, como cada protocolo é transportado num VC separado, não há a necessidade de transportar explicitamente informação no campo de dados das tramas CPCS-PDU da camada AAL5. Esta decisão acarreta como benefícios a minimização do acréscimo de processamento e de largura de banda. O protocolo transportado pode ser configurado manualmente ou negociado dinamicamente usando procedimentos de sinalização, durante o estabelecimento das ligações [33], como se preconiza no presente trabalho.

3.5 Conclusão

A descrição do estúdio ATLANTIC, ao permitir a identificação dos requisitos de comunicações, abriu caminho para a escolha da pilha protocolar a suportar.

Os protocolos de comunicação envolvidos, o ATM e o TCP/IP, foram aqui descritos na perspectiva da sua aplicação no ATLANTIC.

A cobertura dos aspectos de maior relevo de cada protocolo constituiu uma base para o conjunto de decisões tomado relativamente a cada protocolo, definido-se assim a arquitectura de comunicações do ATLANTIC.

4 O ATM na plataforma Linux

A escolha do sistema operativo Linux como ambiente de implementação e teste assenta no facto de todo o código fonte ser de domínio público e na existência de uma intensa actividade de desenvolvimento visando o suporte do ATM neste ambiente. A disponibilização de ferramentas para o suporte do ATM no Linux é feito num pacote de *software*, continuamente disponível através da Internet, e objecto de sucessivos trabalhos de actualização. Para além disso, a integração do TCP/IP no *kernel* do sistema operativo e o acesso ao respectivo código fonte foram factores determinantes na consolidação da opção pelo Linux, até pela relevância que adquirem no âmbito do ATLANTIC.

4.1 O sistema operativo Linux

O sistema operativo Linux é um "clone" do sistema UNIX com uma grande base de apoio. Todo o *kernel* e a totalidade das aplicações necessárias à constituição de um sistema UNIX estão disponíveis gratuitamente. O código fonte correspondente está também disponível sob a licença GPL (*General Public License*) da GNU.

O Linux é um sistema operativo aberto. O ambiente de desenvolvimento é deveras amigável, até pelo facto de tantas pessoas se encontrarem debruçadas sobre as diversas componentes do sistema operativo. Assim, na eventualidade de problemas surgidos no decurso de algum trabalho sobre esta plataforma, o pedido de auxílio pode encontrar múltiplos ecos.

Não deixa de impressionar o facto de o Linux ter resultado do trabalho voluntarioso de uma vasta comunidade académica e de investigação. Como contrapartida, podem surgir atrasos de desenvolvimento. As inovações tecnológicas, por exemplo, são habitualmente de difícil propagação uma vez que as correspondentes especificações e licensas ou são de difícil acesso ou exigem algum esforço monetário. Estão deste modo constituídos alguns obstáculos ao desenvolvimento fluido do Linux.

4.1.1 Interface de *sockets* BSD

A interface do protocolo TCP/IP no Linux é baseda na API (*Application Programming Interface*) de *sockets* BSD, comum à grande parte dos sistemas operativos UNIX. A interface do ATM surge no Linux fortemente inspirada na interface de *sockets* BSD e encontra-se detalhadamente documentada [34].

Para um determinado protocolo, é possível criar uma associação entre fonte e destino, com o objectivo de transferir dados, de forma orientada às ligações ou não.

4.1.2 Estrutura do Linux

No sistema operativo Linux podem destacar-se duas grandes componentes: o *kernel* e o *user space*. O primeiro é usado por código compilado no *kernel* ou por um módulos de *kernel* carregados após o período de inicialização do sistema operativo. Componentes que necessitem de aceder a interrupções do *hardware* e a outras funções só disponíveis neste espaço são, por exemplo, os *drivers* para suporte de determinados componentes. O *user space* destina-se a suportar as aplicações que correm sobre o *kernel*. Como exemplo de aplicações executadas pelos utilizadores e que correm em *user space* pode referir-se o caso dos *daemons* e das aplicações interactivas.

O *kernel* é o coração do sistema operativo. Supervisiona os ficheiros em disco, inicia programas que executa de forma concorrente, atribui memória e recursos aos vários processos activos, recebe informação da rede e envia informação para a mesma. Assim, para aceder ao *hardware* é condição necessária recorrer às ferramentas disponibilizadas pelo *kernel* (por intermédio de chamadas ao sistema).

Do Linux fazem também parte compiladores e as correspondentes bibliotecas (ex.: o compilador GCC e a biblioteca de C).

4.2 O ATM no Linux

O suporte do ATM no ambiente Linux teve início em 1995. Com o objectivo de abrir novas portas a investigadores e académicos, o LRC (*Laboratoire de Réseaux de Communication*) da EPFL encontra-se desde então a trabalhar activamente no suporte do ATM no Linux. Actualmente, grande parte da funcionalidade é já uma realidade para os utilizadores do sistema operativo Linux.

As ligações ATM, PVCs e SVCs, são ambas suportadas embora no caso das ligações dinâmicas, devido à elevada complexidade dos protocolos de sinalização envolvidos, se tenha procedido à introdução de algumas simplificações. A sinalização é, em Linux, implementada por intermédio de um processo *daemon* que corre em *user space*.

Um facto inelutável, a considerar no desenvolvimento e implementação do ATM em Linux, é a constatação da escassa quantidade de aplicações concebidas especificamente para correr sobre ATM. A política de coexistência com as tecnologias de maior divulgação (ex.:

TCP/IP) é por isso óbvia. Na sequência desta orientação, o ATM Forum especificou o LANE[35], enquanto o IETF emitiu o CLIP[31], como forma de suportar imediatamente e sem modificação, aplicações desenvolvidas para LANs e/ou TCP/IP.

No Linux, o suporte do IP sobre ATM baseia-se no RFC 1577 [31]. Para ligações PVC, o transporte de pacotes IP é feito procedendo ao encapsulamento respectivo, definido no RFC 1483 [32]. O estabelecimento de SVCs é também possível, pois o CLIP é baseado no protocolo ATMARP (uma extensão do protocolo ARP) [31, 33]. O ATMARP, implementado no Linux como um *daemon* - `atmarpd`, funciona como esquematizado na figura 15. Tanto o utilizador A como o B executam uma primeira tarefa na qual registam o seu endereço IP e o seu endereço ATM num servidor ATMARP. Se agora o utilizador A desejar estabelecer uma ligação SVC com B, basta conhecer o endereço IP correspondente, para efectuar um pedido ao servidor. Este retorna o endereço ATM de B, com o qual A pode estabelecer a ligação SVC.

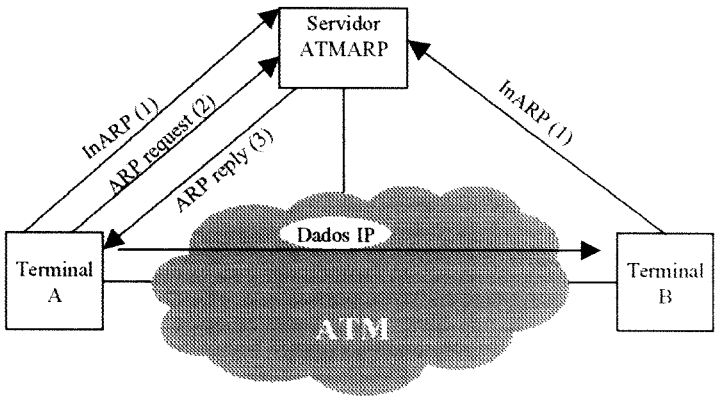


Figura 15 - Fluxo de mensagens ATMARP.

Vários estudos estão em curso no âmbito do IP sobre ATM. O IETF encontra-se a desenvolver o protocolo NHRP [36] que permite estabelecer ligações ATM para além do domínio de uma única sub-rede IP. A integração de mecanismos IP, capazes de negociar com a rede parâmetros de qualidade de serviço, é outra ambição em vias de concretização com o protocolo RSVP[37, 38].

O suporte da especificação LANE encontra-se em gestação na Universidade de Tecnologia de Tampere, Finlândia.

Os primeiros adaptadores ATM a merecer um *driver* foram a carta EN155p da Efficient Networks e a ZN1221 da ZeitNet, ambas para o barramento PCI, capazes de suportar tráfego ATM a 155 Mbps sobre fibra multi-modo.

Os primeiros resultados de avaliação de desempenho com tráfego IP sobre ATM situaram-se, no entanto, muito aquém das expectativas. O número elevado de cópias no *kernel* era o responsável pela degradação observada. A solução passou por adoptar um mecanismo, designado de "single copy", que consiste na cópia directa entre o *user space* e o *device driver*.

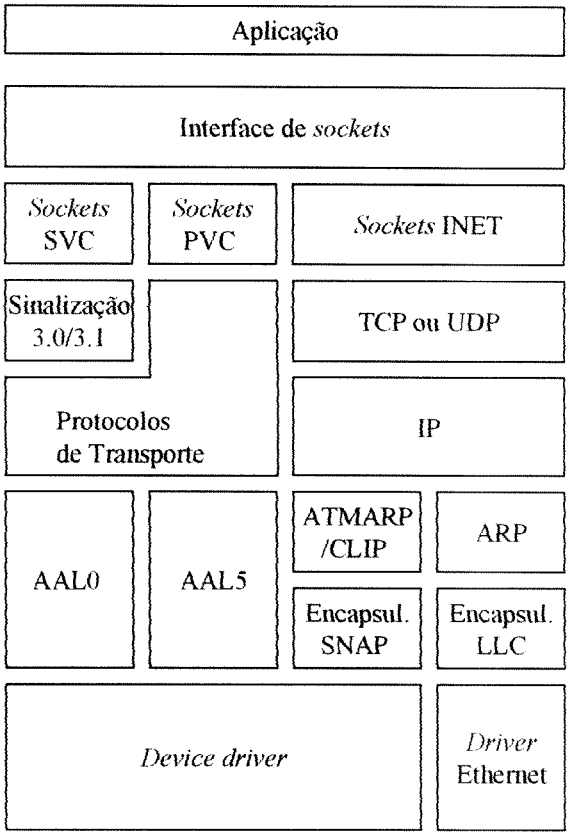


Figura 16 - Protocolos de rede no *kernel* do Linux.

A figura 16 esquematiza a pilha dos protocolos de comunicação mais relevantes, implementados no *kernel*. Os elementos resultantes da adição do ATM podem observar-se no lado direito da figura. De notar que a sinalização UNI 3.0/3.1 é implementada por um processo *daemon* que corre em *user space*. Uma aplicação a correr em Linux fá-lo, naturalmente, sobre a pilha protocolar ilustrada. Tratando-se de uma aplicação ATM, pode utilizar a interface de *sockets* para ligações SVC ou PVC se não recorre aos *sockets* INET (InterNET). É na interface de *sockets* que os dados têm de ser copiados uma vez que o *kernel* e o *user space* correm em zonas de memória diferentes. Os SDUs AAL5 são copiados para a memória principal através do barramento, sob controlo do *device driver*. Ao nível do IP, a escolha pode recair sobre o uso da *Ethernet* ou da interface ATMARP/CLIP para transporte dos pacotes IP sobre ATM. Apesar de não estar ilustrado na figura 16, o *device driver* inclui

uma componente, o *device driver* INESC, descrito atrás no contexto do *testbed* no INESC, capaz de possibilitar a utilização das cartas ATM da Fore, como dispositivos de interface física.

Em [39, 40] demonstra-se a funcionalidade das várias ferramentas contidas no pacote de *software* correspondente, através de um exemplo de configuração de dois PCs para suporte do CLIP.

4.2.1 Estado da arte do ATM no Linux

O suporte do ATM no ambiente Linux está em contínuo progresso. No momento da elaboração da presente tese, a última versão do *patch* ATM, e portanto a usada no *testbed* do INESC, é a versão 0.31 que corre sobre a versão 2.025 do *kernel*.

A tabela III lista as cartas de interface física (PICs) suportadas.

Tipo de interface física	Construtor	Referência da carta
155 Mbit/s	Efficient Networks	ENI155p-MF/U5
	SMC	ATM Power 155
	Rolf Fiedler	Placa TNETA1570 / UniNET1570
	Zeitnet	ZN1221 / adaptador ATM ZN1225 155 Mbps
25 e 155 Mbit/s	IDT ("NICStAR")	77901 / 77903 (77201 / 77211 SAR)

Tabela III - Cartas de interface física ATM suportadas pelo Linux.

Para além destas, é suportada também uma pseudo interface designada "ATM over TCP" destinada a possibilitar o uso do ATM sobre o protocolo TCP. Sem uma interface física de ATM torna-se possível encapsular células ATM em PDUs do protocolo TCP e proceder ao correspondente transporte numa ligação TCP/IP.No entanto, a carta de interface física usada no *testbed* INESC não é por ora suportada pelo pacote de *software* ATM, daí a razão do desenvolvimento do *device driver* INESC.

Ambos os tipos de ligação ATM, PVC e SVC, são actualmente possíveis. Podem ser bidireccionais e terão forçosamente de ser ponto a ponto neste momento. Relativamente ao tipo de tráfego, é apenas possível transportar tráfego UBR ou CBR. A sinalização utilizada no estabelecimento das ligações SVC pode ser a UNI 3.0 ou a UNI 3.1 (conforme o estado da *flag* respectiva no *daemon* `atmsigd`). A sinalização UNI 4.0 encontra-se em fase de desenvolvimento não estando, no entanto, ainda disponível. Assim que tal aconteça, o suporte de ligações ponto a multi-ponto poderá ser suportado. As categorias de serviço ABR e VBR estão igualmente por suportar.

Na utilização da extensão da interface de *sockets* para o ATM apenas são possíveis duas alternativas relativamente à camada de adaptação AAL: AAL0 (células ATM sem AAL) e AAL5. O CLIP encontra-se também disponível. O serviço ATMARP, igualmente fornecido, corresponde ao definido no RFC 1577 [31] e o encapsulamento pode ser nulo ou LLC/SNAP.

A actual versão do pacote de *software* providencia também o suporte de LANE (LAN Emulation) que inclui todos os módulos (LES, LEC, LECS e BUS).

Adicionalmente, é suportado um mecanismo desenvolvido no decorrer do projecto "Web over ATM", designado por AREQUIPA (Application REQuested IP over ATM), que se constituiu já num RFC [41]. Este mecanismo permite associar uma ligação ATM, com qualidade de serviço negociada, a uma conexão TCP/IP e portanto assegurar qualidade de serviço a ligações TCP.

4.2.2 API de *sockets* ATM

A evolução do presente trabalho, apesar de ter como objectivo implementar uma solução final em que o protocolo TCP corre sobre ATM, passou anteriormente por uma fase de desenvolvimento, na qual apenas o ATM foi utilizado. Na base desta opção podem ser invocadas várias razões: teste e estudo dos mecanismos do ATM em Linux e identificação de eventuais problemas que pudessem surgir ao nível do ATM com repercussões a outros níveis.

Usando apenas o ATM, para o estabelecimento de ligações PVC ou SVC, para além do *device driver* é necessário o recurso a uma API. Em paralelo com o desenvolvimento do *device driver* foi desenvolvida de forma articulada uma API baseada na interface de *sockets* BSD. Apesar de não corresponder à especificação "Native ATM Services", elaborada pelo grupo de trabalho SAA API do ATM Forum, as evoluções deste trabalho estão a ser seguidas de forma a possibilitar o acompanhamento de mudanças significativas.

O grupo de trabalho SAA Directory do ATM Forum encontra-se a definir um serviço de resolução de nomes ATM (ANS – *ATM Name Service*) e a estudar a representação textual dos respectivos endereços. Este trabalho está também a ser considerado para posteriores versões da API. Na versão actual do pacote de *software* é possível associar nomes a endereços ATM recorrendo ao ficheiro `/etc/atm.hosts`.

Esta interface constitui uma extensão da interface de *sockets* 4.3 BSD para UNIX e suporta todas as funcionalidades adicionais requeridas pela tecnologia ATM. Neste contexto, configurar uma ligação PVC ou SVC consiste no uso de uma interface de *sockets*.

Uma ligação PVC ATM requer, no entanto, procedimentos de gestão prévios executados no comutador ATM da Fore através da AMI, de acordo com [42]. Para uma ligação PVC ATM bidireccional, por exemplo, é necessário implementar as seguintes instruções:

```
localhost::configuration vcc >new <port1><vpi1><vci1><port2><vpi2><vci2>
localhost::configuration vcc >new <port2><vpi2><vci2><port1><vpi1><vci1>
```

Os procedimentos posteriores, inerentes ao estabelecimento de uma ligação PVC, usam a interface de *sockets* de forma semelhante ao que se passa para o protocolo TCP/IP.

Uma ligação ATM, quer se trate de um PVC ou de um SVC, passa tipicamente por quatro fases como descrito em [34]: a preparação da ligação, o seu estabelecimento, a troca de dados e por fim eliminação da ligação. Segue-se um exemplo ilustrativo da forma como uma ligação PVC é estabelecida e controlada do lado do cliente.

```
struct sockaddr_atmpvc addr;
struct atm_qos qos;
int s,size;

/*PREPARAÇÃO DA LIGAÇÃO*/
/*criação do socket*/
if ((s = socket(PF_ATMPVC,SOCK_DGRAM,0)) < 0) {
  perror("socket");
  return -1;
}
/*inicialização do descritor de endereço*/
memset(&addr,0,sizeof(addr));

/*inicialização do descritor de qualidade de serviço*/
memset(&qos,0,sizeof(qos));

/*especificação dos parâmetros de tráfego*/
qos.aal = ATM_AAL5;
qos.txtp.traffic_class = ATM_UBR;
qos.txtp.max_sdu = BSIZE;

/*atribuição dos parâmetros de tráfego*/
if (setsockopt(s,SOL_ATM,SO_ATMQOS,&qos,sizeof(qos)) < 0) {
```

```
    perror("setsockopt SO_ATMQOS");
    return -1;
}
/*ESTABELECIMENTO DA LIGAÇÃO*/
/*designação do endereço*/
if (text2atm(vpivci,(struct sockaddr *) &addr,sizeof(addr),T2A_PVC)<0)
{
    printf("Address not converted!\n");
    return -1;
}
/*estabelecimento da ligação completo*/
if (connect(s,(struct sockaddr *) &addr,sizeof(addr)) < 0) {
    perror("connect");
    return -1;
}
/*TROCA DE DADOS*/
/*envio de dados*/
if (size = write(s, filename, strlen(filename)) < 0)
{
    printf("Error performing write()!\n");
    return -1;
}
else
{
    printf("%d\n",strlen(filename));
    if (size < 0) printf(" (%s)\n",strerror(errno));
    return s;
}
/*TERMINAÇÃO DA LIGAÇÃO*/
close(s);
```

Em termos de código, para criar uma ligação PVC, o primeiro passo consiste na abertura de um *socket* PVC, para associar um identificador a uma ligação previamente estabelecida (por meio dos comandos invocados na AMI), invocando a chamada ao sistema `socket`. O segundo passo envolve a especificação dos parâmetros de qualidade de serviço e o estabelecimento e inicialização da estrutura que contém o endereço. Para abrir o circuito virtual podem ser invocadas as chamadas ao sistema `bind` ou `connect` que, quando usadas com PVCs, têm a mesma funcionalidade. A abertura de PVCs pode ser efectuada segundo um

mecanismo designado "two phase connect": primeiro apenas uma parte do endereço é fornecida e os recursos para a ligação são imediatamene atribuídos após o que se segue uma segunda chamada à função `connect`, desta feita com o endereço completo. A troca de dados pode agora ter lugar até à invocação da chamada ao sistema `close`, responsável pela terminação da ligação.

O estabelecimeno de um SVC recorre a um conjunto de procedimentos semelhante, responsáveis por, de forma transparente, invocarem o *daemon* de sinalização que negociará o estabelecimento dinâmico de uma ligação ATM.

4.2.3 Sinalização

Como já atrás foi afirmado, dada a complexidade dos protocolos de sinalização ATM, no ambiente Linux o suporte de ligações SVC passou por implementar um processo *daemon*. Entre este processo e o *kernel* há um protocolo interno de comunicação, neste momento na sua versão 0.2 [43].

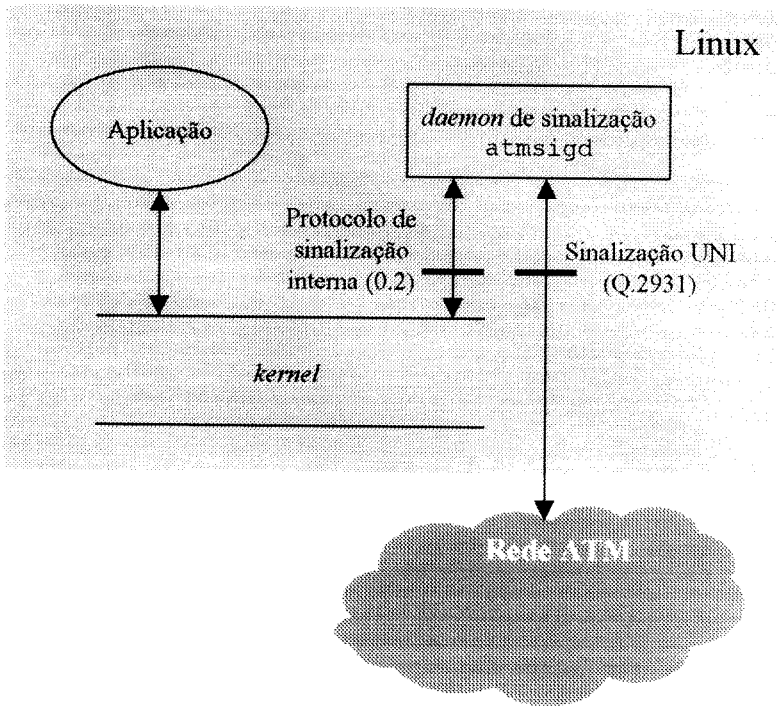


Figura 17 - Sinalização ATM em Linux.

A opção tomada na implementação do suporte do ATM pelo Linux foi portanto, no sentido de relegar toda a complexidade da sinalização ATM para um processo *daemon* que corre em *user space*, o processo `atmsigd`.

O facto de o protocolo de sinalização interna ser substancialmente mais simples que o protocolo Q.2931 é legitimado pelos seguintes pressupostos: a comunicação é bem

comportada (fiável, com garantias de entrega ordenada de células); os pares envolvidos na ligação respeitam a igualdade relativamente a todos os detalhes do protocolo, cooperam sempre e partilham a mesma arquitectura.

A fiabilidade da comunicação bem como as garantias da entrega das células de forma ordenada tornam redundantes os cálculos de *checksum*, os temporizadores de protocolo e as retransmissões. Redundantes tornam-se também os mecanismos que possibilitam compatibilidade de diferentes versões do protocolo de sinalização. Como a arquitectura é coincidente, o código pode ainda ser especificamente escrito adequado à arquitectura comum.

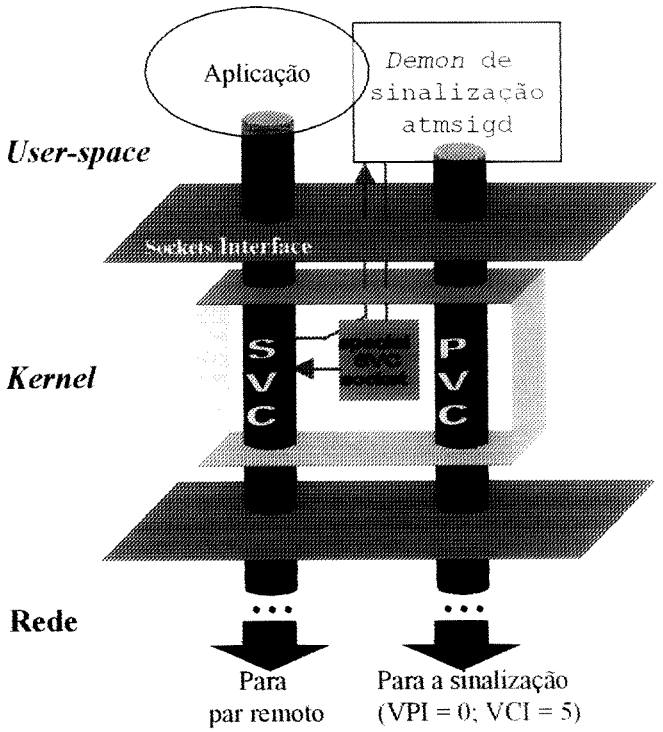


Figura 18 - Procedimentos de sinalização no *kernel*.

O processo `atmsigd` começa por criar uma ligação PVC destinada à troca de mensagens de sinalização com VPI = 0 e VCI = 5, como valores por omissão. Assim o canal de sinalização automaticamente estabelecido pelo comutador é efectivamente utilizado pelo processo *daemon*. Um *socket* SVC especial é também utilizado para troca de mensagens Linux de sinalização interna [43] entre o *kernel* e o *user space*. Ao invocar o estabelecimento de uma ligação SVC, ao nível da aplicação, como especificado em [34], uma mensagem é enviada do *kernel* para o *daemon*, o qual efectua o diálogo de sinalização com a entidade de sinalização do lado da rede. Assim que a ligação esteja estabelecida o processo `atmsigd` notifica o *kernel*. A ligação SVC é ainda localmente tratada pelo *kernel* que tem a responsabilidade de notificar a aplicação assim que o SVC esteja estabelecido por completo.

Adicionalmente, o uso do *daemon* de sinalização proporciona facilidades de analisar as mensagens trocadas entre uma máquina e a entidade de sinalização.

O protocolo ILMI [44], a especificação que descreve a forma de usar o protocolo SNMP e uma MIB ATM, de grande utilidade em operações de configuração foi posteriormente adicionado, igualmente sob a forma de *daemon*, `ilmid`. Uma vez posto a correr usa por omissão o PVC com VPI = 0 e VCI = 16. O protocolo ILMI é usado para configurar os endereços ATM combinando a informação relativa ao endereço local com a informação correspondente ao endereço da rede, fornecida pelo comutador ATM.

4.2.4 AREQUIPA

A disposição protocolar em camadas conjugada com o facto de existir o protocolo IP de permeio, de natureza *connectionless*, aliado à possibilidade de várias conexões TCP serem multiplexadas num único fluxo IP, inviabilizam a associação de uma ligação TCP/IP (endereço IP + porta TCP) a uma ligação ATM (VPI/VCI). Para além disso, é impossível especificar qualidade de serviço numa ligação TCP. As vantagens inerentes ao uso da tecnologia ATM parecem deste modo inibidas pelo cenário final proposto no estúdio ATLANTIC. A garantia de qualidade de serviço proporcionada ao nível da ligação ATM pode, no entanto, ser disponibilizada a uma ligação TCP recorrendo à utilização do AREQUIPA (Application REQUested IP over ATM) [45], já objecto de estudo do ATM Forum [46] e proposto como RFC [41] à IETF. Este mecanismo, desenvolvido na EPFL, permite a máquinas que usufruam de conectividade ATM extremo a extremo dispor de ligações TCP directamente associadas a ligações ATM.

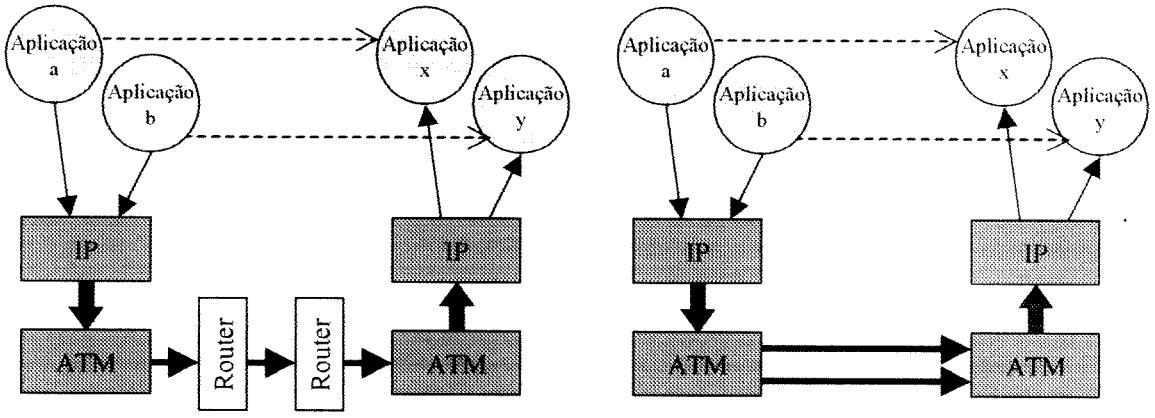


Figura 19 - Comunicação sem e com AREQUIPA.

O AREQUIPA está implementado como um *daemon* que uma vez posto em execução, disponibiliza um conjunto de 3 funções adicionais ao TCP/IP constituintes da API do

AREQUIPA. Ao nível da aplicação é então possível estabelecer uma ligação TCP que, de forma transparente, se estabelece sobre uma ligação ATM, com a qualidade de serviço pedida para satisfazer o pedido ao nível do TCP/IP.A figura 19 ilustra dois cenários demonstrativos da mais valia introduzida pelo AREQUIPA. No primeiro, as ligações virtuais entre as aplicações são multiplexadas ao nível do IP. O tráfego TCP é isolado pelo IP da camada ATM, isto é, o ATM comporta-se como uma qualquer sub-rede. Na segunda situação, as aplicações dispõem de ligações extremo a extremo dedicadas, ao nível da camada ATM.

Ao contrário de outros mecanismos que se aproximam da funcionalidade do AREQUIPA (NHRP e RSVP) apenas as máquinas que usam este mecanismo têm necessidade de ter instalado o *software* correspondente.

4.3 Conclusão

Neste capítulo, foi feita uma breve descrição do ambiente Linux. De forma mais aprofundada, detalha-se uma análise das ferramentas ATM relevantes no contexto do estúdio ATLANTIC.

Conhecidos os objectivos do projecto ATLANTIC é possível constatar as potencialidades do pacote de *software* ATM no Linux na satisfação dos requisitos ATLANTIC.

Uma fase intermédia do desenvolvimento do presente trabalho assenta numa solução exclusivamente ATM. A API revela-se desta forma de grande utilidade no estabelecimento de PVCs ou de SVCs. A necessidade de estabelecer ligações SVC necessita ainda de recorrer ao *daemon* de sinalização disponibilizado. Finalmente, a fase final do trabalho poderá beneficiar do mecanismo AREQUIPA quando, a uma ligação TCP/IP sobre ATM, se pretenda assegurar qualidade de serviço.

5 Protocolo de comando e controlo - DSM-CC

O potencial mercado de serviços multimédia de banda larga adivinha-se promissor a muito curto prazo. A conjuntura assim desenhada torna imperiosa a emergência de protocolos e interfaces abertos, capazes de disponibilizar de forma normalizada serviços provenientes de uma multiplicidade de servidores, passíveis de ser acedidos via computador ou STB.

O DAVIC adoptou o DSM-CC para controlo de sessões multimédia interactivas e gestão dos recursos envolvidos bem como para possibilitar interacções ao nível do serviço. Um exemplo elucidativo da aplicação do DSM-CC está patente na especificação do serviço de vídeo a pedido, do ATM Forum [1, 2].

O DSM-CC foi adoptado pelo ATLANTIC para a troca de comandos ao nível da aplicação, possível por meio de uma interface de utilização semelhante aos vulgares leitores de vídeo. Esta componente do DSM-CC é designada por *User to User* (U-U).

No entanto, para que estas mensagens de controlo e dados (ex.: *streams* MPEG-2) possam ser trocados, é necessário o estabelecimento das respectivas ligações a nível de rede (no caso presente ligações ATM). A parte do DSM-CC proposta para este efeito tem a designação *User to Network* (U-N). A configuração, a gestão das sessões DSM-CC e o controlo dos recursos da rede estarão assim entregues também ao DSM-CC.

Apesar das semelhanças que o contexto ATLANTIC apresenta com o serviço de vídeo a pedido, os seus requisitos ao nível do controlo de recursos são bastante menos exigentes. Como tal, não se justificaria a implementação da componente U-N do DSM-CC na sua totalidade optando-se antes por um modelo simplificado que, seguindo de muito perto a especificação DAVIC, tem como prioridade adequar-se às particularidades do ATLANTIC.

Este capítulo trata com ênfase acrescida a parte U-N do DSM-CC uma vez que o trabalho de implementação, descrito mais adiante, se relaciona de forma muito directa com essa área protocolar. O DSM-CC U-U, em desenvolvimento igualmente no âmbito do ATLANTIC, adquire porém algum relevo no contexto deste trabalho devido à necessidade de integração de ambas as partes do protocolo.

5.1 Modelo de referência do DSM-CC

O DSM-CC foi desenvolvido no sentido de se constituir como protocolo aberto possibilitando como tal, de uma forma normalizada, o fornecimento de serviços aos utilizadores independentemente do serviço ou do servidor.

A história do DSM-CC começou com um anexo à norma MPEG-2 em que o DSM-CC era apenas um protocolo de controlo de MPEG-2 TS. Foi a fase embrionária dum protocolo que passou por uma grande evolução e permite agora o suporte de serviços multimédia de banda larga, constituindo-se como norma internacional do ISO/IEC [5].

A flexibilidade apresenta-se como o ponto fulcral do DSM-CC. Cada protocolo definido no DSM-CC pode ser utilizado de forma isolada ou, pelo contrário, de forma concertada com outros protocolos definidos no DSM-CC, de acordo com os imperativos da aplicação. O modelo protocolar do DSM-CC apresenta uma camada U-U (*User to User*) que pode dispor da camada U-N (*User to Network*) para o estabelecimento e controlo de sessões.

As operações U-U usam um mecanismo de comunicação RPC enquanto as operações U-N recorrem a um método de troca de mensagens, não necessariamente fiável.

O modelo funcional de referência inclui dois utilizadores, um cliente e um servidor, que comunicam através da rede, entendida como um conjunto de elementos de comunicação que proporcionam aos utilizadores as ligações necessárias bem como o controlo sobre as ligações e sobre as sessões.

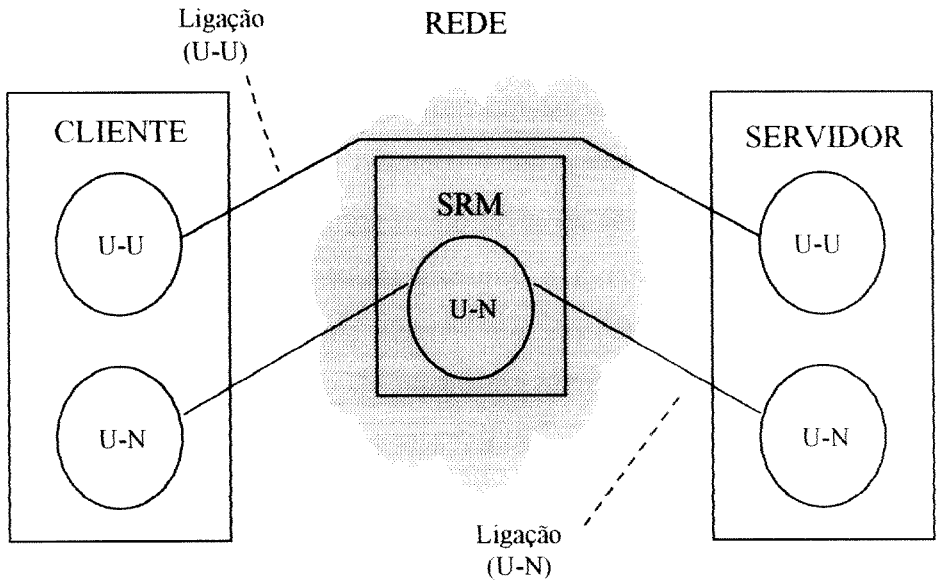


Figura 20 - Modelo de referência de um sistema DSM-CC.

O conceito de sessão é basilar, no contexto do DSM-CC, sendo entendido como a associação entre dois utilizadores e o respectivo conjunto de recursos reservados para uma instância de um determinado serviço. A entidade DSM-CC rede inclui um elemento funcional lógico, o gestor de sessões e recursos – SRM (*Source and Resource Manager*), que pode, no entanto, estar fisicamente distribuído por mais que um sistema. A supervisão das ligações

U-U e a gestão de sessões e recursos associados (a rede tem recursos finitos), o fornecimento de informação de configuração aos utilizadores e a tarefa de autenticação dos clientes estão sob a alçada desta entidade lógica.

A figura 21 mostra as ligações DSM-CC aos dois níveis, U-U e U-N, para transporte de informação U-U e U-N, respectivamente.

O DSM-CC é independente da camada de transporte. Para além disso, é aplicável a um vasto conjunto de redes físicas que poderão nem sequer ser homogéneas. Prevê-se que redes HFC e FTTC se tornem vulgares na interligação de servidores aos clientes, nas áreas residenciais.

As topologias de rede suportadas podem compreender múltiplos servidores, múltiplos clientes e vários protocolos. O suporte de mais que uma sessão é igualmente contemplado. Os tipos de ligação suportados podem ser ponto a ponto ou ponto a multi-ponto, uma vez que serviços de difusão são também contemplados no DSM-CC.

Outro requisito que o DSM-CC não descura é a obrigatoriedade de minimizar a latência visto tratar-se de um protocolo destinado a aplicações interactivas.

5.2 DSM-CC U-U

A componente U-U do DSM-CC abarca dois sub-conjuntos de serviços. No primeiro assegura a compatibilidade entre aplicações (proporcionando ao cliente um método de informar a rede ou o servidor das suas capacidades) que compreende o protocolo de *Download*, um protocolo de transferência de ficheiros de informação de tamanho considerável capaz de suportar serviços com fluxo controlado (ex.: ligações ponto a ponto) ou não (ex.: serviços de difusão). Tal como todas as funcionalidades fornecidas pelo DSM-CC U-U, estas envolvem sinalização U-U apresentando contudo diferenças quanto ao mecanismo de transporte. A gestão da *gateway* de sessão e da *gateway* de serviço ilustradas na figura 21, o controlo do serviço de *streams*, do serviço de ficheiros e do serviço de base de dados e a gestão de eventos, adicionam-se ao conjunto de funcionalidades englobadas nesta parte do protocolo.

O protocolo U-U define, neste último sub-conjunto, os serviços que um servidor pode fornecer de forma interactiva a um cliente. A invocação destes serviços recorre a um mecanismo de RPCs sobre uma ligação U-U estabelecida para o efeito. Acresce ainda que a implementação neste caso é orientada aos objectos e toda a sinalização entre utilizadores é abstraída pelas interfaces definidas em IDL (*Interface Description Language*).

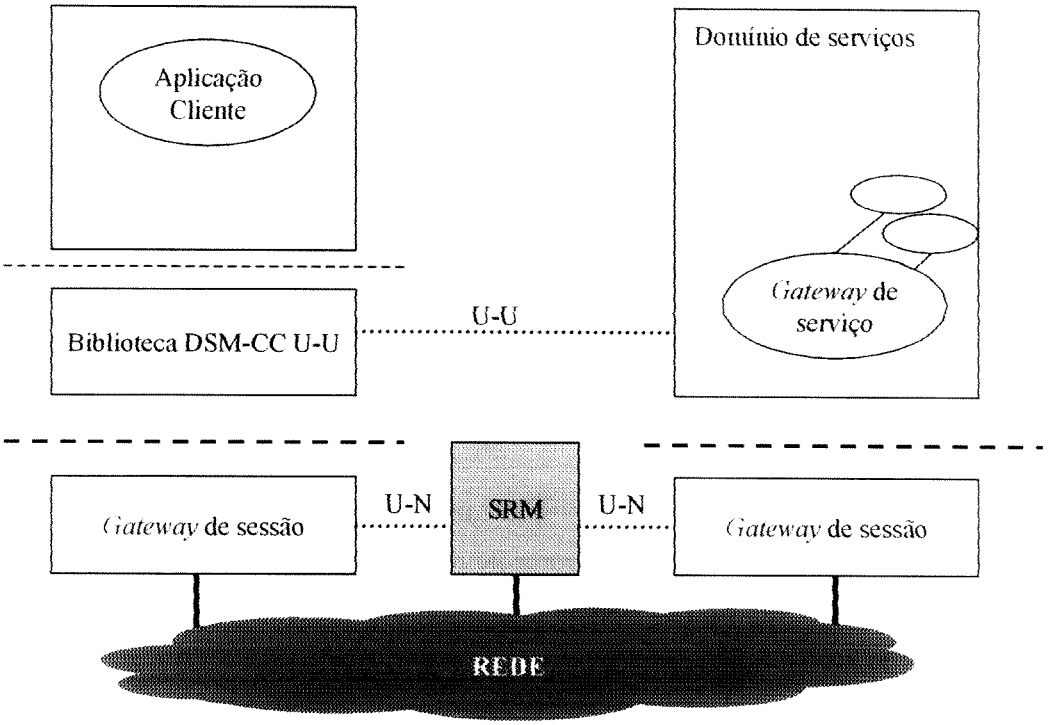


Figura 21 - Ambiente da componente U-U.

Uma sessão DSM-CC compreende uma ligação de controlo U-U e uma ou mais ligações de dados U-U, entre um cliente e um servidor. A primeira destina-se ao transporte mensagens de controlo, suportadas pelo mecanismo de RPC. As ligações destinadas ao transporte de informação constituem a quase totalidade dos recursos de rede atribuídos.

As ligações U-U não têm obrigatoriamente de terminar no mesmo NSAP, do lado do servidor, podendo portanto haver mais que uma fonte de informação.

A norma MPEG-2 não define, para a codificação/descodificação de vídeo, um esquema de temporização adequado a serviços de visualização, como o serviço de vídeo a pedido. Na realidade, um fluxo MPEG-2 obedece a um sistema interno de relógio que permite apenas a sincronização dos fluxos de áudio e vídeo. No entanto, um serviço de vídeo, à semelhança do que acontece nos leitores de vídeo domésticos, tem de possibilitar o posicionamento aleatório do fluxo e uma variedade de velocidades de visualização. Surgiu assim no DSM-CC o conceito de NPT (*Normal Play Time*) que recorre a um esquema de endereçamento temporal. É um "relógio" que um utilizador associa ao "programa". O NPT progride a um ritmo normal quando a visualização ocorre no modo normal, rapidamente quando a visualização é acelerada e regride também de forma acelerada quando a visualização ocorre no sentido inverso.

Associado à interface de *streams*, descrita adiante, concebida para a manipulação de fluxos MPEG-2, existe uma máquina de estados que oferece um conjunto de funções análogo ao disponibilizado por um leitor de vídeo analógico.

A parte U-U oferece um conjunto de interfaces multimédia, possíveis de usar de forma modular na construção de aplicações multimédia, como o vídeo a pedido, televendas, ensino à distância e jogos. As interfaces são definidas na norma [5] em linguagem IDL e, como mostra a figura 22, o DSM-CC U-U proporciona duas interfaces já identificadas na figura 21: *Application Portability Interface* e *Service Interoperability Interface*. A *Application Portability Interface* providencia uma interface para programadores de aplicações destinadas a correr nos clientes; a *Service Interoperability Interface*, por seu turno, está definida para possibilitar a clientes e servidores de diferentes fabricantes interoperarem de forma transparente.

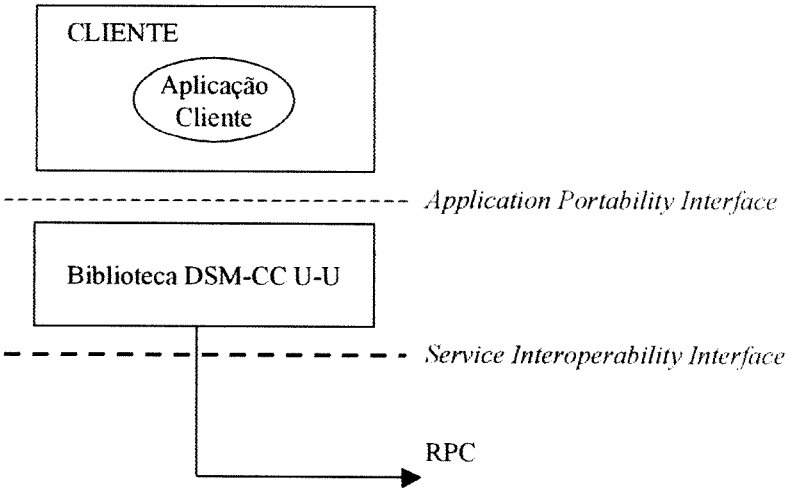


Figura 22 - Interfaces *Application Portability Interface* e *Service Interoperability Interface*.

Apesar de não especificar qual o mecanismo de RPC mais apropriado, o DSM-CC visa permitir a interoperabilidade com a especificação CORBA 2.0 (*Common Object Request Broker: Architecture and Specification, revision 2.0*) [47]. Em conformidade com o enunciado na especificação do DSM-CC, o DAVIC opta pelo mecanismo de RPCs do CORBA 2.0, UNO – *Universal Networked Objects*. No CORBA 2.0, os dados são representados segundo uma sintaxe própria, CDR – *Common Data Representation*, também seguida pelo DSM-CC.

Apesar do elevado grau de flexibilidade inerente ao DSM-CC a ponto de a parte U-U dispensar a parte U-N, é especificada a forma de integração destas duas grandes componentes por forma a implementar um sistema totalmente DSM-CC. Adicionalmente, um cliente ou

servidor DSM-CC pode suportar interfaces de outro tipo permitindo o suporte de aplicações híbridas.

5.2.1 Interfaces U-U

A especificação [5] define para o DSM-CC U-U dois conjuntos de interfaces: o grupo nuclear (*Core Interfaces*) e o grupo de extensão (*Extended Interfaces*). O primeiro compreende as interfaces abstractas (herdadas por outras interfaces) e as instanciáveis.

As interfaces instanciáveis são as seguintes:

Stream - possibilita o controlo interactivo de fluxos MPEG-2 contínuos;

File - oferece aos clientes a possibilidade de ler/escrever ficheiros armazenados no servidor;

Directory - proporciona facilidades de navegação;

BindingIterator - fornece ao cliente a capacidade de processamento de listas de informação;

Session - proporciona a associação aos serviços da interface *ServiceGateway*;

ServiceGateway - oferece um conjunto de seviços e permite a um cliente associar-se a um domínio de serviço.

As interfaces abstractas são comuns a mais que uma interface instanciável. A interface *First*, por exemplo, permite a um cliente a obtenção dos seus primeiros objectos.

O grupo de interfaces de extensão é opcional. As interfaces disponibilizadas são:

SessionGateway - responsável pelo mapeamento de mensagens RPC U-U em mensagens U-N;

Service - permite à interface *Directory* passar informação relativa a um objecto ao cliente (informação de compatibilidade, identificação de utilizador, informação de recursos da rede), na altura em que se determina a referência ao objecto;

Interface - oferece um método de definir e verificar novas interfaces;

LifeCycle - é usada pelas implementações para assegurar referências únicas a objectos num ambiente DSM-CC;

Security - assegura uma forma de passagem de informação de autenticação;

View - permite a ordenação e filtragem de objectos;

Associado à interface de *streams*, descrita adiante, concebida para a manipulação de fluxos MPEG-2, existe uma máquina de estados que oferece um conjunto de funções análogo ao disponibilizado por um leitor de vídeo analógico.

A parte U-U oferece um conjunto de interfaces multimédia, possíveis de usar de forma modular na construção de aplicações multimédia, como o vídeo a pedido, televendas, ensino à distância e jogos. As interfaces são definidas na norma [5] em linguagem IDL e, como mostra a figura 22, o DSM-CC U-U proporciona duas interfaces já identificadas na figura 21: *Application Portability Interface* e *Service Interoperability Interface*. A *Application Portability Interface* providencia uma interface para programadores de aplicações destinadas a correr nos clientes; a *Service Interoperability Interface*, por seu turno, está definida para possibilitar a clientes e servidores de diferentes fabricantes interoperarem de forma transparente.

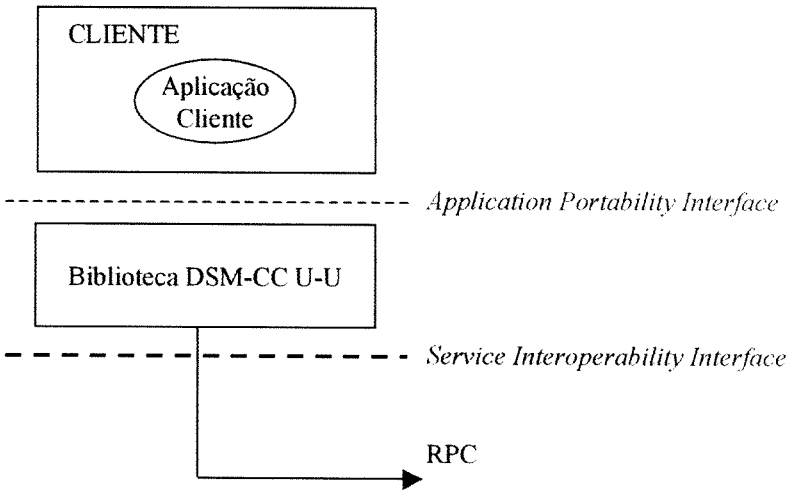


Figura 22 - Interfaces *Application Portability Interface* e *Service Interoperability Interface*.

Apesar de não especificar qual o mecanismo de RPC mais apropriado, o DSM-CC visa permitir a interoperabilidade com a especificação CORBA 2.0 (*Common Object Request Broker: Architecture and Specification, revision 2.0*) [47]. Em conformidade com o enunciado na especificação do DSM-CC, o DAVIC opta pelo mecanismo de RPCs do CORBA 2.0, UNO – *Universal Networked Objects*. No CORBA 2.0, os dados são representados segundo uma sintaxe própria, CDR – *Common Data Representation*, também seguida pelo DSM-CC.

Apesar do elevado grau de flexibilidade inerente ao DSM-CC a ponto de a parte U-U dispensar a parte U-N, é especificada a forma de integração destas duas grandes componentes por forma a implementar um sistema totalmente DSM-CC. Adicionalmente, um cliente ou

servidor DSM-CC pode suportar interfaces de outro tipo permitindo o suporte de aplicações híbridas.

5.2.1 Interfaces U-U

A especificação [5] define para o DSM-CC U-U dois conjuntos de interfaces: o grupo nuclear (*Core Interfaces*) e o grupo de extensão (*Extended Interfaces*). O primeiro compreende as interfaces abstractas (herdadas por outras interfaces) e as instanciáveis.

As interfaces instanciáveis são as seguintes:

Stream - possibilita o controlo interactivo de fluxos MPEG-2 contínuos;

File - oferece aos clientes a possibilidade de ler/escrever ficheiros armazenados no servidor;

Directory - proporciona facilidades de navegação;

BindingIterator - fornece ao cliente a capacidade de processamento de listas de informação;

Session - proporciona a associação aos serviços da interface *ServiceGateway*;

ServiceGateway - oferece um conjunto de seviços e permite a um cliente associar-se a um domínio de serviço.

As interfaces abstractas são comuns a mais que uma interface instanciável. A interface *First*, por exemplo, permite a um cliente a obtenção dos seus primeiros objectos.

O grupo de interfaces de extensão é opcional. As interfaces disponibilizadas são:

SessionGateway - responsável pelo mapeamento de mensagens RPC U-U em mensagens U-N;

Service - permite à interface *Directory* passar informação relativa a um objecto ao cliente (informação de compatibilidade, identificação de utilizador, informação de recursos da rede), na altura em que se determina a referência ao objecto;

Interface - oferece um método de definir e verificar novas interfaces;

LifeCycle - é usada pelas implementações para assegurar referências únicas a objectos num ambiente DSM-CC;

Security - assegura uma forma de passagem de informação de autenticação;

View - permite a ordenação e filtragem de objectos;

Download - implementa um protocolo interactivo para descarregar código ou objectos para aplicações clientes;

Event - permite a adesão/desistência de eventos assíncronos (ex.: um objecto *Stream* pode enviar eventos a um cliente no fluxo MPEG);

Config - oferece uma API através da qual o RPC pode ser configurado para operação síncrona ou assíncrona.

5.2.2 Serviços de difusão digital

O DSM-CC, para além de aplicações interactivas, prevê ainda o acesso a serviços de difusão. Com este propósito é especificado um Carrossel de Objectos (*Object Carousel*) U-U e um protocolo, BIOP – *Broadcast Interoperability Protocol*, capazes de proporcionar um ambiente no qual os objectos U-U (associados a um serviço) são entregues fisicamente aos clientes. O carrossel consiste em periodicamente enviar dados aos clientes. O fluxo associado é unidireccional e portanto cada cliente possui informação que lhe permite decidir que canal de difusão pretende aceder.

Este mecanismo pode ser usado para proporcionar ao cliente o acesso a objectos *Stream* transmitidos em regime de difusão. Para tal, os tipos de objectos U-U suportados são o *Name Space*, *File* e *Stream*.

A adequação a serviços de rede de banda larga motivou o aparecimento de uma área autónoma no âmbito do DSM-CC – o protocolo SDB CC (*Switched Digital Broadcast Channel Change*). Este protocolo é vocacionado para serviços como a difusão de televisão digital. O mercado residencial favorece arquitecturas de rede FTTC associadas a IWUs (*Interworking Units*). Das IWUs até ao cliente circula um número limitado de programas sendo, no entanto, dada a oportunidade ao cliente de dialogar com a IWU respectiva, no sentido de mudar de canal. É neste diálogo que se encaixa o protocolo SDB. As mensagens deste protocolo foram concebidas para usarem o suporte AAL5 sobre ATM.

5.3 DSM-CC U-N

Na parte U-N do DSM-CC é definido um protocolo de interface U-N. Desta componente do protocolo faz parte o protocolo U-N Config. Os utilizadores conseguem por intermédio dele aceder a parâmetros de configuração, relativos à rede, que lhes permitirão funcionar devidamente. O DSM-CC não torna, no entanto, obrigatória a presença deste

protocolo podendo evitar-se mesmo a sua implementação se os utilizadores puderem de outro modo aceder à informação de que necessitam.

A principal motivação da camada protocolar U-N assenta na necessidade de gerir as sessões e recursos subjacentes. Para tal é exigido da camada protocolar de transporte o fornecimento de um serviço mínimo. A troca de mensagens U-N, apesar de não impor requisitos de fiabilidade, exige todavia a detecção e eliminação de mensagens corrompidas. Protocolos como o UDP sobre IP ou AAL5 sobre ATM satisfazem tais condições.

No estabelecimento de uma sessão é gerado um identificador de sessão, o *sessionId*, de carácter global que é associado a todos os recursos atribuídos para essa mesma sessão. A adição ou remoção de recursos numa sessão é dinâmica podendo qualquer uma destas operações ser levada a efeito, mediante as interacções dos utilizadores que, para tal, recorrem a mensagens U-N.

Os recursos invocados são caracterizados, na perspectiva do DSM-CC, através de estruturas de dados designadas descritores de recursos. As mensagens DSM-CC U-N que impliquem a manipulação de recursos incluem campo(s) destinados ao transporte desta(s) estrutura(s) de dados.

Dado o amplo horizonte de aplicação a que se propõe o DSM-CC, a visão que o servidor tem dos recursos é distinta da do cliente. De facto, o cliente e o servidor podem aceder à rede sobre suportes díspares, situação a que correspondem descritores de recursos de tipos distintos.

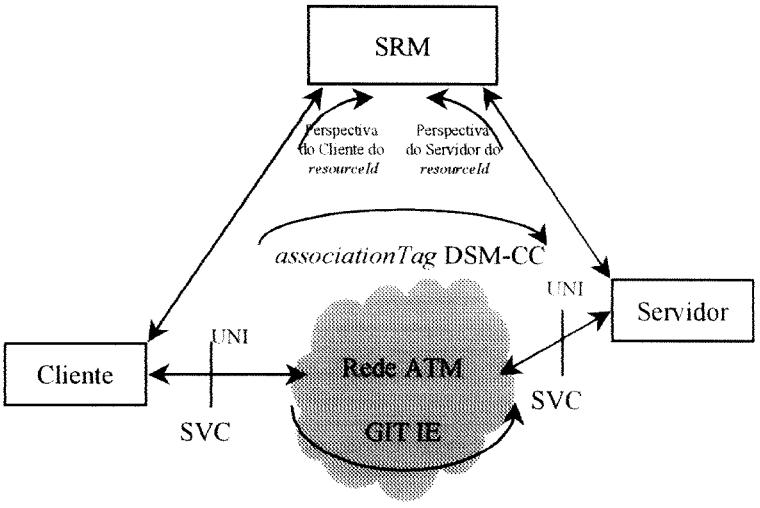


Figura 23 - Cenário ATM extremo a extremo, com controlo da sessão desacoplado da sinalização Q.2931.

protocolo podendo evitar-se mesmo a sua implementação se os utilizadores puderem de outro modo aceder à informação de que necessitam.

A principal motivação da camada protocolar U-N assenta na necessidade de gerir as sessões e recursos subjacentes. Para tal é exigido da camada protocolar de transporte o fornecimento de um serviço mínimo. A troca de mensagens U-N, apesar de não impor requisitos de fiabilidade, exige todavia a detecção e eliminação de mensagens corrompidas. Protocolos como o UDP sobre IP ou AAL5 sobre ATM satisfazem tais condições.

No estabelecimento de uma sessão é gerado um identificador de sessão, o *sessionId*, de carácter global que é associado a todos os recursos atribuídos para essa mesma sessão. A adição ou remoção de recursos numa sessão é dinâmica podendo qualquer uma destas operações ser levada a efeito, mediante as interacções dos utilizadores que, para tal, recorrem a mensagens U-N.

Os recursos invocados são caracterizados, na perspectiva do DSM-CC, através de estruturas de dados designadas descritores de recursos. As mensagens DSM-CC U-N que impliquem a manipulação de recursos incluem campo(s) destinados ao transporte desta(s) estrutura(s) de dados.

Dado o amplo horizonte de aplicação a que se propõe o DSM-CC, a visão que o servidor tem dos recursos é distinta da do cliente. De facto, o cliente e o servidor podem aceder à rede sobre suportes díspares, situação a que correspondem descritores de recursos de tipos distintos.

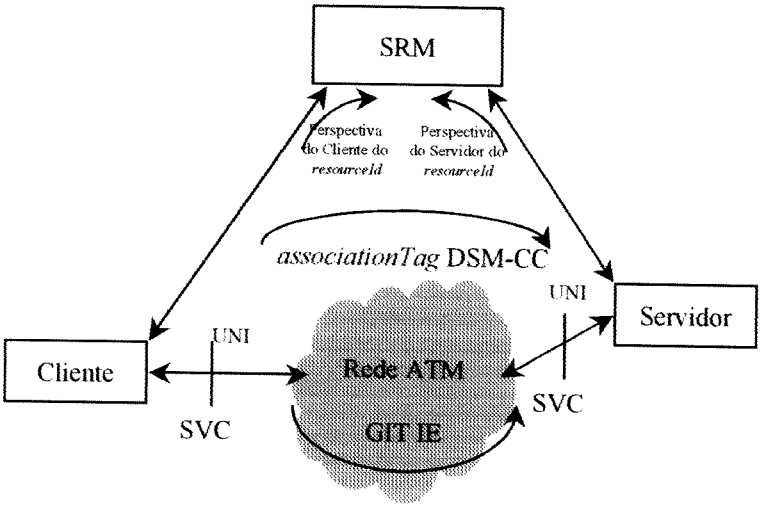


Figura 23 - Cenário ATM extremo a extremo, com controlo da sessão desacoplado da sinalização Q.2931.

Como mostra a figura 23, os recursos de cada ligação de dados U-U incluem nos descritores respectivos um campo, o *associationTag*, com significado global que permite ao cliente e ao servidor associar os vários recursos, vistos sob perspectivas diferentes, a uma mesma ligação de dados. Os valores dos campos *resourceId*[3] e *associationTag* são transportados na mensagem de sinalização ATM usando para tal o elemento de informação GIT que integra a mensagem SETUP da norma Q.2931.

O elemento de informação GIT foi definido pelo SG11 do ITU-T [48] com o propósito específico de transportar o identificador de sessão, *sessionId*, e o número de recurso, *resourceNum*, permitindo deste modo às aplicações associar as ligações ATM com os recursos DSM-CC. Um cliente regista assim todos os recursos ATM atribuídos pelo servidor, através da simples associação do campo *resourceNum* com o campo *sessionId*. A integração destes campos DSM-CC na sinalização ATM garante o suporte de várias sessões do lado do cliente.

Mais de uma ligação U-U pode ser estabelecida no âmbito de uma sessão. Uma aplicação de vídeo a pedido, por exemplo, envolve uma ligação U-U para controlo através da troca de mensagens RPC e pelo menos outra para transporte dos dados MPEG-2.

Uma linha divisória clara é traçada pelo DSM-CC entre gestão de sessão e recursos e os protocolos que gerem as ligações. Por exemplo, o controlo das ligações na rede ATM é implementado através do protocolo de sinalização Q.2931 o que permite o estabelecimento de ligações SVC.

Cada entidade interveniente numa sessão – cliente, servidor e SRM – obedece a uma máquina de estados específica. Num determinado estado uma entidade cliente, por exemplo, está receptiva apenas a um conjunto restrito de eventos que despoletarão acções pré-definidas na transição para o estado seguinte. As tabelas de estado definidas em [5] detalham os eventos que ocorrem no protocolo, as acções a executar, a gestão de temporizadores associados e o estado resultante.

5.4 O DSM-CC aplicado ao serviço de vídeo a pedido

No âmbito deste trabalho, o serviço de vídeo a pedido mereceu particular atenção por revelar fortes semelhanças com os cenários apresentados no estúdio ATLANTIC. Apesar das

[3] *resouceId* = *resourceNum* + *sessionId*; *resourceNum* é o número atribuído ao recurso.

simplificações acarretadas pelos requisitos do ATLANTIC prevalecem ainda paralelismos significativos, nomeadamente no que respeita a protocolos necessários.

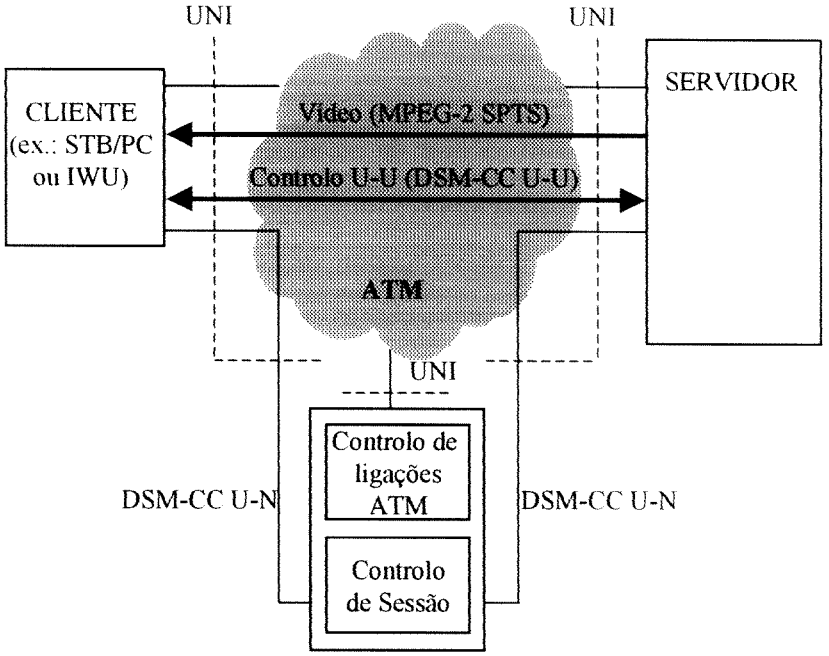


Figura 24 - Configuração de referência do serviço de vídeo a pedido.

Um cenário típico de aplicação do DSM-CC é precisamente o serviço de vídeo a pedido. Um cliente munido de uma STB/PC procede inicialmente à sua auto-configuração através de informação disponível na rede, por meio do protocolo U-N Config. Ocorre então o estabelecimento de uma sessão entre um cliente e um servidor através do uso de mensagens U-N. O servidor estabelece as ligações de dados para proceder à transferência de *software* adequado para o cliente. Uma aplicação de navegação permite a seguir, através do uso de funções da interface *Directory*, o acesso a um menu que lista os serviços disponibilizados. Após a selecção, pode ter lugar uma nova transferência de *software* para o cliente de acordo com a sua opção. Caso ocorra a selecção de um vídeo, as mensagens U-N AddResource estabelecem novas ligações U-U por forma a possibilitar o transporte dos fluxos de vídeo e de áudio. A informação de qualidade de serviço a negociar é para tal extrapolada, de forma a utilizar de modo eficiente os recursos da rede e a adequar os contratos negociados à natureza do tráfego a que se destina cada ligação. O controlo U-U, semelhante ao oferecido por um vídeo doméstico, fica então disponível do lado do cliente.

O ATM Forum definiu em [1, 2] as interfaces requeridas, numa rede ATM, pelo serviço de vídeo a pedido. A preocupação fundamental desta especificação tem a ver com os

aspectos relacionados com o ATM indo também aqui ao encontro da realidade do estúdio ATLANTIC.

Como ilustra a figura 24, da configuração de referência podem ser extrapoladas dois tipos de ligações: as ligações de dados e as de controlo. Nestas, assumiu-se a presença do DSM-CC apesar da especificação contemplar outras soluções. As ligações de controlo, à semelhança do que estabelece o DSM-CC, situam-se em dois níveis. Existe o controlo U-U, através duma ligação U-U destinada a transportar o fluxo de controlo interactivo entre o cliente e o servidor. As ligações U-N permitem o controlo das sessões e das ligações ATM. Tal como enuncia o DSM-CC é clara a distinção entre ambas as funcionalidades. O DSM-CC U-N é responsável pelo controlo de sessões e recursos envolvidos sendo o controlo das ligações de rede entregue ao protocolo de sinalização Q.2931. Todavia, os procedimentos de sinalização são despoletados na sequência de acções DSM-CC U-N.

O modelo protocolar de referência do serviço de vídeo a pedido, implementado quer do lado do cliente quer do lado do servidor, mostra com detalhe o conjunto de protocolos necessário para assegurar o serviço. Mais uma vez, assumindo que o protocolo de controlo é o DSM-CC, constata-se que o controlo U-U é assegurado pelo DSM-CC U-U e o controlo de sessões e de ligações é suportado pelo DSM-CC U-N. O controlo U-U pode, segundo a especificação [5], dispensar uma ligação U-U própria podendo portanto o fluxo de controlo partilhar a ligação de dados. Constitui-se deste modo uma ligação U-U assimétrica. No modelo protocolar depreende-se facilmente esta característica, pois o controlo U-U assenta sobre uma de duas pilhas protocolares. A dos dados ou a do controlo de sessões.

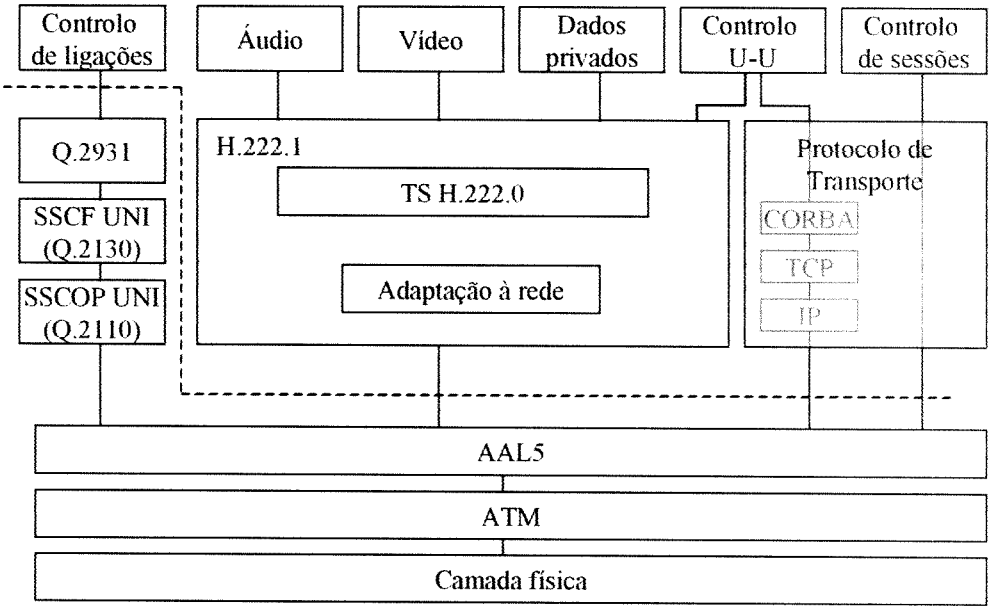


Figura 25 - Modelo protocolar de referência do serviço de vídeo a pedido.

Assumindo a utilização do DSM-CC, o facto de o DSM-CC U-N não exigir do protocolo de transporte fiabilidade permite o transporte das mensagens U-N directamente sobre AAL5. O mecanismo de RPCs asociado ao controlo U-U assenta porém, no caso do DSM-CC, numa plataforma CORBA sobre TCP/IP.

A adaptação à rede dos MPEG-2 SPTS consiste no mapeamento destes pacotes em CPCS-PDU. Cada SPTS MPEG-2 tem 188 octetos. Por omissão um PDU AAL5 transporta dois pacotes de camada superior. O *trailer* do CPCS-PDU implica um acréscimo de oito octetos. O CPCS-PDU obtido mapeia-se em oito células ATM.

As soluções previstas para a implementação da sinalização antecipam uma solução para o caso de ou o cliente ou o servidor ou ambos não suportarem a pilha protocolar associada ao controlo de ligações. Uma sessão genérica é iniciada pela interacção do cliente com o controlador de sessão (ex.: *gateway* de nível 1) através do DSM-CC U-N. Quando o servidor concordar com o estabelecimento da sessão, o controlador de sessão pede ao controlador de ligações ATM o estabelecimento das ligações que necessita. O controlador de ligações, por sua vez, estabelece diálogo com a rede. No caso de não existir interface UNI entre um dos utilizadores e a rede o controlo de ligações é feito por uma entidade *proxy*, o PSA (*Proxy Signalling Agent*), só suportado na UNI 4.0.

5.5 Especificação do protocolo U-N para o ATLANTIC

Para assegurar a funcionalidade exigida pelo ATLANTIC não é necessário implementar a totalidade do protocolo DSM-CC U-N. Esta consideração adquire particular pertinência no presente contexto uma vez que a especificação a elaborar destina-se a uma rede de banda larga, privada e totalmente implementada sobre ATM. Assim, o protocolo a especificar e a implementar seguirá de perto o protocolo DSM-CC U-N no que respeita às mensagens e parâmetros correspondentes e à sequência de acções, mas recorrerá apenas a um sub-conjunto do DSM-CC U-N [49].

A especificação beneficia portanto de um conjunto de circunstâncias simplificativas. É assumido que o protocolo da camada de transporte será o AAL5 sobre ATM. Os requisitos mínimos impostos pelo DSM-CC U-N são satisfeitos por esta solução que explora ainda na totalidade a tecnologia de rede disponível. A troca de mensagens U-N pode deste modo ser feita sobre PVCs pré-configurados através de procedimentos de gestão.

O DSM-CC é do tipo *server centric*, ou seja, o servidor determina os recursos que um dado serviço exige (em função do serviço pedido pelo cliente). Um cliente que inicie uma

sessão e solicite um serviço disponível no servidor despoletará o estabelecimento de uma ligação, caracterizada por parâmetros de tráfego e de qualidade de serviço determinados pelo servidor. Esta especificação assume que o cliente inicia e termina uma sessão ("end to end ATM with first party signalling") ficando a cargo do servidor as decisões relativas a recursos. Tal é possível porque o servidor tem meios para, mediante o conhecimento do serviço solicitado, identificar os recursos necessários.

O número de mensagens U-N será também substancialmente reduzido em resultado de se assumir que a iniciativa de iniciar e terminar uma sessão fica a cargo do cliente.

Outra simplificação, legitimada pela consideração dos vários cenários cliente-servidor no estúdio ATLANTIC, consiste em assumir que todos os recursos de uma sessão são atribuídos na fase de estabelecimento correspondente. As mensagens Add/DeleteResource são como tal dispensáveis, pelo menos na versão inicial.

De acordo com [5] os identificadores *sessionId* e *resourceId* podem ser atribuídos indiferentemente quer pelo cliente quer pelo servidor. A presente especificação assume que o cliente atribui o identificador de sessão uma vez que será, em todas as circunstâncias, a entidade que iniciará o seu estabelecimento. Ao servidor cabe a responsabilidade de designar os identificadores de recursos uma vez que a este caberá a tarefa de os atribuir.

No estúdio ATLANTIC, todas as interacções com relevância para o DSM-CC ocorrem do lado do servidor, factor que pesou na decisão de distribuir o SRM pelos servidores ATLANTIC implementando-o como um processo independente em cada um deles.

Segundo [5], existem quatro métodos de utilização do DSM-CC com a tecnologia ATM. Todos apresentam, no entanto, uma característica comum: o campo *Call Reference* (identificador de uma chamada no protocolo Q.2931) e o valor conjunto do identificador de sessão e do número do recurso, *sessionId* + *resourceNum*, (representativo do conceito de sessão para o DSM-CC) são associados de forma a possibilitar a obtenção de um deles a partir do outro até que a ligação SVC ATM seja libertada.

O método que melhor se adequa ao presente cenário recebe do DAVIC a designação de "Network Method with AddResource messages between the Server and the SRM". Entre os vários cenários físicos possíveis, como o ATLANTIC assenta totalmente sobre a tecnologia ATM, enquanto suporte de rede, o cenário "Client Direct" (a sinalização no cliente, o SRM na rede, referido com o cenário #2 do DAVIC) é o que melhor se adequa aos requisitos do ATLANTIC.

O método adoptado exige que os recursos sejam atribuídos antes de se completar o estabelecimento da sessão respectiva. Para tal, o servidor terá de invocar procedimentos da sinalização Q.2931. A entidade SRM é então notificada da atribuição de recursos por meio de mensagens U-N sendo posteriormente capaz de invocar a libertação de uma ligação.

De acordo com [50], o cenário contemplado nesta especificação recebe a designação de arquitectura "End-to-end ATM, Session control outside Q.2931". Ainda segundo [50], no futuro, a evolução conduzirá a uma arquitectura "End-to-end ATM with integrated Session signalling", ou seja, as mensagens do protocolo DSM-CC U-N vão ser transportadas de forma embebida nas mensagens de sinalização do protocolo Q.2931. Todavia, o actual protocolo de sinalização não permite ainda um tal nível de integração, que se adivinha tão promissor.

Do vasto conjunto de sequências de mensagens que a especificação [5] apresenta apenas dois são relevantes para o problema aqui tratado: o estabelecimento e a terminação de uma sessão, ambos da iniciativa do cliente.

5.5.1 Estabelecimento de uma sessão pelo cliente

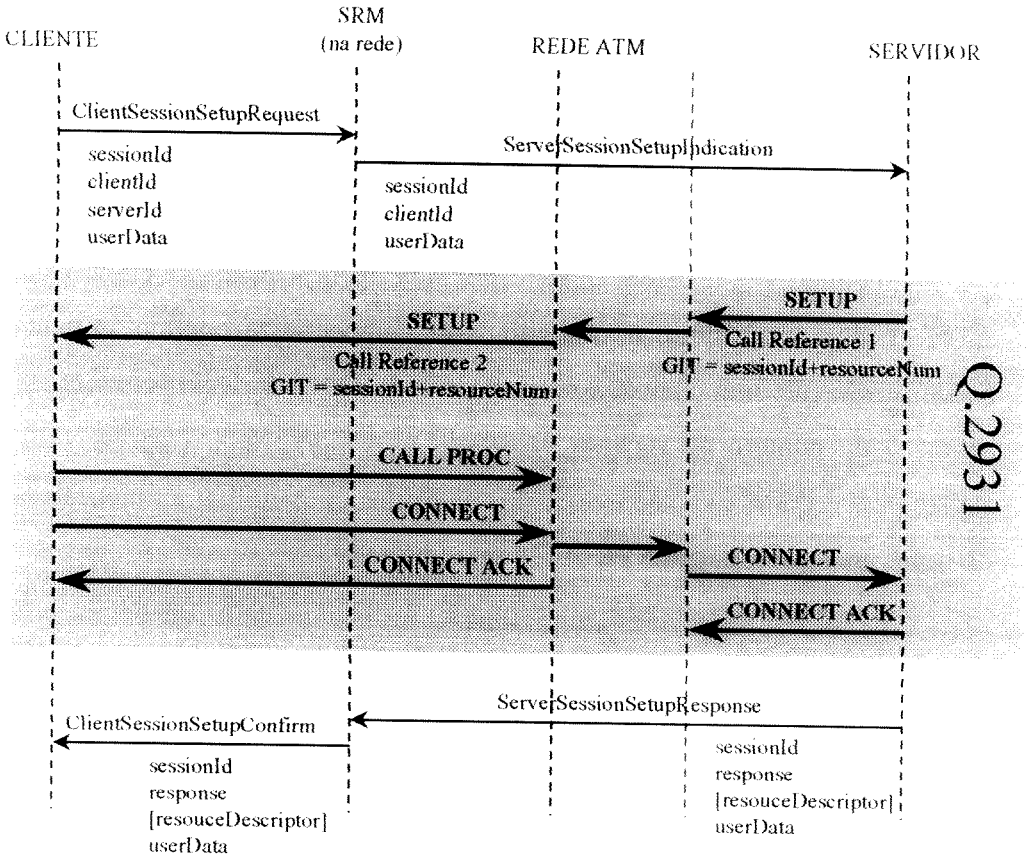


Figura 26 - Estabelecimento de uma sessão por iniciativa do cliente, com o servidor a iniciar o estabelecimento da ligação ATM.

Como já anteriormente foi afirmado, na arquitectura adoptada o servidor ao receber um pedido de estabelecimento de sessão atribui os recursos necessários para o fornecimento do serviço requerido, por meio de procedimentos de sinalização Q.2931 após o que notifica o SRM.

Se algum acordo previamente estabelecido entre o servidor e rede for violado entretanto, em resultado das acções do servidor (a qualidade de serviço excede o valor máximo que a rede se dispõe a garantir, por exemplo), o SRM pode eventualmente terminar a sessão.

A figura 26 mostra a sequência de mensagens DSM-CC U-N e Q.2931 relativas ao estabelecimento de uma sessão. Sublinhe-se o facto do fluxo de informação extremo a extremo, associado à mensagem de SETUP, garantir que os recursos de uma ligação são associados à respectiva sessão.

5.5.1.1 Protocolo DSM-CC U-N

Após atribuir um identificador de sessão, *sessionId*, o cliente envia a mensagem ClientSessionSetupRequest para o SRM. Este verifica o identificador de cliente, *clientId*, segundo a visão da rede, como fornecedora de um serviço (ex.: se o cliente se encontra na lista de subscritores), para avaliar sobre a validade ou não do cliente. Caso o diagnóstico resulte positivo, o SRM contacta o servidor identificado pelo campo *serverId* através da mensagem ServerSessionSetupIndication.

Após a recepção da mensagem que lhe é enviada, o servidor verifica novamente o campo *clientId* com o objectivo de verificar a validade do cliente segundo a sua própria perspectiva. Caso o cliente seja um utilizador válido deste servidor, são identificados todos os recursos de rede requeridos para o suporte do serviço seleccionado. Os recursos são então pedidos à rede iniciando-se assim os procedimentos de sinalização. Cada recurso atribuído implica o preenchimento de uma ou mais estruturas de dados *resourceDescriptor* adequadas ao recurso a que se referem, cujo campo *resourceNum*, como se especificou atrás neste capítulo, é atribuído pelo servidor. Uma vez completados todos estes procedimentos, tem lugar o envio da mensagem ServerSessionSetupResponse ao SRM, que contém toda a informação sobre os recursos atribuídos. A mensagem ServerSessionSetupResponse informa a rede de que o servidor está preparado para fornecer o serviço para que foi solicitado e usar as ligações estabelecidas. Faz parte desta mensagem uma lista de descritores dos recursos atribuídos.

O campo *response* transporta um código que permite ao cliente tomar conhecimento da forma como foi processado o seu pedido de estabelecimento de sessão.

Ao cliente chega então a mensagem ClientSessionSetupConfirm, com uma lista de descritores de recursos caso o campo *response* indique que a sessão foi correctamente estabelecida. Mesmo tratando-se de uma arquitectura ATM extremo a extremo, a lista de descritores de recursos que chega ao cliente não coincide com a gerada pelo servidor, pelo facto de o cliente e o servidor possuirem perspectivas independentes da rede (VPI e VCI diferentes, por exemplo). Este facto, descrito na secção DSM-CC U-N é aqui contrariado por particularidades de implementação. O facto da funcionalidade do SRM ser implementada de forma distribuída pelos servidores implica que entre o servidor e o SRM não sejam necessários recursos de rede, pelo que há uma lista única de recursos numa sessão.

Do ponto de vista do protocolo DSM-CC U-N, está completo o estabelecimento da sessão podendo iniciar-se o fornecimento do serviço invocado.

5.5.1.2 Protocolo Q.2931

A sinalização é iniciada pelo servidor assim que o processo de identificação dos recursos necessários termina. A primeira mensagem, a ser enviada através da interface ATM UNI correspondente, é a mensagem de SETUP que tem de incluir os seguintes elementos:

- *Call Reference*, seleccionado pelo servidor;
- *Calling Party Number* = endereço ATM do servidor;
- *Called Party Number* = endereço ATM do cliente inferido do identificador de cliente *clientId*;
- parâmetros da AAL, ATM *User Cell Rate* e parâmetros de qualidade de serviço;
- elemento de informação GIT = *sessionId* + *resourceNum*.

Tanto o SRM como o servidor têm de manter um registo actualizado dos valores de *Call Reference* e do GIT para que a qualquer altura possam calcular um a partir do outro. Esta associação deve ser mantida até à ligação SVC ATM ser libertada.

O procedimento de estabelecimento de ligações SVC ATM será efectuado tantas vezes quantas o número de ligações requeridas pelo serviço. Nesta especificação, apenas uma ligação desta natureza será em princípio requerida para transporte de dados MPEG-2 ficando

o fluxo de informação de controlo sob a tutela de um mecanismo RPC implementado sobre CORBA.

5.5.2 Terminação de uma sessão a pedido do cliente

Em condições normais, uma sessão termina a pedido do cliente após terem sido satisfeitos os serviços solicitados. Tal como no caso anterior, uma mensagem DSM-CC é previamente enviada do cliente para o servidor tendo a seu cargo a invocação das mensagens de sinalização Q.2931, adequadas à libertação dos recursos atribuídos à sessão em causa.

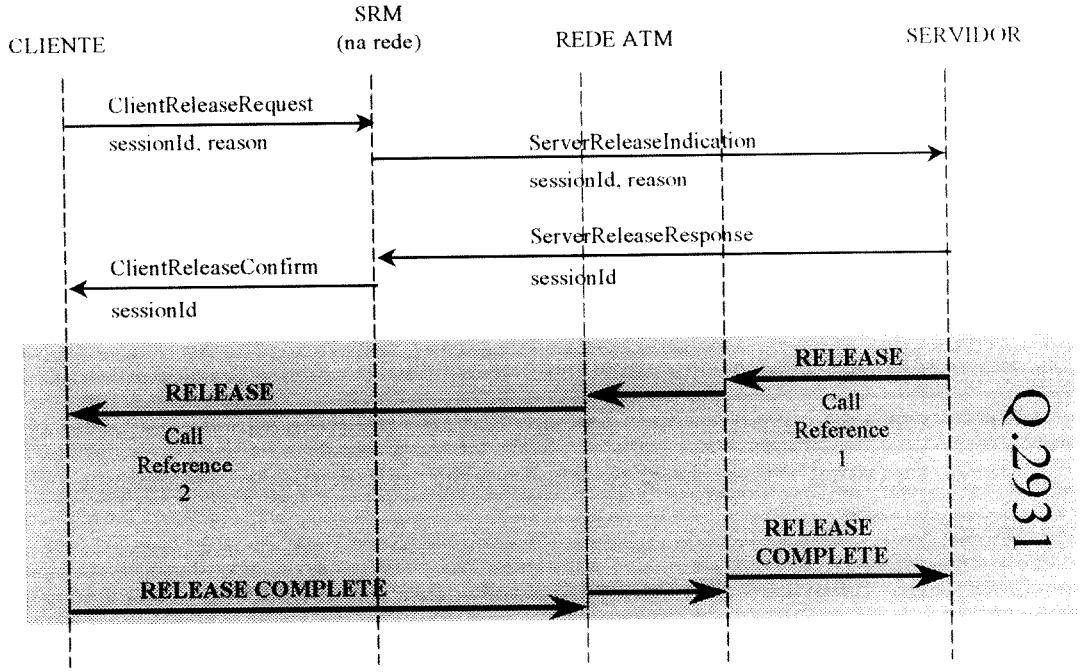


Figura 27 - Terminação de uma sessão por iniciativa do cliente, com as ligações ATM a serem libertadas pelo servidor.

Como a figura 27 mostra, o cliente envia uma mensagem ClientReleaseRequest ao SRM com indicação da sessão que pretende ver terminada bem como a razão pela qual a termina. O SRM notifica o servidor da intenção do cliente através da mensagem ServerReleaseIndication. O servidor responde ao SRM com a emissão da mensagem ServerReleaseResponse que por sua vez, envia ao cliente a mensagem ClientReleaseConfirm.

Assim que esta sequência termina têm lugar os procedimentos de sinalização apropriados. Na realidade, se o servidor encontrar, entre os recursos atribuídos à sessão recentemente terminada, uma ligação SVC ATM pode obter o campo *Call Reference* a partir do valor de *sessionId* + *resourceNum* e iniciar a sequência de *Call/Connection Clearing* na

sua interface ATM UNI. Se efectivamente tal ocorrer é enviada uma mensagem RELEASE do protocolo de sinalização Q.2931 com os seguintes elementos:

- *Call Reference* = *Call Reference* obtido a partir do valor de *sessionId* + *resourceNum*.

O cliente extrai da mensagem de RELEASE enviada o valor do campo *Call Reference* e deste o valor de *sessionId* + *resourceNum* que é posteriormente fornecido a uma função local de gestão, para actualização dos recursos atribuídos ao cliente.

5.5.3 Considerações de compatibilidade

As duas sequências de mensagens, relativas ao estabelecimento e terminação de uma sessão são vitais na presente especificação assegurando totalmente a funcionalidade mínima requerida pelo estúdio ATLANTIC. Cada uma das mensagens DSM-CC envolvidas nas sequências atrás descritas é detalhada no anexo B. À luz a especificação do DSM-CC [5] todas as mensagens envolvidas na especificação do protocolo ATLANTIC U-N pertencem ao grupo funcional "User-to-Network Session and Resource Management Messages Core Group".

Os requisitos de compatibilidade da especificação [5] não impõem que uma implementação compatível com o DSM-CC inclua todos os grupos funcionais listados. Todavia, implementando funções de um grupo, deve-se implementar totalmente a sintaxe e a semântica desse mesmo grupo funcional. Se porventura não se implementarem operações de um grupo, então uma chamada a uma função não implementada deve retornar "Function Not Implemented".

A presente implementação segue assim de perto a especificação do DSM-CC [5] relativamente ao grupo funcional a que se refere. O formato das mensagens identificadas como requeridas, os campos respectivos, as máquinas de estados das entidades cliente, SRM e servidor, serão implementados de acordo com [5], tendo sempre em vista a conformidade com o DSM-CC.

5.5.4 Descrição dos campos das mensagens U-N

Cada mensagem DSM-CC U-N é construída através da concatenação de vários elementos básicos. Cada elemento não é mais que uma variável cujo valor se encontra

confinado a um determinado intervalo, de acordo com o especificado em [5]. O valor que cada um destes campos elementares toma é atribuído pelos utilizadores - cliente ou servidor - ou pelo SRM.

A tabela V do Anexo A lista os campos identificados como necessários bem como o intervalo correspondente onde cada um pode tomar valores.

5.5.4.1 Descritores de recursos

No contexto de um cenário ATM extremo a extremo, os recursos associados a ligações são necessariamente circuitos virtuais ATM. Do vasto leque de descritores de recursos oferecidos pela especificação do DSM-CC [5] (instanciados no campo *resourceDescriptorType* da estrutura DSM-CC que define um recurso) apenas um número reduzido merece análise.

- AtmConnection (0x0002)

Este campo refere-se a um PVC ou a um SVC previamente aestabelecido. Embora se possa pontualmente recorrer a PVCs, como a implementação descrita em [51] ilustra, a arquitectura escolhida assenta na sinalização ATM.

- AtmSvcConnection (0x0007)

Este descritor é usado aquando do pedido de uma ligação ATM. A rede passa este pedido ao dispositivo que na rede estabelece as ligações. É então estabelecida uma ligação através do uso da sinalização. O formato deste descritor contém todos os parâmetros da mensagem de SETUP do protocolo Q.2931, necessários ao estabelecimento de uma ligação SVC ATM. O interesse deste descritor é reduzido uma vez que ambos os terminais, cliente e servidor, suportam a sinalização ATM e como tal os pedidos de ligação são directamente controlados pelos utilizadores intervenientes na sessão.

- AtmConnectionNotify (0x0008)

Como já foi referido, a especificação aqui elaborada assume que o servidor estabelecerá as ligações SVC ATM requeridas notificando *a posteriori* o SRM. Do formato deste descritor não constam quaisquer campos uma vez que a sua finalidade é a de apenas permitir a correlação entre uma ligação e o *resourceId* respectivo.

- IP (0x0009)

O descritor de recurso IP contém os seguintes campos:

- sourceIPAddress (4 octetos);
- sourceIPPort (2 octetos);
- destinationIPAddress (4 octetos);
- destinationIPPort (2 octetos);
- ipProtocol = 0x0001 (identificando o protocolo TCP).

O estúdio ATLANTIC prevê um cenário final de integração do protocolo TCP/IP sobre ATM. No contexto do ATLANTIC, será necessário associar os parâmetros de qualidade de serviço de uma ligação SVC ATM a um endereço IP e correspondente porta TCP. Todavia, o protocolo IP isola a camada ATM que está por baixo e o protocolo TCP não permite a especificação de parâmetros de qualidade de serviço. O estudo destas questões está na base da decisão de adoptar uma estratégia de implementação a dois tempos: primeiro usar apenas o protocolo AAL5 e só posteriormente migrar para a solução final sobre TCP/IP. A segunda fase de desenvolvimento poderá eventualmente recorrer ao uso deste descritor.

5.5.5 Interacção da camada U-U com a camada U-N

Entre as interfaces do protocolo DSM-CC U-U que são consideradas como de extensão, a interface *Session Gateway* tem de ser suportada. Os seguintes aspectos são assegurados por esta interface:

- o mapeamento das mensagens RPC U-U em mensagens U-N (mensagens de estabelecimento de sessão, por exemplo);
- negociação de recursos com o SRM através da invocação de mensagens de adição/libertação de recursos (função dispensável na presente implementação).

Os procedimentos de rede são inicialmente invocados ao nível da aplicação culminando posteriormente na troca de mensagens DSM-CC U-N.

5.5.6 Estabelecimento e terminação de uma sessão

A figura 28 ilustra o conjunto de operações que antecede a emissão de uma mensagem ClientSessionSetupRequest ao nível U-N.

A aplicação cliente invoca a operação `attach` da interface *ServiceGateway*, a qual chama a operação `attachUU` ao nível da interface *Service Inter-operability*. Os parâmetros de entrada desta última operação são passados à mensagem U-N ClientSessionSetupRequest a

qual disponibiliza o campo *User Data* precisamente com esse propósito. Este campo fica assim preenchido:

- inSC - *Service ContextList* (opcional) orientado para o transporte de informação útil ao mecanismo de RPCs mas que pode não ser visível pela aplicação;
- aPrincipal – identificador do utilizador (por exemplo, humano);
- downloadInfoReq – toma o valor 0 nesta especificação significando que não há nenhuma estrutura associada a operações de *Download* ;
- aSuspendContext – valor correspondente a um *User Context* para possibilitar o recomeço de uma aplicação anteriormente suspensa;
- rPathSpec – este parâmetro inclui o *ServiceGateway Name* e opcionalmente o *Service Name*.

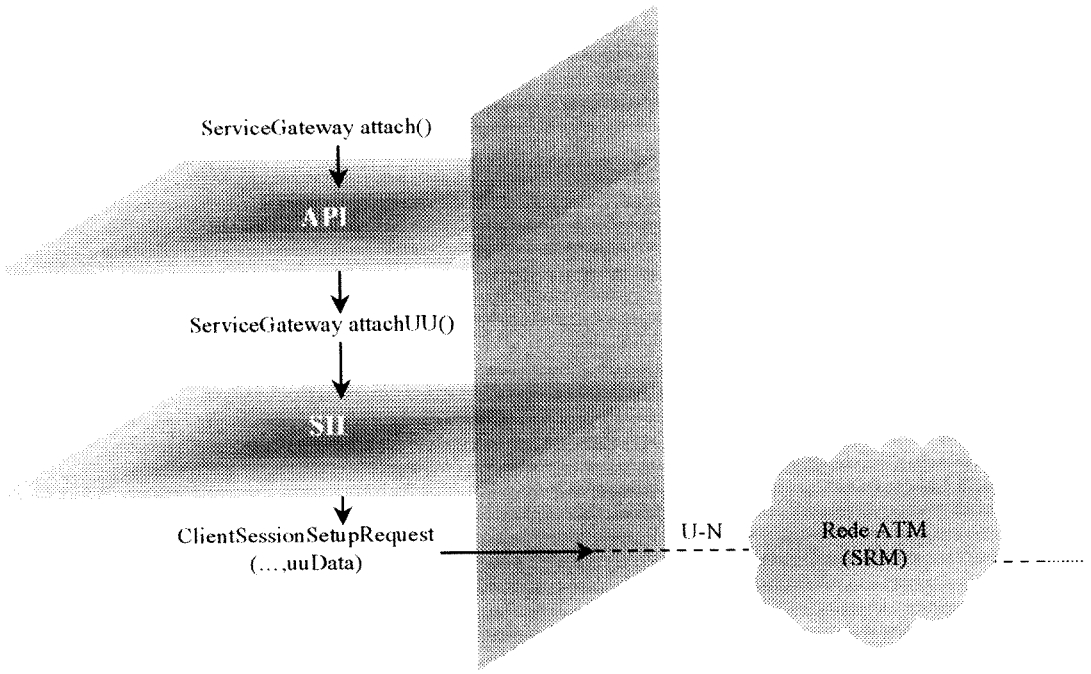


Figura 28 - Estabelecimento de sessão.

O serviço desejado é assim especificado permitindo à interface *SessionGateway* determinar os recursos de rede que devem ser pedidos no estabelecimento de uma sessão.

A mensagem ClientSessionSetupConfirm completa o estabelecimento de uma sessão. O campo *uuData* desta mensagem transporta os parâmetros de retorno da função `attachUU`, do lado do servidor, quer a sessão tenha sido estabelecida com sucesso ou não.

Para a aplicação cliente terminar uma sessão invoca a função `detachUU`, da interface *ServiceGateway*. Os seus parâmetros de entrada são transportados no campo uuData da

mensagem ClientReleaseRequest enquanto os valores de retorno são obtidos do campo *uuData* da mensagem ClientReleaseConfirm.

5.6 Conclusão

A descrição do DSM-CC, efectuada neste capítulo, permite fundamentar a especificação do protocolo de controlo U-N a implementar no estúdio ATLANTIC. Após uma breve descrição da componente U-U do DSM-CC, a análise de um cenário de aplicação, como o serviço de vídeo a pedido, revela-se oportuna sobretudo se se considerar as semelhanças com o estúdio ATLANTIC ao nível de requisitos de controlo.

A componente U-N do DSM-CC mereceu contudo, no âmbito deste trabalho, um maior relevo tendo em conta tratar-se do ponto de partida do trabalho de especificação envolvido.

Esta especificação visa constituir uma base sólida sobre a qual o trabalho de implementação possa firmemente apoiar-se nunca perdendo de vista os requisitos e características particulares dos cenários ATLANTIC e procurando, sempre que possível, ir ao encontro da especificação DSM-CC. Fica assim salvaguardada uma eventual necessidade evolutiva (em resultado por exemplo de um nível de exigência crescente por parte do estúdio ATLANTIC) que se queira forçosamente compatível com o promissor DSM-CC.

Por fim, tem lugar uma breve abordagem à forma como se integram as duas partes do DSM-CC, extrapolada para o caso da especificação do protocolo U-N do ATLANTIC. A necessidade de integrar o controlo U-N com o protocolo DSM-CC U-U é vital, e como tal tratada neste trabalho.

6 Protocolo U-N – Desenvolvimento e Teste

Após a identificação dos requisitos de comunicações no estúdio ATLANTIC, seguiu-se o trabalho de especificação. O levantamento dos vários protocolos envolvidos e a análise das ferramentas e plataformas mais adequadas adquire verdadeiro significado na fase de desenvolvimento e teste.

Este capítulo trata da descrição do trabalho realizado no desenvolvimento do *software* correspondente ao protocolo U-N especificado. Com o propósito de garantir uma integração final pacífica com todos os outros módulos de *software* em desenvolvimento, no âmbito do projecto ATLANTIC, estudou-se primeiro o conjunto de interfaces necessárias tendo como base o modelo arquitectónico de referência.

6.1 Ambientes de execução - CORBA

A consideração da especificação CORBA adquire importância na fase de desenvolvimento do protocolo U-N especificado devido à necessidade de integração com os diversos componentes em desenvolvimento no contexto do ATLANTIC. A componente DSM-CC U-U utiliza, como se referiu, o mecanismo de RPCs fornecido pelo CORBA.

Tendo em vista o teste preliminar do protocolo U-N desenvolvido, a concepção de um emulador simples, implementado em CORBA, pode revelar-se oportuna para ensaio da funcionalidade pretendida ao nível U-N. Outro benefício que uma solução destas acarreta será ao nível da integração final.

O CORBA é uma especificação aberta [47] da qual é possível, actualmente, encontrar diversas implementações.

O OMA (OMG *Object Management Architecture*) pretende definir, com um nível elevado de abstracção, as várias componentes necessárias a uma linguagem distribuída orientada aos objectos. O ORB (*Object Request Broker*), um mecanismo que assegura a transparência relativamente à localização, activação e comunicação dos objectos, constitui o núcleo do OMA.

Na figura 29 está esquematizada uma aplicação distribuída executada sobre CORBA.

De acordo com [52] o CORBA pode ser decomposto em cinco grandes blocos: o ORB, a IDL, a DII (*Dynamic Invocation Interface*), o *Interface Repository* e os *Object Adapters*.

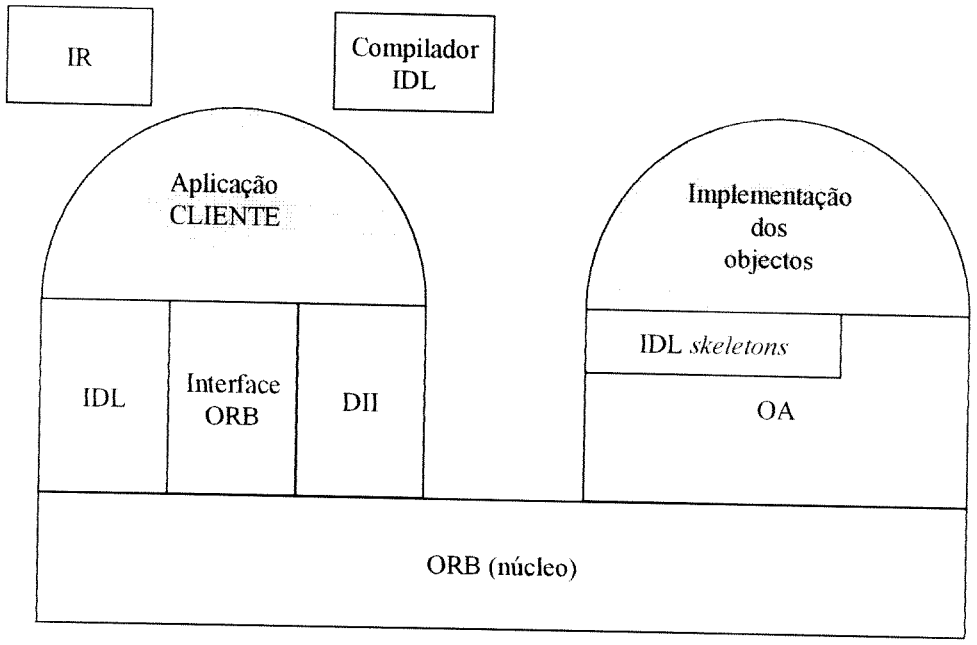


Figura 29 – Aplicação distribuída usando o ORB.

- **ORB**

No modelo OMA, os objectos fornecem serviços e os clientes emitem pedidos invocando esses serviços. O objectivo do ORB é entregar pedidos a objectos que podem eventualmente resultar em retorno aos clientes correspondentes, de forma totalmente transparente para objectos e clientes. O ORB localiza o objecto, activa-o se necessário e entrega-lhe o pedido.Um cliente para emitir um pedido necessita de possuir uma referência para o objecto, IOR (*Interoperable Object Reference*), que o ORB utiliza para identificar e localizar o objecto. Segundo o CORBA o pedido de serviço pode ser estático, por meio de invocações estáticas através de *stubs* específicos da interface, ou dinâmicas, por meio de invocações dinâmicas através da DII do ORB.

- **IDL**

Por forma a evitar que o encapsulamento seja violado, as aplicações distribuídas orientadas a objectos devem manipular interfaces sem terem de preocupar-se com a localização física de um dado objecto. O CORBA pretende responder à imposição que o mercado actual estabelece; os sistemas devem ser abertos e as interfaces normalizadas, para que as plataformas de *software* que os potenciais clientes possuem possam continuar a ser utilizadas. Tal implica não só independência do sistema mas também da própria linguagem. No CORBA as interfaces dos objectos são descritas em linguagem IDL, declarativa e com uma sintaxe semelhante ao C++, permitindo a definição das interfaces dos objectos e separando claramente estes dois conceitos. A especificação do DSM-CC U-U [5] apresenta

também a definição das interfaces U-U escritas em IDL. Uma vez escritas as interfaces podem ser traduzidas para C++, por exemplo.

- **DII**

Esta interface permite a um cliente aceder directamente aos mecanismos de pedidos fornecidos pelo ORB. As aplicações usam a DII para dinamicamente emitirem pedidos sem que tenham necessidade de conhecer as interfaces dos objectos aquando da compilação.

- ***Interface Repository***

Este bloco guarda as referências que descrevem as interfaces dos objectos.

- ***Object Adapters***

O CORBA permite que a implementação dos objectos varie de forma substancial; múltiplas interfaces IDL podem ser implementadas num simples programa servidor ou uma interface IDL pode ser implementada à custa de uma série de *scripts*, um por operação. Algumas destas operações podem ter sido escritas antes da existência do CORBA. Para garantir o suporte de objectos implementados como programas separados cada ORB é suposto proporcionar um *Object Adapter*, o BOA (*Basic Object Adapter*).

6.1.1 GIOP e IIOP

O CORBA especifica um protocolo – o GIOP (*General Inter-ORB Protocol*) – que assegura o inter-funcionamento do ORB. Especifica também o mapeamento do GIOP em TCP/IP, o protocolo IIOP (*Internet Inter-ORB Protocol*), que proporciona o mapeamento da transferência de mensagens GIOP no TCP/IP. Ambos os protocolos foram especificados atendendo a considerações de custo, escalabilidade, simplicidade e generalidade.

A especificação do GIOP assenta no seguinte: CDR, que define representações para todos os tipos de dados OMG IDL; formatos das mensagens GIOP, um conjunto de sete mensagens que asseguram ao cliente e ao servidor CORBA a emissão de pedidos e o respectivo retorno; pressupostos de transporte, o protocolo de transporte deve se orientado às ligações, assegurar a entrega fiável das mensagens GIOP, notificar eventuais situações de dados desordenados e o estabelecimento de ligações deve relacionar-se directamente com o TCP/IP.

Um cliente inicia uma ligação recorrendo à informação no IOR relativo ao objecto a que pretende enviar um pedido. Um servidor apenas pode aceitar uma ligação, nunca iniciá-la. Uma vez aberta, uma ligação pode ser fechada pelo cliente ou pelo servidor.

Um servidor cria objectos e torna as referências a estes (IOR) disponíveis para que os clientes sejam capazes de formular pedidos.

6.1.2 O CORBA no estúdio ATLANTIC

Com o propósito de encorajar o inter-funcionamento, a especificação do DSM-CC [5] propõe o mecanismo de RPCs proporcionado pelo CORBA. A representação de dados CDR é também a proposta.

Como se concluiu em [53], no CORBA, o elevado nível de abstracção conseguido apresenta como contrapartida um desempenho pobre quando comparado com mecanismos com nível de abstracção reduzido, como o de *sockets*. Assim, tratando-se de aplicações com elevados requisitos de desempenho, como é o caso de aplicações presentes no estúdio ATLANTIC, ao CORBA resta um domínio de aplicação restrito.

No estúdio ATLANTIC, o CORBA implementa os mecanismos de RPC necessários à transferência de comandos. A transferência de dados, porém, tem de ser executada através de ligações TCP/IP sobre ATM, como ilustra a figura 30.

6.2 *Interfaces com o protocolo U-N*

A implementação do protocolo DSM-CC U-N exige a interacção entre diversos elementos de diferentes camadas protocolares. Assim as várias interfaces têm de ser asseguradas [54].

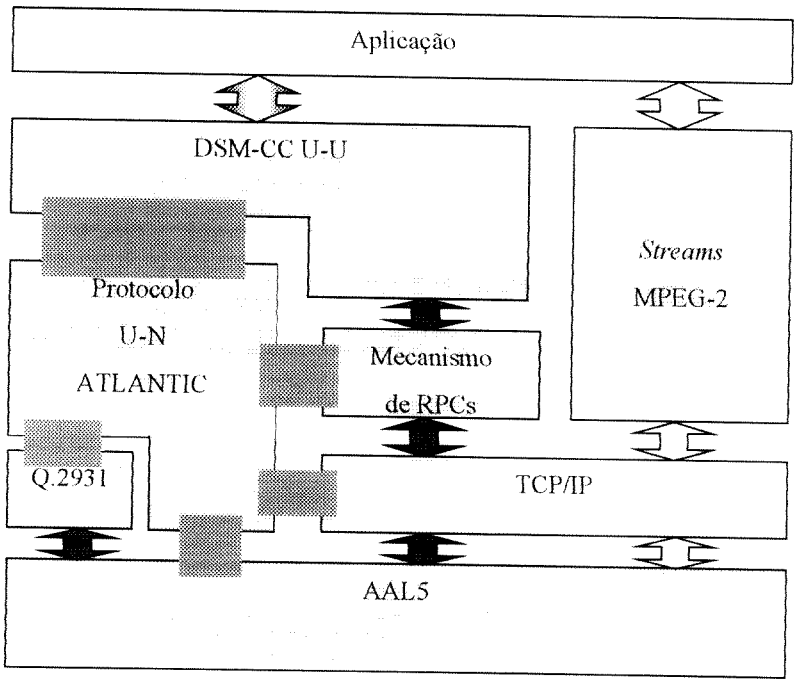


Figura 30 - Interfaces com o protocolo U-N ATLANTIC.

Como enunciado no capítulo 4, a plataforma sobre a qual todo o trabalho de implementação tem lugar é o Linux (versão do *kernel* 2.0.25, versão do *patch* ATM 0.31).

As mensagens DSM-CC U-N são trocadas através da camada protocolar de transporte AAL5 sobre PVCs ATM. Os circuitos virtuais permanentes ATM são previamente configurados por procedimentos de gestão, mas a nível do código são requeridos procedimentos adicionais para que a interacção entre o protocolo U-N e a camada AAL5 seja efectiva.

O protocolo DSM-CC U-N assegura a separação entre o protocolo de gestão de sessões e recursos e o protocolo de controlo de ligações. Este, no presente cenário e em virtude da adopção da tecnologia ATM, respeita a recomendação Q.2931 do ITU-T, aplicável à Especificação de Sinalização da Interface Utilizador-Rede ATM 3.0. Tal impõe que seja definida uma interface entre o protocolo U-N e o protocolo de sinalização Q.2931.

A interface com o TCP/IP sobre ATM tem também de ser garantida uma vez que, na fase final de implementação, as ligações a estabelecer serão TCP. Como se referiu anteriormente, a solução oferecida pelo AREQUIPA vai ao encontro deste requisito.

Na especificação do protocolo DSM-CC U-U é sugerido o CORBA como mecanismo de RPCs a usar. No entanto, como explicado em [47], a camada ORB abstrai o modo como as ligações TCP/IP são estabelecidas e geridas. A monitoração dos recursos atribuídos internamente ao nível U-U (ligação destinada ao fluxo de controlo U-U), torna-se assim complexo.

Adicionalmente, a integração de ambas as partes do DSM-CC requer a troca de dados mútua pois cada mensagem trocada ao nível U-N resulta de uma acção ao nível U-U. É pois requerida uma interface também entre ambas as partes do DSM-CC.

6.2.1 Arquitectura de integração do protocolo U-N

O modelo esquematizado na figura 31 representa a pilha protocolar, associada a uma aplicação, em conformidade com a especificação [5] e com os requisitos impostos pelo estúdio ATLANTIC.

Na figura 31 estão ilustradas de forma hierárquica as camadas protocolares de um cliente DSM-CC, mais concretamente a *Journalist Workstation* do estúdio ATLANTIC. Esta esquematização respeita a parte 7 da especificação [55] e pormenoriza os protocolos das camadas médias e superiores. Neste contexto, o modelo de referência do sistema DSM-CC,

especificado e descrito em [5], inclui uma ligação U-N para transporte de mensagens que consiste num PVC pré configurado sobre o protocolo de transporte AAL5.

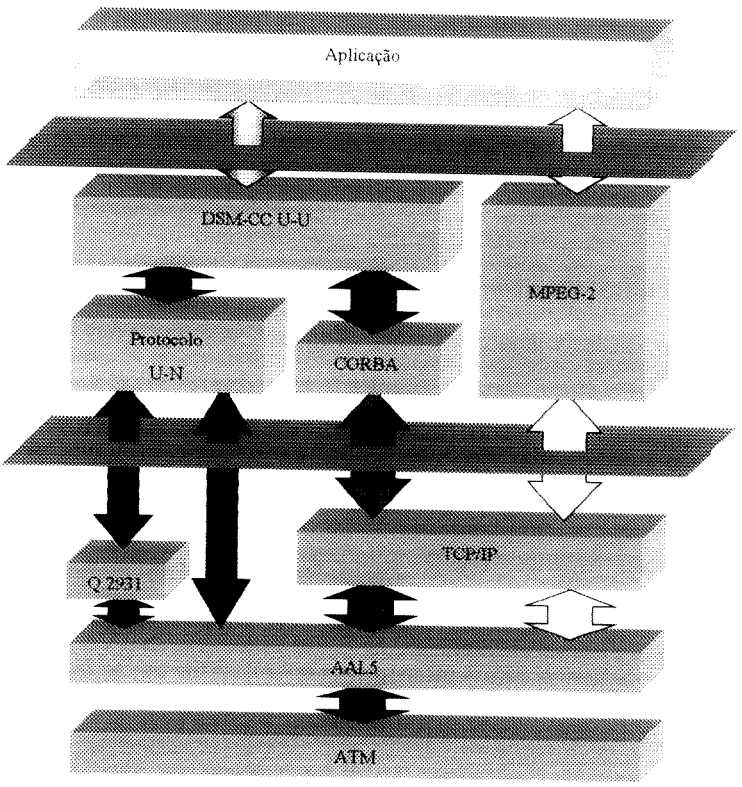


Figura 31 - Modelo protocolar de uma aplicação DSM-CC no estúdio ATLANTIC.

Para transporte de dados do servidor para o cliente o protocolo prevê o uso de ligações SVC estabelecidas e controladas ao nível U-N através do protocolo de sinalização Q.2931. As mensagens de sinalização inerentes circulam num PVC, previamente estabelecido, como prevê a especificação de sinalização [56, 57].

A aplicação cliente envia comandos DSM-CC U-U para o servidor através do mecanismo de RPCs implementado em CORBA.

6.2.2 Interface entre o protocolo U-N e o AAL5

O protocolo AAL5 sobre ATM foi escolhido para suportar a troca de mensagens DSM-CC U-N por respeitar as exigências impostas nesta troca. Não sendo um protocolo fiável, garante a detecção e eliminação de mensagens corrompidas. O facto de o estúdio ATLANTIC ser suportado por uma rede ATM foi também preponderante na tomada desta decisão.

Em [34], como referido no capítulo 4, é definida uma API para serviços relacionados com o ATM no ambiente Linux. Os problemas levantados na manipulação das interacções

entre o protocolo DSM-CC U-N e o protocolo AAL5 ficam deste modo solucionados. Recorrendo à interface de *sockets*, disponibilizada em Linux, o estabelecimento e controlo de ligações PVC é relativamente simples como o demonstra o código listado no Anexo C.

A troca de mensagens DSM-CC U-N está deste modo completamente suportada, na fase actual do trabalho.

6.2.3 Interface entre o protocolo U-N e o protocolo de sinalização Q.2931

O conjunto de mensagens DSM-CC U-N especificado para o estabelecimento e terminação de uma sessão correlaciona-se com um conjunto bem definido de mensagens do protocolo de sinalização Q.2931. Estas são destinadas a estabelecer e a libertar ligações SVC à medida das necessidades do serviço solicitado.

Utilizador → comutador ATM		comutador ATM → Utilizador
Servidor	Cliente	
SETUP	**CALL PROCEEDING**	**SETUP**
CONNECT ACKNOWLEDGE	**CONNECT**	**CONNECT**
RELEASE	**RELEASE COMPLETE**	**CONNECT ACKNOWLEDGE**
		RELEASE
		RELEASE COMPLETE

Tabela IV - Conjunto requerido de mensagens Q.2931.

A interface entre o DSM-CC e a sinalização Q.2931 tem de considerar vários aspectos: o ambiente Linux, o comutador ATM utilizado e o conjunto de mensagens de sinalização requerido.

O comutador ATM usado no *testbed* INESC suporta a interface de sinalização ATM UNI 3.0. Em conformidade com este facto, e de acordo com [42], o comutador estabelece de forma automática um canal de sinalização ATM UNI 3.0. Existe um VCI reservado para a sinalização (VCI = 5). O protocolo ILMI usa por omissão VCI = 16. Estes dois VCIs são atribuídos no VPI = 0.

A tabela IV lista o conjunto de mensagens de sinalização requeridas pelo sub-conjunto de mensagens DSM-CC U-N necessário ao estabelecimento e terminação de uma sessão.

O *kernel* do Linux suporta a sinalização ATM [56, 57] através de um protocolo interno de sinalização muito simples [43]. Tal é possível porque, como se referiu, o ATM proporciona uma comunicação dita bem comportada, os pares intervenientes na comunicação coincidem em todos os detalhes do protocolo, partilham a mesma arquitectura e cooperam em todas as ocasiões.

A sinalização ATM em Linux foi, como detalhadamente é descrito no capítulo 4, implementada num processo *daemon*, o `atmsigd`, que corre em *user space*.

O DSM-CC exige que a mensagem de SETUP do protocolo de sinalização Q.2931 inclua os seguintes elementos:

- endereço do elemento que inicia a chamada;
- endereço do elemento de destino;
- parâmetros da camada de adaptação ATM;
- descritor de tráfego ATM;
- parâmetro de qualidade de serviço;
- elemento de informação GIT.

Como especificado em [10], cada mensagem Q.2931 contém um campo destinado ao valor *Call Reference*. O DSM-CC impõe que em ambos os extremos do canal de comunicação, isto é, no servidor e no cliente, seja estabelecida uma relação bi-unívoca entre o identificador de recurso (*resourceId* = *sessionId* + *resourceNum*) e o valor de *Call Reference*. Esta relação deverá ser mantida até que a ligação SVC ATM seja libertada. Em termos de implementação as repercussões desta exigência traduzem-se em adicionar, ao cliente e ao servidor, rotinas destinadas a tal fim.

No entanto, o uso da plataforma Linux, que implementa o protocolo de sinalização à custa do processo *daemon* `atmsigd`, condiciona alguns aspectos de implementação do protocolo. A versão da sinalização suportada pelo Linux e pelo comutador não permitem ainda a utilização do elemento de informação GIT. Este problema será solucionado por meio de uma mensagem adicional, de carácter proprietário, a ser trocada entre cliente e servidor, destinada a transportar o conteúdo do elemento de informação GIT. Como o Linux avança a passos largos para o suporte da versão 4.0 da especificação ATM UNI [9], a opção aqui tomada poderá brevemente dar lugar a uma outra, mais de acordo com o especificado em [5].

De acordo com o DSM-CC, tem de ser estabelecida uma relação bi-unívoca entre o *resourceId* e o valor de *Call Reference*. Este último é tipicamente atribuído por uma entidade de nível 3 para uso do respectivo protocolo. No contexto particular do Linux, o valor de *Call*

Reference é atribuído pelo *daemon* de sinalização igualmente sem a intervenção da aplicação. Uma vez que o valor de *Call Reference* tem significado local na interface UNI e a relação bi-unívoca entre *resourceId* e *Call Reference* não necessita de ser directa basta que exista um identificador único na máquina simultaneamente associado ao *resourceId* e ao *Call Reference*. A associação entre *socketId* e *Call Reference*, apesar de não ser "visivel" em *user space*, é mantida pelo *daemon*. Conclui-se portanto, que em substituição do *Call Reference* pode muito bem ser utilizado o identificador de *socket* que respeita as condições acima descritas.

6.2.4 Interface do protocolo U-N com o protocolo DSM-CC U-U e com o CORBA

A integração do código desenvolvido no âmbito do protocolo U-N com a parte U-U levanta um conjunto de questões pertinentes: a troca de dados, a forma como o código associado ao protocolo U-N é invocado, troca de informação de qualidade de serviço e identificação de recursos atribuídos pelo CORBA. Como o protocolo DSM-CC U-U é desenvolvido em CORBA é difícil dissociar a interface entre o protocolo U-N com o protocolo DSM-CC U-U da interface entre o protocolo U-N com o CORBA. À partida, sabe-se que a ligação de controlo U-U, ao invés das ligações de dados, estará sob o controlo do CORBA, e portanto fora do alcance do protocolo U-N.

6.3 Implementação da infra-estrutura de comunicação

A primeira aproximação ao protocolo U-N, em termos de implementação, foi feita ao nível das comunicações. O cenário final previsto para a primeira fase dispensa o TCP e como tal assenta nesta infra-estrutura.

O código desenvolvido proporciona a troca de informação, através de um PVC sobre AAL5, entre cliente e servidor. A informação assim trocada, que consiste simplesmente no nome de um ficheiro a transferir, dará na versão final lugar às mensagens DSM-CC U-N.

O servidor, ao receber a informação via PVC, verifica se possui o ficheiro com o nome correspondente. No caso da análise se confirmar, é estabelecida uma ligação SVC destinada ao transporte dos dados contidos no ficheiro. O cliente cria então uma cópia do ficheiro que pediu.

Este cenário tem como objectivo simplesmente emular a estrutura de comunicações, proposta pelo protocolo DSM-CC U-N. Como tal, os parâmetros de qualidade de serviço foram atribuídos de forma arbitrária com os valores mostrados na figura 32.

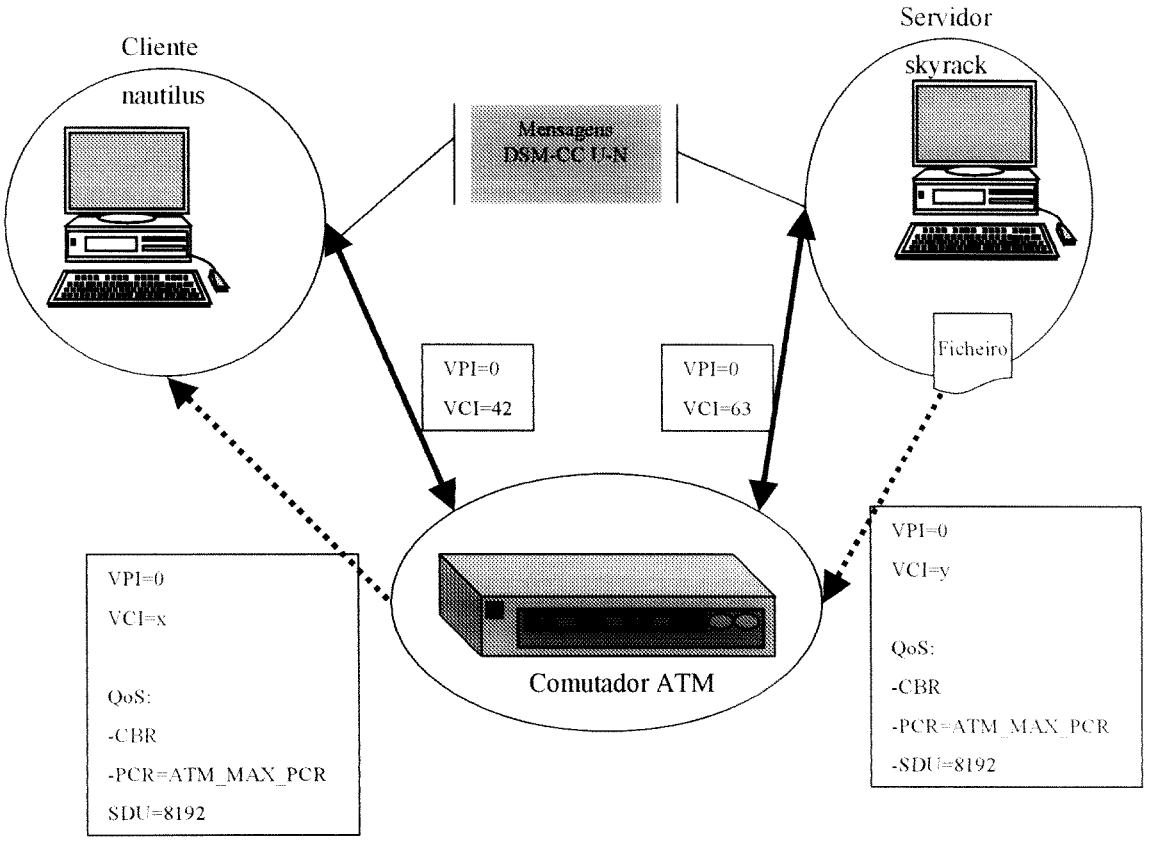


Figura 32 - Infra-estrutura de comunicação.

Como o esqueleto de comunicações permite verificar, o pedido de sessão por um cliente justifica apenas o uso de um descritor de recursos nas mensagens de confirmação provenientes do servidor/SRM: *AtmConnectionNotify*. A ligação ATM pode assim ser correlacionada com o identificador de recurso respectivo (*resourceId*). No entanto, as ligações a estabelecer pelo servidor serão na fase posterior ligações TCP/IP sobre ATM. O descritor de recursos então necessário será o *IP*.

6.4 Arquitectura de software

O prosseguimento da implementação do modelo protocolar especificado passa agora pela definição das estruturas de dados em cada um dos processos: cliente, servidor e SRM.

A entidade SRM será implementada como um processo independente, em cada servidor DSM-CC, por forma a reduzir a complexidade da arquitectura do sistema.

As mensagens DSM-CC U-N são construídas de acordo com [5]. Cada processo tem de ser capaz de construir e descodificar correctamente cada uma das mensagens, listadas no anexo B.

A solução inicial encontrada para as comunicações U-N, suportada directamente sobre AAL,5 migrará para um suporte TCP/IP. No entanto, de acordo com o estado da arte actual, as aplicações TCP/IP não são capazes de usufruir das vantagens que o ATM oferece sendo difícil convertê-las em aplicações nativas ATM. Existe contudo o mecanismo AREQUIPA, que permite às aplicações o estabelecimento de ligações ATM directas extremo a extremo, com uma qualidade de serviço desejada especificada ao nível da ligação. O cenário final de implementação contempla o uso do AREQUIPA que, em primeira análise, serve os propósitos do estúdio ATLANTIC.

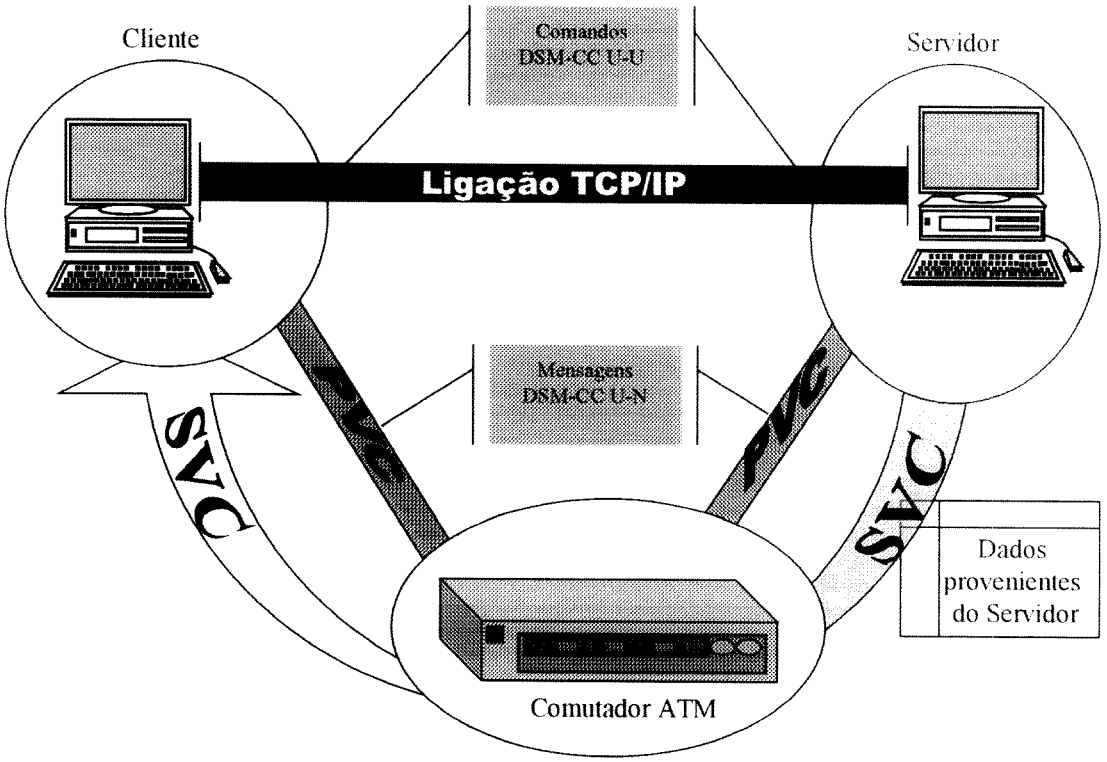


Figura 33 - Cenário final de implementação.

De forma a estudar de perto os problemas levantados pela integração com a parte U-U do protocolo, torna-se oportuna a implementação de um emulador simples desta parte, sobre uma plataforma CORBA. Pretende-se deste modo, assegurar uma migração pacífica para o cenário final.

6.4.1 Modelo arquitectónico do cliente

No modelo arquitectónico do cliente é possível discernir blocos funcionais e definir as interacções que previsivelmente irão ocorre entre eles.

Sobre o bloco responsável pelo estabelecimento e gestão das ligações, o de comunicações, funciona um bloco responsável por duas tarefas: construir e interpretar as

mensagens U-N quando solicitado. As mensagens U-N transportam campos de dados, disponibilizados pela estrutura de dados presente no cliente. O bloco mais importante é, no entanto, o de controlo e gestão que, de acordo com a máquina de estados do cliente definida em [5], executa acções, responde a eventos, transita de estado e interactua com a estrutura de dados (ex.: geração de identificador de sessão, preenchimento da tabela que correlaciona o identificador de *socket* com o *resourceId*). O controlo e gestão tem ainda a seu cargo as tarefas relacionadas com a parte U-U do *software*.

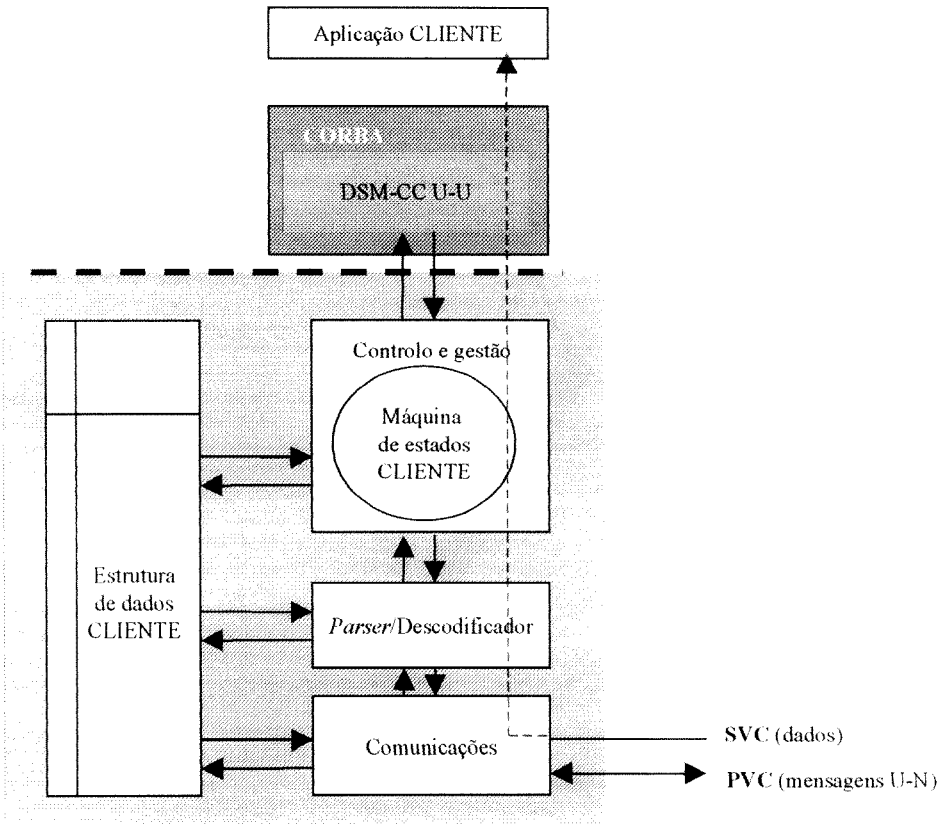


Figura 34 - Principais componentes *software* do cliente U-N.

De notar que os dados provenientes de um servidor, enquanto uma sessão decorre, passam transparentemente pelo código da componente U-N da aplicação. O bloco de comunicações apesar de gerir todos os aspectos relacionados com a rede, tratando-se da recepção de dados destinados a ser processados a um nível superior, tem de possuir a capacidade de encaminhar incólume a informação recebida nestas condições.

6.4.2 Modelo arquitectónico do servidor

Os blocos constituintes do código a implementar, do lado do servidor, são em tudo semelhantes aos blocos básicos identificados como essenciais, do lado do cliente.

A complexidade adicional neste cenário advém da presença da entidade SRM. Como anteriormente foi concluído, foi tomada a decisão de implementar o SRM como um processo independente, presente em cada servidor. Esta solução distribuída justifica-se com a relativa simplicidade que comporta a gestão dos vários cenários cliente-servidor U-N, presentes no estúdio ATLANTIC. Numa primeira fase, a troca de informação entre SRM e servidor poderá dispensar a sintaxe e a semântica DSM-CC U-N, mas o caminho a seguir contempla a sua implementação. No entanto, a troca de informação entre estas duas entidades pode legitimamente usufruir de mecanismos de comunicação privilegiados, disponíveis na máquina onde fisicamente se encontram, sem comprometer os requisitos de compatibilidade.

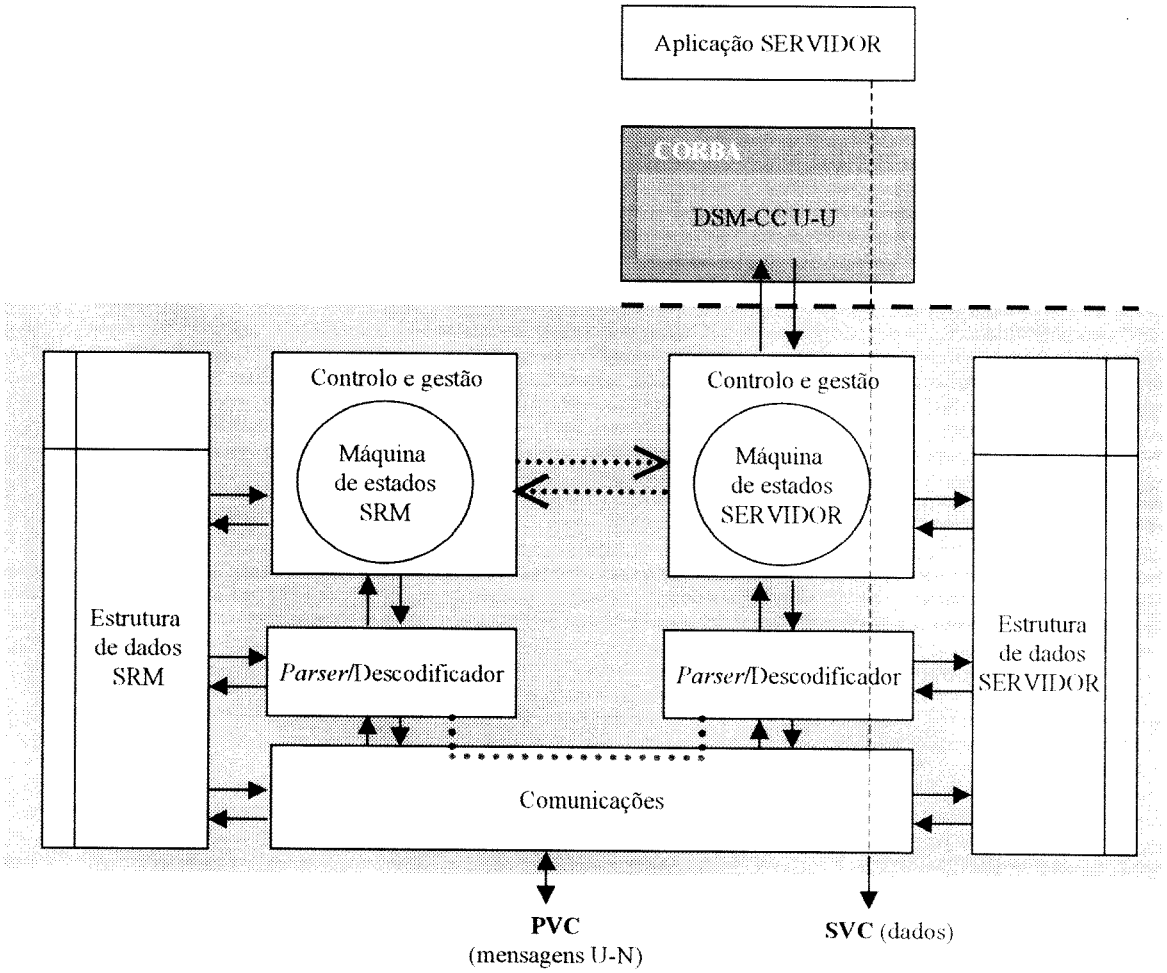


Figura 35 - Principais componentes *software* do servidor e do SRM U-N.

Assim numa primeira fase, a funcionalidade da entidade lógica SRM, definida em [5], pode mesmo ser totalmente absorvida pelo servidor. A evolução poderá conduzir, posteriormente, à constituição de um processo autónomo responsável pela implementação do SRM. Todavia, com base no facto de serem executados fisicamente na mesma máquina, a

gestão de ligações e a troca de mensagens, entre servidor e SRM, pode disfrutar de simplificações acrescidas.

6.5 Conclusão

O desenvolvimento do protocolo U-N especificado foi descrito neste capítulo. O facto de inter-operar com o CORBA obrigou à análise desta especificação, nomeadamente nos aspectos relacionados com o protocolo IIOP.

A identificação das múltiplas interfaces que o protocolo U-N tem de assegurar - U-N/AAL5, U-N/Q.2931 e U-N/DSM-CC U-U (CORBA) – constitui o primeiro passo do processo de desenvolvimento. Sucedeu-se então a concepção do esqueleto de comunicações que implementa a funcionalidade mínima de comunicações exigida pelo protocolo U-N.

Finalmente, foram construídos os modelos arquitectónicos do cliente e do servidor, na base do trabalho de desnvolvimento do protocolo U-N.

7 Conclusões

Neste capítulo são apresentadas as conclusões deste trabalho. São igualmente discutidas as perspectivas de trabalho futuro, em particular o que se espera concretizar com a integração e demonstração dos componentes que constituem a rede ATLANTIC.

Devido aos requisitos de comunicação impostos pelo estúdio ATLANTIC, a rede destinada a interligar os elementos constitutivos do estúdio baseia-se na tecnologia ATM, capaz de proporcionar o desempenho pretendido. A selecção do protocolo TCP para a camada de transporte assegura a fiabilidade que os fluxos MPEG-2 exigem. Para o comando e controlo do transporte de fluxos MPEG-2 foi adoptado o DSM-CC. A componente U-N do DSM-CC assegura a gestão e controlo dos recursos da rede. As semelhanças flagrantes, ao nível dos requisitos de controlo, de alguns cenários presentes no estúdio ATLANTIC como o serviço de vídeo a pedido, que prevê a utilização do DSM-CC, corroboram as opções tomadas ao nível do comando e controlo.

Uma vez definida a pilha protocolar a suportar, foi especificada a arquitectura de comunicações do estúdio ATLANTIC. Foi tomado um conjunto de decisões relativamente a cada protocolo, resultante da análise dos requisitos do estúdio ATLANTIC e da consideração das características de cada protocolo. A interacção entre os protocolos foi igualmente ponderada.

O ambiente Linux possui um conjunto de ferramentas ATM que como se viu são de grande utilidade aos propósitos de desenvolvimento no estúdio ATLANTIC. Numa fase intermédia o desenvolvimento assentou numa solução exclusivamente ATM. A API revelou-se neste contexto de grande utilidade no estabelecimento de PVCs e de SVCs. A necessidade de estabelecer ligações SVC recorreu ao *daemon* de sinalização disponibilizado. Na fase final do trabalho o mecanismo AREQUIPA foi explorado no sentido de garantir qualidade de serviço às ligações TCP/IP sobre ATM.

A especificação DSM-CC serviu de inspiração ao protocolo de controlo U-N a desenvolver no estúdio ATLANTIC. A componente U-N do DSM-CC mereceu assim, no âmbito deste trabalho, especial relevo pois serviu de ponto de partida ao trabalho aqui apresentado. A especificação do protocolo U-N constituiu uma base sólida sobre a qual assentou o trabalho de desenvolvimento. Sempre que possível tentou-se ir ao encontro da especificação DSM-CC, ficando salvaguardada uma eventual necessidade evolutiva (em

resultado por exemplo de um nível de exigência acrescido da parte do estúdio ATLANTIC) que se queira forçosamente compatível com o promissor DSM-CC.

O primeiro passo no desenvolvimento do protocolo U-N consistiu na identificação das múltiplas interfaces a assegurar - U-N/AAL5, U-N/Q.2931 e U-N/DSM-CC U-U (CORBA). Sucedeu-se então a concepção do esqueleto de comunicações que implementa a funcionalidade mínima de comunicações exigida pelo protocolo U-N.

A identificação dos principais blocos funcionais no cliente, no servidor e no SRM, bem como a análise do conjunto de interacções entre os diversos blocos, foram ambas sistematizadas nos modelos arquitectónicos, apresentados para o cliente e para o servidor/SRM, seguidos pelo trabalho de desenvolvimento em *software*.

O processo de desenvolvimento decorreu em duas etapas: a obtenção de uma solução cujas ligações de dados U-U recorrem apenas ao AAL5 sobre ATM e a solução final contemplando já o uso do protocolo TCP. Esta última obrigou à utilização do mecanismo AREQUIPA, para que a aplicação pudesse usufruir de forma transparente das vantagens da tecnologia ATM.

O cenário ATLANTIC pensado para ensaiar as primeiras iterações do protótipo incluiu a *Journalist Workstation* e o *Edit Browse Server*. O conjunto de requisitos e interacções entre ambas as entidades é em tudo semelhante ao serviço de vídeo a pedido, tendo sido por isso considerado como o cenário ideal de exploração.

No entanto, o primeiro ensaio de integração foi efectuado entre o *Format Converter* e o *Edit Source Server*. Tendo sido identificada a necessidade que o conversor de formatos tem de enviar dados para o servidor, desenvolveu-se de forma concertada as componentes necessárias, orientadas numa primeira fase para assegurar a funcionalidade mínima neste cenário, de forma a esboçar uma primeira iteração das aplicações cliente e servidor DSM-CC. Para além disso, a integração passou pela utilização do *device driver* desenvolvido para suporte da carta ATM da Fore. Todavia, o trabalho de optimização do desempenho do TCP sobre ATM, nomeadamente o estudo da inibição do *slow start*, pode apenas ser integrado posteriormente.

As soluções protótipo alcançadas exigem no futuro imediato o teste e eventuais optimizações que se revelem oportunas. Estas têm que ver não só com o protocolo desenvolvido em si mas também com o contexto de integração. A consideração de todas as componentes envolvidas e a análise do conjunto de interacções subsequentes tem que possuir um papel activo no processo de optimização.

Uma vez integrados, espera-se que os diversos componentes permitam a uma aplicação cliente invocar serviços a uma aplicação servidor. O servidor ao fornecer o serviço terá de ser capaz de localizar os dados a transferir, se de tal se tratar, e estabelecer e negociar a ligação de dados de acordo com os requisitos da transferência a efectuar. A aplicação cliente poderá então invocar um novo serviço. A forma como os recursos da rede são geridos pelo protocolo U-N é totalmente transparente para o nível de aplicação.

A extrapolação da funcionalidade obtida neste cenário particular terá posteriormente de concretizar-se para os outros cenários presentes no estúdio que envolvam igualmente a transferência de fluxos MPEG-2.

Bibliografia

[1] The ATM Forum Technical Committee, "Video on Demand Specification 1.0", Janeiro de 1996.

[2] The ATM Forum Technical Committee, "Video on Demand Specification 1.1", Março de 1997.

[3] ISO/IEC, "Generic Coding of Moving Pictures and Associated Audio: Systems", (MPEG-2 Systems Specification), ISO/IEC 13818-1, Novembro de 1994.

[4] P. A. Sarginson, "A Tutorial Introduction to the Systems Layer", BBC Research and Development Department.

[5] ISO/IEC, "Information Technology – Generic Coding of Moving Pictures and Associated Audio: Digital Storage Media Command and Control ISO/IEC 13818-6 Draft International Standard", ISO/IEC JTC1/SC29/WG11 MPEG95/N1100, Novembro de 1995.

[6] I. Vila Verde, "Linux Device Driver – Development Status", ATLANTIC Deliverable, Maio de 1997.

[7] P. Inácio, I. Vila Verde, J. Ruela, "Linux Device Driver – Development Status, version 2.0", ATLANTIC Deliverable, Julho de 1997.

[8] W. Almesberger, "Linux ATM device driver interface Draft, version 0.1", EPFL, Fevereiro de 1996, ftp://lrcftp.epfl.ch/pub/linux/atm/docs/.

[9] The ATM Forum Technical Committee, "ATM User-Network Interface (UNI) Signalling Specification, version 4.0", Julho de 1996.

[10] ITU, "ITU-T Recommendation Q.2931, Broadband Integrated Services Digital Network (B-ISDN) – Digital Subscriber Signalling System no. 2 (DSS2) – User-Network Interface (UNI) – Layer 3 Specification for Basic Call/Connection Control", Fevereiro de 1995

[11] ITU, "ITU-T Recommendation Q.2971, B-ISDN DSS2 UNI Layer 3 Specification for Point-to-Multipoint Call/Connection Control", 1996.

[12] ITU, "ITU-T Recommendation Q.2100, B-ISDN Signalling ATM Adaptation Layer (SAAL) Overview Description", Julho de 1994.

[13] ITU, "ITU-T Recommendation Q.2110, B-ISDN SAAL Service Specific Connection Oriented Protocol (SSCOP)", Julho de 1994.

[14] ITU, "ITU-T Recommendation Q.2130, B-ISDN SAAL Service Specific Coordination Function (SSCF) at the UNI, Julho de 1994.

[15] The ATM Forum Technical Committee, "Traffic Management Specification version 4.0", Abril de 1996.

[16] ITU, "ITU-T Recommendation I.362, B-ISDN ATM Adaptation Layer (AAL) Functional Description", 1993.

[17] ITU, "ITU-T Recommendation I.363, B-ISDN ATM Adaptation Layer (AAL) Specification", Março de 1993.

[18] ITU, "ITU-T Recommendation I.361, B-ISDN ATM Layer Specification", 1993.

[19] R. Atkinson, "Default IP MTU for use over ATM AAL5", RFC 1626, Maio de 1994.

[20] A. Romanow, S. Floyd, "Dynamics of TCP Traffic over ATM Networks", ACM SIGCOMM Computer Communications Review, Outubro de 1994, ftp://ftp.ee.lbl.gov/papers/tcp_atm.ps.Z.

[21] W Stevens, "TCP/IP Illustrated", volumes 1 e 2, Addison-Wesley, 1994.

[22] R. Fox, "TCP Big Window and NAK Options", RFC 1106, Junho de 1989.

[23] W. Stevens, "TCP Slow Start, Congestion Avoidance, Fast Retransmit and Fast Recovery", RFC 2001, Janeiro de 1997.

[24] V. Jacobson, M. J. Karels, "Congestion Avoidance and Control", SIGCOMM '88, 1988.

[25] V. Jacobson, R. Braden, D. Borman, "TCP Extensions for High Performance", RFC 1323, Maio de 1992.

[26] G. Mindem, V. Frost, J. Evans, B. Ewy, "TCP/ATM Experiences in the MAGIC Testbed", Telecommunications & Information Sciences Laboratory, University of Kansas, ftp://ftp.tisl.ukans.edu/pub/projects/MAGIC/TCP-ATM-Perform.ps.

[27] H. T. Kung, R. Morris, "Detailed Behavior Of TCP over ATM", GLOBECOM '95, 1995.

[28] H. T. Kung, R. Morris, "Impact of ATM Switching and Flow Control on TCP Performance: Measurements on an Experimental Switch", GLOBECOM '95, 1995.

[29] D. Clark, D. Tennehouse, " Architectural Considerations for a new Generation of Protocols", SIGCOMM '90, Setembro de 1990.

[30] R. Cole, D. Shur, C. Villamizar, "IP over ATM: A Framework Document", RFC 1932, Abril de 1996.

[31] M. Laubach, "Classical IP and ARP over ATM", RFC 1577, Janeiro de 1994.

[32] J. Heianen, "Multiprotocol Encapsulation over ATM Adaptation Layer 5", RFC 1483, Julho de 1993.

[33] M. Perez, F. Liaw, A. Mankin, E. Hoffman, D. Grossman, A. Malis, "ATM Signalling Support for IP over ATM", RFC 1755, Fevereiro de 1995.

[34] W. Almesberger, "Linux ATM API", EPFL, Julho de 1996, ftp://lrcftp.epfl.ch/pub/linux/atm/api/.

[35] The ATM Forum Technical Committee, "LAN Emulation Over ATM version 1.0", Janeiro de 1995.

[36] D. Katz, D. Piscitello, J. V. Luciani, "NBMA Next Hop Resolution Protocol (NHRP) (work in progress)", Internet Draft, Dezembro de 1995, http://info.internet.isi.edu/in-drafts/files/draft-ietf-rolc-nhrp-11.txt.

[37] R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin, "Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification", RFC 2205, Setembro de 1997.

[38] M. Borden, E. Crawley, B. Davie, S. Batsel, "Integration of Real-time Services in an IP-ATM Network Architecture", RFC 1821, Agosto de 1995.

[39] W. Almesberger, "ATM on Linux", EPFL, Março de 1996, ftp://lrcftp.epfl.ch/pub/linux/atm/papers/atm_on_linux.ps.gz.

[40] W. Almesberger, "ATM on Linux – The 3rd year", EPFL, Março de 1997, ftp://lrcftp.epfl.ch/pub/linux/atm/papers/atm_3rd.ps.gz.

[41] W. Almesberger, J. Le Boudec, P. Oechslin, "Application REQuested IP over ATM (AREQUIPA)", RFC 2170, Julho de 1997.

[42] Fore Systems Inc., "ForeRunner ATM Switch Configuration Manual", Março de 1996.

[43] W. Almesberger, "Linux ATM internal signalling protocol, version 0.2", EPFL, Novembro de 1996, ftp://lrcftp.epfl.ch/pub/linux/atm/docs/.

[44] The ATM Forum Technical Committee, "Integrated Local Management Interface (ILMI), version 4.0", Setembro de 1996.

[45] W. Almesberger, "AREQUIPA: Design and Implementation", Technical Report 96/213, EPFL, November 1996, ftp://lrcftp.epfl.ch/pub/arequipa/aq_di-1.tar.gz.

[46] ATM Forum SAA API Working Group, "AREQUIPA Overview", Outubro de 1996.

[47] Object Management Group, "The Common Object Request Broker: Architecture and Specification", revisão 2.0, Julho de 1995, actualizada em Julho de 1996.

[48] ITU-T SG11 WP2 TDR106Rev2, "Session and Resource Correlation Identification Capability of B-ISDN Signalling Protocols", Liaison to SG15 & ISO/IEC JTC1 SC29/WG11 from ITU-T SG11, Contact: C. Kawa, Northern Telecom Inc., October 1995

[49] C. Silva, J. Ruela, "Control of ATLANTIC network resources with DSM-CC", ATLANTIC Deliverable, Abril de 1997.

[50] V. Balabanian, L. Casey, N. Greene, "Digital Storage of Media-Command and Control Protocol Applied to ATM", IEEE JSAC, vol. 14, nº 6, Agosto de 1996.

[51] A. Jain, W. Fischer, P. Sibille, "An Evolvable ATM-Based Video Network Design Supporting Multiple Access Network Technologies", IEEE Communications Magazine, Novembro de 1995.

[52] S. Vinoski, "Distributed Object Computing with CORBA", Hewllet-Packard Company, Agosto de 1993.

[53] A. Gokhale, D. Schmidt, "Measuring the Performance of Communication Middleware on High-Speed Networks", SIGCOMM'96, Agosto de 1996.

[54] C. Silva, J. Ruela, "DSM-CC User to Network interfaces", ATLANTIC Deliverable, Julho de 1997.

[55] Digital Audio-Visual Council, DAVIC 1.0 Specifications," Dezembro de 1995.

[56] The ATM Forum Technical Committee, "ATM User-Network Interface (UNI) Signalling Specification, version 3.0", 1993.

[57] The ATM Forum Technical Committee, "ATM User-Network Interface (UNI) Signalling Specification, version 3.1", Setembro de 1994.

ANEXO A – Formato das mensagens U-N e os seus campos de informação

Todas as mensagens, entre a rede e o cliente ou o servidor, obedecem a um formato comum. Cada mensagem é construída à custa da concatenação de um cabeçalho normalizado e dados. O cabeçalho contém informação relativa ao tipo de mensagem e eventualmente dados de adaptação que porventura sejam requeridos pelo mecanismo de transporte.

UserNetworkMessagePacket()
DsmccMessageHeader()
MessagePayload()

Formato genérico de uma mensagem DSM-CC U-N

DsmccMessageHeader()		**nº octetos**
ProtocolDiscriminator	ISO ou ITU	**1**
DsmccType	0x02 (significando tratar-se de uma mensagem U-N)	**1**
MessageId	Código da mensagem	**2**
TransactionId	Único por um período de tempo; significado local; 2 bits (indicação do gerador da mensagem) + 30 bits (número da transacção)	**4**
Reserved	0	**1**
AdaptationLength	Comprimento do cabeçalho de adaptação	**1**
MessageLength	Comprimento da mensagem (contado a partir do fim do cabeçalho)	**2**
DsmccAdaptationHeader()		

Formato do cabeçalho de uma mensagem DSM-CC MPEG-2

DsmccUNResourceDescriptor()		nº octetos
ResourceRequestId	Estabelecido pelo servidor nesta especificação; permite correlacionar o recurso especificado na mensagem de pedido com o resultado subsequente na mensagem de confirmação	**2**
ResourceDescriptorType	Define o recurso a ser pedido	**2**
ResourceNum	10 bits (resourceNumberAssignor = Servidor) + 30 bits (resourceNumber)	**2**
AssociationTag	Identifica os recursos que fazem parte de uma ligação U-U; significado extremo a extremo	**2**
ResourceAllocator/Attribute/View		**1**
ResourceStatus	Define o estado do recurso pedido;	**2**
ResourceLength	Tamanho do vector que descreve um recurso, [resourceDataValueByte]	**2**
ResourceDataFieldCount	Número de campos de dados que definem tipo de recurso usado	**2**
DsmccResourceDescriptorValue()		
ResourceDataValueByte		**1**

Descritor de recurso DSM-CC U-N.

Nome do campo	Comprimento (octetos)	Intervalo	Descrição
ClientId	20	NSAP OSI	O cliente deve ser identificado por um endereço NSAP global e único da OSI (ou um valor que possa ser convertido pela rede num endereço especifico); pode ser um endereço IP
DeviceId	6	0x00000000000 - 0x3fffffffffff	Um terminal U-N tem um identificador único e global na rede; pode ser um endereço MAC
Reason	2	0x0000 - 0xffff	Identifica a razão de uma falha
Response	2	0x0000 - 0xffff	Indica resposta a um pedido
ResourceCount	2	0x0000 - 0xffff	Número de descritores de recursos
ResourceDescriptor()	Variable	Dependente do descritor	Consiste no cabeçalho de um descritor de recurso e na lista correspondente de campos de dados
ServerId	20	NSAP OSI	O servidor deve ser identificado por um endereço NSAP global e único da OSI (ou um valor que possa ser convertido pela rede num endereço especifico); pode ser um endereço IP
SessionId	10		6 octetos (*deviceId*) + 4 octetos (*sessionNum*)
SessionNum	4	0x00000000 – 0xffffffff	Um número que identifique globalmente a sessão no terminal que o estabelece
UserData()	Variavél	Variável	Contém uuData e estruturas privadas de dados

Tabela V: Campos de dados relevantes para a especificação do protocolo U-N para o ATLANTIC

ANEXO B – Mensagens especificadas no protocolo U-N ATLANTIC

Estabelecimento de sessão

0x4010	ClientSessionSetupRequest	nº octetos
SessionId	Gerado pelo cliente	**10**
ClientId	Estabelecido pelo cliente	**20**
ServerId	Estabelecido pelo cliente	**20**
UserData()	Definido pela parte U-U (uuData)	

Mensagem 1: U-N ClientSessionSetupRequest

0x4011	ClientSessionSetupConfirm	nº octetos
SessionId	Estabelecido pelo SRM para o valor que foi recebido no pedido de sessão correspondente	**10**
ServerId	Estabelecido pelo SRM para o valor que foi recebido na mensagem ServerSessionSetupResponse	**20**
Response	Estabelecido pelo SRM para indicar o estado de um pedido de sessão (ex.: o valor *rspOK* significa serviço estabelecido)	**20**
ResourceCount	Estabelecido pelo servidor; número de recursos locado a um cliente para a sessão correspondente	**2**
ResourceDescriptor()	Estabelecido pelo servidor	
UserData()	Definido pela parte U-U (uuData)	

Mensagem 2: U-N ClientSessionSetupConfirm

0x8012	ServerSessionSetupIndication	nº octetos
SessionId	Estabelecido pelo SRM para o valor que foi recebido na mensagem ClientSessionSetupRequest	**10**
ClientId	Estabelecido pelo SRM para o valor que foi recebido na mensagem ClientSessionSetupRequest	**20**
serverId	Estabelecido pelo SRM para o valor que foi recebido na mensagem ClientSessionSetupRequest	**20**
ForwardCount	0; número de reencaminhamentos de sessão	**2**
ForwardServerId	Sem significado nesta especificação; lista de serverIds dos servidores que reencaminhariam a sessão	**20**
UserData()	Definido pela parte U-U (uuData)	

Mensagem 3: U-N ServerSessionSetupIndication

0x8013	ServerSessionSetupResponse	nº octetos
SessionId	Estabelecido pelo SRM para o valor que foi recebido na mensagem ServerSessionSetupIndication	**10**
ServerId	Estabelecido pelo SRM para o valor que foi recebido na mensagem ClientSessionSetupRequest	**20**
Response	Estabelecido pelo servidor; indica a resposta do servidor à mensagem ServerSessionSetupIndication	**2**
NextServerId	Sem significado nesta especificação; serverId do servidor para o qual a sessão seria reencaminhada	**20**
ResourceCount	Estabelecido pelo servidor; numero de recursos atribuídos à sessão	**2**
ResourceDescriptor()		
UserData()	Definido pela parte U-U (uuData)	

Mensagem 4: DSM-CC U-N ServerSessionSetupResponse

Terminação de uma sessão

0x4020 ClientReleaseRequest		nº octetos
SessionId	Estabelecido pelo cliente para indicar a sessão que deseja terminar	10
Reason	Estabelecido pelo cliente para indicar a razão pela qual deseja terminar a sessão	2
UserData()	Definido pela parte U-U (uuData)	

Mensagem 5: U-N ClientReleaseRequest

0x4021 ClientReleaseConfirm		nº octetos
SessionId	Estabelecido pelo SRM; valor de sessionId recebido na mensagem ClientReleaseRequest	10
Response	Estabelecido pelo SRM para indicar o resultado do pedido de terminação da sessão	2
UserData()	Definido pela parte U-U (uuData)	

Mensagem 6: U-N ClientReleaseConfirm

0x8020 ServerReleaseIndication		nº octetos
SessionId	Estabelecido pelo SRM; sessionId da sessão a terminar	10
Reason	Estabelecido pelo SRM; o mesmo valor que o do campo homólogo da mensagem ClientReleaseRequest	
UserData()	Definido pela parte U-U (uuData)	

Mensagem 7: U-N ServerReleaseIndication

0x8021 ServerReleaseResponse		nº octetos
SessionId	Estabelecido pelo servidor; valor de sessionId recebido na mensagem ServerReleaseIndication	10
Response	Estabelecido pelo servidor; indica a resposta do servidor à mensagem ServerReleaseIndication	2
UserData()	Definido pela parte U-U (uuData)	

Mensagem 8: DSM-CC U-N ServerReleaseResponse

Anexo C – Código fonte do protocolo U-N do ATLANTIC

Esqueleto de comunicações - CLIENTE

```
/*DSM-CC CLIENT*/
/*PVC CLIENT - SVC SERVER*/

#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <errno.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <sys/socket.h>
#include <linux/atm.h>
#include <atm.h>

#define BSIZE 1024

static void usage(const char *name)
{
    fprintf(stderr,"usage: %s [itf.]vpi.vci [file name] \n",name);
    exit(1);
}

int receive_svc_data(char* filename)
{
        struct sockaddr_atmsvc      addr, peer_addr;
        struct                  atm_qos qos;
        int                     s, peer_addr_len, n;
        char                    local_ATM_address[50];
        size_t                  namelen;

/*CONNECTION PREPARATION*/
        /*Socket creation*/
        if ((s = socket (PF_ATMSVC, SOCK_DGRAM, ATM_AAL5)) < 0)
        {
                perror("socket");
                exit(1);
        }
        else
        {
                printf("(socket) Created!\n");
        }
        /*QoS descriptor initialisation*/
        memset(&qos, 0, sizeof(qos));
        qos.txtp.traffic_class = ATM_CBR;     //obsolete
        qos.txtp.min_pcr = ATM_MAX_PCR;       //obsolete
        qos.txtp.max_sdu = 8192;              //obsolete
        qos.rxtp = qos.txtp;                  //obsolete
        /*Traffic parameter setting*/
        if (setsockopt(s, SOL_ATM, SO_ATMQOS, &qos, sizeof(qos)) < 0)
```

```
    {
            perror("setsockopt(SO_ATMQOS)");
            return -1;
    }
    else
    {
            printf("setsockopt succeeded\n");
    }

/*Get local host name - IP name must be equal to the ATM name*/
    namelen = 50;
    if (gethostname(local_ATM_address, namelen) == -1){
        if (errno == EINVAL)
            printf("Host name: %s\n", local_ATM_address);
        else{
            printf("Error on gethostname()\n");
            return -1;
        }
    }
    else
        printf("(2)Host name: %s\n", local_ATM_address);

    /*Address descriptor initialization*/
    memset(&addr, 0, sizeof(addr));
    if (text2atm ( (const char *) local_ATM_address,
                   (struct sockaddr*) &addr,
                   sizeof(addr),
                   T2A_SVC | T2A_WILDCARD | T2A_NAME) < 0)
    {
            printf("Error performing text2atm conversion!\n");
            return -1;
    }
    addr.sas_addr.blli = NULL;
    addr.sas_addr.bhli.hl_type = ATM_HL_NONE;
    /*associate address with end point*/
    if (bind(s, (struct sockaddr *) &addr, sizeof addr) == 0)
{
        printf("Bound!\n");
}
else
{
        printf("Error performing bind()!\n");
        return -1;
}

    /* make endpoint listen for service requests */
    if (listen (s, 5) == 0)
{
        printf("Listening...\n");
}
else
    {
            printf("Error performing listen()!\n");
            return -1;
    }

    /* performs the iterative server activities */
    for(;;)
    {
            char buf[BUFSIZ];
```

```
                int r_bytes, n_sd;
                int peer_addr_len = sizeof peer_addr;

            /*File variables*/
            int fp;

                /* create a new end point for communication */
                while ((n_sd = accept (s,
                                       (struct sockaddr *)&peer_addr,
                                       &peer_addr_len)) == -1
                        && errno == EINTR)
                   continue;
/*DATA EXCHANGE*/
            if ((fp = open(filename, O_RDWR | O_CREAT)) != -1){
                    while ((r_bytes = read (n_sd, buf, sizeof buf)) > 0)
                  write(fp, buf, r_bytes);
                close(fp);
            }
            else{
                while ((r_bytes = read (n_sd, buf, sizeof buf)) > 0)
                        write(1, buf, r_bytes);
            }
            close(n_sd);
        }
/*CONNECTION TEARDOWN*/
       /* close the new end point*/
        (void)close(s);
      return 0;
}

int send_pvc_message(char *vpivci, char filename[])
{
    struct sockaddr_atmpvc addr;
    struct atm_qos qos;
    int s,size;

    if ((s = socket(PF_ATMPVC,SOCK_DGRAM,0)) < 0) {
      perror("socket");
      return -1;
    }
    memset(&addr,0,sizeof(addr));
    if (text2atm(vpivci,(struct sockaddr *) &addr,sizeof(addr),T2A_PVC) <
0)
    {
      printf("Address not converted!\n");
      return -1;
    }
    memset(&qos,0,sizeof(qos));
    qos.aal = ATM_AAL5;
    qos.txtp.traffic_class = ATM_UBR;
    qos.txtp.max_sdu = BSIZE;
    if (setsockopt(s,SOL_ATM,SO_ATMQOS,&qos,sizeof(qos)) < 0) {
      perror("setsockopt SO_ATMQOS");
      return -1;
    }
    if (connect(s,(struct sockaddr *) &addr,sizeof(addr)) < 0) {
      perror("connect");
      return -1;
    }
    if (size = write(s, filename, strlen(filename)) < 0)
```

```
    {
      printf("Error performing write()!\n");
      return -1;
    }
    else
    {
      printf("%d\n",strlen(filename));
      if (size < 0) printf(" (%s)\n",strerror(errno));
      return s;
    }
}
int main(int argc,char **argv)
{
    int s_pvc;
    char filename[50];

    if (argc < 2) usage(argv[0]);
    if (argc == 3)
      strcpy(filename, argv[2]);
    else
      strcpy(filename, "svctestfile.txt");
    if (s_pvc = send_pvc_message(argv[1], filename) < 0)
    {
      printf("Error while handling PVC client!\n");
    }
    else
    {
      if ( receive_svc_data(filename) < 0)
      {
          printf("Error while handling SVC server!\n");
      }
      else
      {
            printf("Success on SVC!\n");
      }
    }
    close(s_pvc);
    return 0;
}
```

Esqueleto de comunicações – SERVIDOR

```
/*DSM-CC SERVER*/
/*PVC SERVER - SVC CLIENT*/
#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <errno.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <linux/atm.h>
#include <linux/atmsap.h>  /*blli and bhli*/
#include <linux/socket.h> /*SOL_SOCKET*/
#include <atm.h>

#define BSIZE 1024
#define MAXLINE 512
```

```
static void usage(const char *name)
{
    fprintf(stderr,"usage: %s [itf.]vpi.vci peerhostname\n",name);
    exit(1);
}

/*
 *read the contents of the FILE *fp, write each line to
 *the stream socket
 *
 *Return to the caller when an EOF is encountered on the input file
*/
int write_file(char *filename, int sockfd)
{
    int         n;
    char        sendline[MAXLINE];
    FILE*       fp;

    if ((fp = fopen(filename, "r")) != NULL)
    {
        while ((fgets(sendline, MAXLINE, fp)) != NULL)
        {
            n = strlen(sendline);
            if (write(sockfd, sendline, n) != n)
            {
                printf("Error on write()!\n");
                fclose(fp);
                return -1;
            }
        }
        if (ferror(fp))
        {
            fclose(fp);
            return -1;
        }
        else
        {
            fclose(fp);
            return 0;
        }
    }
    else
        return -1;
}

/*
 *An SVC client application which writes
 *a file through the esablished SVC, using
 *the write_file() function
 */
int send_svc_data(char *filename, char *peer_ATM_address)
{
        struct sockaddr_atmsvc      addr, peer_addr;
        struct                  atm_qos qos;
        int                     s, i, peer_addr_len, n;
        ssize_t                   size;
      size_t                    namelen;
/*    char                peer_ATM_address[]="skyrack";*/
      char                local_ATM_address[50];
```

```
/*CONNECTION PREPARATION*/
        /*Socket creation*/
        if ((s = socket (PF_ATMSVC, SOCK_DGRAM, ATM_AAL5)) < 0)
        {
                perror("socket");
            return -1;
        }
        else
        {
                printf("(socket) Created!\n");
        }
        /*QoS descriptor initialisation*/
        memset(&qos, 0, sizeof(qos));
        qos.txtp.traffic_class = ATM_CBR;     //obsolete
        qos.txtp.min_pcr = ATM_MAX_PCR;       //obsolete
        qos.txtp.max_sdu = 8192;              //obsolete
        qos.rxtp = qos.txtp;                  //obsolete
        /*Traffic parameter setting*/
        if (setsockopt(s, SOL_ATM, SO_ATMQOS, &qos, sizeof(qos)) < 0)
        {
                perror("setsockopt(SO_ATMQOS)");
                return -1;
        }

      /*Get local host name - IP name must be equal to the ATM name*/
      namelen = 50;
        if (gethostname(local_ATM_address, namelen) == -1){
          if (errno == EINVAL)
            printf("Host name: %s\n", local_ATM_address);
          else{
                  printf("Error on gethostname()\n");
            return -1;
          }
      }
      else
          printf("(2)Host name: %s\n", local_ATM_address);

        /*Address descriptor initialization*/
        memset(&addr, 0, sizeof(addr));
        if (text2atm ( (const char*) local_ATM_address,
                       (struct sockaddr*) &addr,
                       sizeof(addr),
                         T2A_SVC | T2A_WILDCARD | T2A_NAME) < 0)
        {
                printf("Error performing text2atm conversion!\n");
                return -1;
        }
        else
        {
                printf("local ATM address converted!\n");
        }
        addr.sas_addr.blli = NULL;
        addr.sas_addr.bhli.hl_type = ATM_HL_NONE;

/*CONNECTION SETUP*/
        /*Addressing of the remote side (assignements)*/
        memset(&peer_addr, 0, sizeof(peer_addr));
        if (text2atm ( (const char*) peer_ATM_address,
                       (struct sockaddr*) &peer_addr,
                       sizeof(peer_addr),
```

```
                        T2A_SVC | T2A_NAME) < 0)
        {
                printf("Error performing peer text2atm conversion!\n");
                return -1;
        }
        else
      {
                printf("peer ATM address converted!\n");
        }
        peer_addr.sas_addr.blli = NULL;
        peer_addr.sas_addr.bhli.hl_type = ATM_HL_NONE;
        /*Connection setup*/
        if (connect (s,
                     (struct sockaddr *) &peer_addr,
                     sizeof(peer_addr)) < 0)
      {
                if (errno == EINPROGRESS)
                {
                        printf ("Connecting...\n");
                }
                else
                {
                        perror("connect(1)");
                        return -1;
                }
        }

/*DATA EXCHANGE*/
      if ((write_file(filename, s)) == 0)
          printf("File %s successfully written!\n", filename);
      else
          printf("Error handling file or file does not exist!\n");
/*CONNECTION TEARDOWN*/
        (void)close(s);
      return 0;
}

/*
 *Is a PVC Server apllication, which upon
 *receipt of a message over the PVC (a file name),
 *tries to open an SVC, to send the file
 *
 */
int receive_pvc_message(char *vpivci, char *peer_name)
{
    struct sockaddr_atmpvc addr;
    struct atm_qos qos;
    int s;

/*CONNECTION PREPARATION*/
    /*socket creation*/
    if ((s = socket(PF_ATMPVC,SOCK_DGRAM,0)) < 0) {
      perror("socket");
      return -1;
    }
    /*address descriptor initialisation*/
    memset(&addr,0,sizeof(addr));
    if (text2atm(vpivci,(struct sockaddr *) &addr,sizeof(addr),
      T2A_PVC | T2A_UNSPEC | T2A_WILDCARD) < 0)
    {
```

```
      printf("Address not converted!\n");
      return -1;
    }
    /*QOS descriptor initialisation*/
    memset(&qos,0,sizeof(qos));
    qos.aal = ATM_AAL5;
    qos.rxtp.traffic_class = ATM_UBR;
    qos.rxtp.max_sdu = BSIZE;
    /*traffic parameter setting*/
    if (setsockopt(s,SOL_ATM,SO_ATMQOS,&qos,sizeof(qos)) < 0) {
      perror("setsockopt SO_ATMQOS");
      return -1;
    }
/*CONNECTION SETUP*/
    /*full setup*/
    if (bind(s,(struct sockaddr *) &addr,sizeof(addr)) < 0) {
      perror("bind");
      return -1;
    }
/*CONNECTION STATUS*/


/*DATA EXCHANGE*/
    while (1) {
      char buffer[50];
      int size,i;

      for (i=0; i<50;i++) buffer[i]=0;
      if (size = read (s, buffer, BSIZE) == -1)
      {
            printf("Error while reading from PVC server socket!\n");
      }
      else
      {
            printf ("size = %d; buffer: %s\n", size, (char*)buffer);
            write (1, buffer, size);
            /*CALL SVC ROUTINE FROM HERE*/
            if (send_svc_data(buffer, peer_name) == -1)
            {
                printf("Error handling SVC client!\n");
                return -1;
            }
            else
            {
                printf("Success handling SVC!!!\n");
            }
      }
    }
    return s;
}
int main(int argc,char **argv)
{
    int s_pvc;

    if (argc != 3) usage(argv[0]);
    if (s_pvc = receive_pvc_message(argv[1], argv[2]) < 0)
    {
      printf("Error handling PVC server!\n");
    }
    else
```

```
    {
      printf("Success handling PVC server!\n");
    }
    close(s_pvc);
    exit(0);
}
```