Desenvolvendo Agentes Inteligentes para a Gerência Pró-Ativa de Redes ATM

Este exemplar corresponde à redação final da Dissertação devidamente corrigida e defendida por Adriano Sá Nascimento e aprovada pela Banca Examinadora.

Fortaleza, 23 de março de 1999.

Prof. Dr. Mauro Oliveira
Centro Federal de Educação Tecnológica - CEFET-CE (Orientador)

Dissertação apresentada ao Mestrado em Ciência da Computação, UFC, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

Mestrado em Ciência da Computação
Universidade Federal do Ceará

Desenvolvendo Agentes Inteligentes para a Gerência Pró-Ativa de Redes ATM

Adriano Sá Nascimento[1]

Março de 1999

Banca Examinadora:

- Prof. Dr. Mauro Oliveira
  Centro Federal de Educação Tecnológica - CEFET-CE (Orientador)
- Prof. Dr. José Neuman de Souza
  Universidade Federal do Ceará - UFC
- Prof. Dr. Paulo Roberto Freire Cunha
  Universidade Federal de Pernambuco - UFPE

[1]Financiado pela CAPES

Conteúdo

Introdução **1**

I Conceitos Básicos 4

1 Gerência de Redes **5**
1.1 Introdução 5
1.2 Conceitos Básicos 6
1.2.1 Objeto Gerenciado e MIB 6
1.2.2 Paradigma Gerente-Agente 7
1.2.3 SMI 7
1.3 Gerência OSI 9
1.4 Gerência Internet 10
1.4.1 Introdução 10
1.4.2 SNMPv1 11
1.4.3 SNMPv2 14
1.4.4 RMON 16
1.4.5 SNMPv3 17

2 Agentes Inteligentes **18**
2.1 Introdução 18
2.2 Características 19
2.3 Modelo Conceitual 20
2.4 Agentes Inteligentes na Gerência de Redes 22
2.4.1 Gerência Pró-Ativa 22
2.4.2 Agentes Móveis 23

3 Redes Neurais **25**
3.1 Introdução 25
3.2 Paradigmas de Aprendizado das Redes Neurais 27

3.2.1 Aprendizado Supervisionado 27
3.2.2 Aprendizado Não Supervisionado 27
3.2.3 Aprendizado por Reforço 28
3.3 Topologias de Redes Neurais 28
3.3.1 Redes Neurais Feedforward 28
3.3.2 Redes Neurais Parcialmente Recorrentes 29
3.3.3 Redes Neurais Totalmente Recorrentes 29
3.4 Modelos de Redes Neurais 30
3.5 Treinamento de Redes Neurais 30
3.6 Agentes Inteligentes e Redes Neurais 34

4 Redes ATM **35**
4.1 Introdução 35
4.2 RDSI-FL 36
4.3 Características Gerais de ATM 37
4.4 Interfaces ATM 39
4.5 Conexões ATM 41
4.6 Célula ATM 42
4.7 Arquitetura ATM 44
4.7.1 Camada Física 45
4.7.2 Camada ATM 47
4.7.3 Camada AAL 48
4.8 Endereçamento ATM 51
4.9 LAN Emulation 52
4.10 IP sobre ATM 55
4.11 Aplicações de Redes Neurais em ATM 56

5 Gerência de Redes ATM **57**
5.1 MIBs SNMP Aplicadas ao ATM 58
5.1.1 AToM MIB 59
5.1.2 MIBs SNMP e o Modelo das Cinco Interfaces 61
5.2 ILMI 62
5.2.1 ILMI MIB 63
5.2.2 Integração ILMI e AToM MIB 67
5.3 Funções OAM 68
5.3.1 Células OAM 69

II RENATA **73**

6 Arquitetura da RENATA **74**
6.1 Redes Neurais na Gerência ATM 74
6.2 Arquitetura Funcional 75
6.2.1 Módulo de Treinamento - Simulador ATM 76
6.2.2 Módulo de Treinamento - MSPD 76
6.2.3 Módulo de Treinamento - Simulador de Redes Neurais 77
6.2.4 Módulo Neural 77
6.2.5 Módulo de Gerência 78
6.3 Arquitetura Física 78
6.3.1 Produtos Utilizados 80
6.3.2 Ferramentas Implementadas 80
6.3.3 Utilizando RENATA 82

7 Descrição do Protótipo **85**
7.1 DocRN 87
7.1.1 Configuração do Critério de Aceitação 88
7.1.2 Configuração do Modo de Ativação 89
7.1.3 Configuração das Entradas da Rede Neural 89
7.1.4 Configuração das Saídas da Rede Neural 93
7.2 RenataSetup 94
7.2.1 Configuração da Gerência do Dispositivo 95
7.3 Módulo de Gerenciamento 96
7.3.1 Navegador de MIBs 97
7.4 Módulo de Prototipação 98
7.4.1 Configuração das Fontes das Entradas da Rede Neural 100
7.4.2 Configuração das Ações do Agente 101

8 Desenvolvendo Agentes RENATA **102**
8.1 Introdução 102
8.2 Características Gerais do Agente RENATA 102
8.3 Estrutura de Dados 103
8.3.1 Informações da Rede Neural 103
8.3.2 Informações dos Dispositivos 106
8.3.3 Informações de Prototipação do Agente 107
8.4 Funcionamento do Agente RENATA 108
8.4.1 ReceiverThread 109
8.4.2 ActionThread 110

8.4.3 PredictionThread 111
8.4.4 Obtenção e Preparação das Entradas 112
8.5 Algoritmo de Prototipação 115
8.6 MIB RENATA 117
8.7 Estudo de Caso 119

9 Conclusões **121**

Bibliografia **123**

III Apêndices **129**

A MIB RENATA **130**

B Estrutura de Dados da RENATA **148**
B.1 Informações da Rede Neural 148
B.2 Informações sobre Dispositivo 155
B.3 Informações de Prototipação do Agente 158
B.4 Estrutura de Dados do Agente RENATA 163

Lista de Tabelas

3.1 Modelos e Funções de Redes Neurais 31
3.2 Parâmetros de Treinamento de Redes Neurais 34

4.1 Interfaces da Camada Física 46
4.2 Classificação dos Serviços AAL 49

5.1 Mapeamento das MIBs para o Modelo das Cinco Interfaces 62
5.2 Campos da Célula OAM 70

6.1 Passos do Desenvolvimento de um Agente na RENATA 78
6.2 Principais Classes da API Java SNMP 82
6.3 Passos e Ferramentas do Desenvolvimento de um Agente na RENATA . . . 84

7.1 Técnicas de Normalização de Vetores 90

B.1 Classe NNInfo 150
B.2 Classe AcceptanceInfo 150
B.3 Classe ActivationInfo 151
B.4 Classe InputInfo 151
B.5 Classe ScaleInfo 152
B.6 Classe ScalePieceInfo 152
B.7 Classe CodeInfo 152
B.8 Classe ArrayInfo 153
B.9 Classe CompAttInfo 153
B.10 Classe OutputInfo 153
B.11 Classe OutRangeInfo 154
B.12 Classe OutRescaleInfo 154
B.13 Classe RenataInfo 156
B.14 Classe DeviceInfo 156
B.15 Classe MibInfo 156
B.16 Classe IlmiInfo 157

B.17 Classe DevAgentInfo 157
B.18 Classe AgentInfo 159
B.19 Classe InSourceInfo 160
B.20 Classe SourceInfo 160
B.21 Classe MibObjInfo 161
B.22 Classe AttSourceInfo 161
B.23 Classe OutActionInfo 161
B.24 Classe ActionInfo 162
B.25 Classe AgentMIB 164
B.26 Classe InputEntry 165
B.27 Classe OutputEntry 165
B.28 Classe ActionEntry 165
B.29 Classe OperationEntry 166
B.30 Classe MibObjInfo 166

Lista de Figuras

1.1 Modelo Gerente-Agente 7
1.2 MIB-II 12
1.3 Operações do SNMPv1 13
1.4 Mensagens SNMPv1 14
1.5 Mensagens SNMPv2 15
1.6 Exemplo de Configuração com RMON 16

2.1 Elementos de um Agente Inteligente 21
2.2 Modelo Conceitual de um Agente Inteligente 21

3.1 Estrutura de uma Unidade de Processamento 26
3.2 Redes Neurais Feedforward 29
3.3 Redes Neurais Parcialmente Recorrentes 29
3.4 Redes Neurais Totalmente Recorrentes 30
3.5 Processo de Treinamento de Redes Neurais 32

4.1 RDSI-FE 36
4.2 RDSI-FL 37
4.3 Integração de Serviços no ATM 38
4.4 Interfaces ATM 39
4.5 Identificadores de Conexão ATM 42
4.6 Comutação de Canais e Caminhos Virtuais 43
4.7 Estrutura da Célula ATM (UNI) 43
4.8 Modelo de Referência de Protocolos da RDSI-FL 44
4.9 Plano do Usuário e de Controle 45
4.10 Funções da Camada Física 47
4.11 Funções da Camada ATM 48
4.12 Estrutura da Camada AAL 50
4.13 Estrutura da SAAL 51
4.14 Estrutura da LANE 53
4.15 IP Sobre ATM 55

5.1 Modelo das 5 Interfaces 58
5.2 Estrutura da AToM MIB 59
5.3 Integrated Local Management Interface (ILMI) 63
5.4 Estrutura da ILMI MIB 64
5.5 Mecanismo de Registros de Endereços na ILMI 68
5.6 Integração ILMI - AToM MIB 69
5.7 Payload da Célula OAM 69
5.8 Mecanismo de Loopback das Células OAM 72

6.1 Arquitetura Funcional da RENATA 75
6.2 Arquitetura Física da RENATA 79
6.3 Protótipo da RENATA 83

7.1 Ferramentas Implementadas da RENATA 85
7.2 DocRN 86
7.3 DocRN - Configuração do Critério de Aceitação 87
7.4 DocRN - Configuração do Modo de Ativação 88
7.5 DocRN - Configuração das Entradas da Rede Neural 89
7.6 DocRN - Configuração da Preparação dos Dados - Atributo Computado . . 91
7.7 DocRN - Configuração da Preparação dos Dados - Escalonamento 92
7.8 DocRN - Configuração das Saídas da Rede Neural 93
7.9 RenataSetup 94
7.10 RenataSetup - Configuração da Gerência do Dispositivo 95
7.11 Módulo de Gerenciamento 96
7.12 Módulo de Gerenciamento - Navegador de MIBs 97
7.13 Módulo de Prototipação 98
7.14 Módulo de Prototipação - Configuração das Fontes das Entradas 99
7.15 Módulo de Prototipação - Configuração das Ações do Agente 100

8.1 MIB RENATA 118
8.2 Rede Neural - Estudo de Caso 119

Introdução

Esta década vem sendo notadamente marcada por mudanças intensas influenciadas pela Tecnologia da Informação; talvez, comparáveis àquelas proporcionadas pelo advento da imprensa [1]. O filósofo e político inglês Francis Bacon já afirmava no século XVI que "saber é poder". Contudo, qualquer que seja o futuro, ele passa pela necessidade de se democratizar verdadeiramente o acesso à informação, como condição de cidadania nessa nova "Aldeia Global" (conceito criado por Marshall McLuhan) [2].

Por trás desta revolução, estão as redes de computadores, que possibilitam o compartilhamento de informações e de recursos, independentemente de localização física. Nos últimos anos, as redes de computadores experimentaram grande popularização, tanto em ambientes locais e metropolitanos de empresas e universidades como em ambientes continentais, notadamente em função do crescimento exponencial da Internet, a rede mundial de computadores. A Internet guarda em si o potencial revolucionário necessário para provocar mudanças na ordem mundial.

Paralelamente, começa a se esboçar uma convergência entre a infra-estrutura de comunicação e a indústria da mídia, à medida que ambas se digitalizam. As aplicações multimídia têm se tornado cada vez mais comuns, sendo sua utilização em redes de computadores uma evolução natural. Porém, a atual Internet, que foi projetada para transporte de dados textuais, não suporta bem aplicações multimídia.

Estas aplicações em redes de computadores exigem maiores taxas de transferência, bem como requisitos mais acurados, tais como mínimo atraso e baixa taxa de perda nos meios utilizados. As tecnologias convencionais utilizadas na comunicação de dados têm se mostrado incapazes de garantir Qualidade de Serviço (QoS) ao integrar em uma mesma infra-estrutura serviços de voz, vídeo e dados textuais. O Modo de Transferência Assíncrono (ATM - *Asynchronous Transfer Mode*) tem se mostrado a tecnologia mais satisfatória, tendo sido escolhido pelo ITU-T (*International Telecommunications Union - Telecommunications*) como o suporte das Redes Digitais de Serviços Integrados de Faixa Larga (RDSI-FL) [3].

Devido à complexidade e flexibilidade inerentes ao ATM, um sistema eficiente que permita o funcionamento correto da rede torna-se indispensável. Os sistemas de gerência de redes são os responsáveis pela coordenação (controle de atividades e monitoração do uso)

de recursos materiais (modems, roteadores, etc) e lógicos (protocolos), fisicamente distribuídos na rede, assegurando, na medida do possível, confiabilidade, tempos de resposta aceitáveis e segurança das informações [1].

Entretanto, soluções de gerência que eram adequadas para outras tecnologias, em especial na comutação de pacotes, têm se mostrado insuficientes para o ATM. A diversidade de serviços oferecidos, o ambiente de alta velocidade e a necessidade de uma solução integrada exigem novos requisitos por parte do sistema de gerenciamento [4].

Além disso, as ferramentas de gerência devem antecipar possíveis problemas antes que eles aconteçam e não apenas atuar de maneira reativa. É importante observar um comportamento anormal da rede, coletar seus sintomas e diagnosticar corretamente um problema maior que possa vir a acontecer. Ações de gerenciamento que visam a causar mudanças em vez de apenas reagir a elas são consideradas pró-ativas. Portanto, a adoção de uma abordagem pró-ativa é desejável em um sistema de gerência, em especial em ambientes ATM que demandam novos requisitos. Técnicas de Inteligência Artificial e Computacional têm sido utilizadas para dotar os sistemas de gerência, do conhecimento que os tornem capazes de uma atitude pró-ativa. Entre estas, as redes neurais artificiais têm despertado grande atenção. As redes neurais são inspiradas no funcionamento dos neurônios do cérebro humano. Assim, podem aprender e generalizar diante de um caso desconhecido. Além disso, são adaptáveis, robustas e tolerantes a falhas [5].

Além da pró-atividade, as questões referentes à complexidade, ao custo e à escalabilidade da gerência de redes têm sido discutidas. Diferentes estudos indicam que o caminho para a resolução destes problemas é a distribuição de inteligência entre os componentes da rede. Entre as propostas apresentadas, o paradigma de agentes inteligentes parece ser a solução mais promissora [6]. Agentes inteligentes são elementos de software autônomos responsáveis pela realização de tarefas no lugar do usuário ou de algum processo [7].

Este trabalho apresenta RENATA (**RE**des **N**eurais **A**plicadas ao **T**ráfego **A**TM), uma nova proposta para o desenvolvimento de agentes inteligentes, baseados em redes neurais, destinados à gerência pró-ativa de redes ATM. Devido às suas características particulares, as redes neurais possibilitam a esses agentes a detecção antecipada de situações anormais em uma rede ATM. Os agentes da RENATA poderão, assim, atuar diretamente ou apenas comunicar o problema ocorrido ao administrador da rede. Este poderá, então, tomar medidas preventivas que evitem maiores complicações no sistema gerenciado.

A arquitetura RENATA tem sido implementada contemplando desde a obtenção das informações necessárias ao treinamento da rede neural, passando pela sua criação até o desenvolvimento do agente. Este interage com os mecanismos de gerência ATM e com a rede neural para possibilitar o desejado comportamento pró-ativo. Além da especificação das arquiteturas física e funcional, é descrito um protótipo que integra as ferramentas necessárias para a implementação dos agentes.

Este trabalho tem a seguinte organização:

- A parte 1 (*Conceitos Básicos*) contextualiza este trabalho e apresenta as tecnologias envolvidas. O capítulo 1 introduz aspectos da gerência de redes, concentrando-se nas especificações da gerência Internet. O capítulo 2 discute a tecnologia de agentes inteligentes, assim como sua aplicação na gerência pró-ativa. Como possíveis componentes destes agentes, as redes neurais são discutida no capítulo 3. O capítulo 4 caracteriza a tecnologia ATM, sua arquitetura e aspectos básicos. O capítulo 5 apresenta as técnicas e infra-estrutura da gerência ATM.

- A parte 2 (*RENATA*) apresenta a principal contribuição deste trabalho: RENATA (Redes Neurais Aplicadas ao Tráfego ATM). O capítulo 6 descreve as arquiteturas funcional e física da RENATA. A primeira define a funcionalidade de cada módulo da arquitetura e comos estes interagem para o desenvolvimento de agentes inteligentes baseados em redes neurais. Já a arquitetura física descreve as ferramentas implementadas e os produtos utilizados no protótipo. O capítulo 7 apresenta o protótipo, descrevendo o ambiente implementado. O capítulo 8 comenta características e o funcionamento do agente desenvolvido, apresentando também detalhes de sua implementação e da estrutura de dados do protótipo. Esta seção também descreve um estudo de caso realizado a partir de trabalhos relacionados. Por fim, o capítulo 9 apresenta conclusões e sugestões de trabalhos futuros.

- A parte 3 (*Apêndices*) traz informações complementares a este trabalho. O apêndice A mostra a definição da MIB RENATA, criada para permitir o monitoramento dos agentes desenvolvidos no protótipo. Já o apêndice B detalha a estrutura de dados utilizada neste trabalho.

Parte I

Conceitos Básicos

Capítulo 1

Gerência de Redes

1.1 Introdução

Dentro das organizações, as redes de computadores tendem a aumentar de tamanho e complexidade, suportando um número cada vez maior de usuários e aplicações. Rapidamente, sistemas vitais das organizações passam a depender das redes. Conseqüentemente, quedas de desempenho e falhas passam a ser indesejáveis. Assim, sua gerência adquire importância fundamental.

A gerência de redes pode ser conceituada como a coordenação (controle de atividades e monitoração do uso) de recursos materiais (modems, roteadores, etc) e lógicos (protocolos), fisicamente distribuídos na rede, assegurando, na medida do possível, confiabilidade, tempos de resposta aceitáveis e segurança das informações [1].

A complexidade do gerenciamento de uma grande rede impõe o uso de ferramentas automatizadas. A necessidade por estas ferramentas e a dificuldade em provê-las aumentam se a rede inclui equipamentos de diferentes fornecedores com soluções próprias de gerência.

Para garantir a interoperabilidade, diversas organizações têm se preocupado em criar padrões para a gerência de redes. Visando tratar toda complexidade dos sistemas de comunicação, uma abordagem orientada a objetos foi adotada pela ISO (*International Standardization Organization*) dentro do contexto OSI (*Open Systems Interconnection*).

Na gerência OSI, cada elemento de interesse à gerência é modelado como um objeto com seus atributos, operações que pode executar, notificações que pode emitir e seu relacionamento com outros objetos [8].

Uma abordagem mais simples e, conseqüentemente, mais fácil de ser implementada, foi adotada pela IETF (*Internet Engineering Task Force*) através da especificação do SNMP (*Simple Network Management Protocol*). Contando atualmente com uma imensa base de objetos e agentes instalados, o SNMP se tornou o padrão *de facto* da gerência de redes

de computadores.

Este capítulo tem a seguinte organização. Na seção 1.2, são explicados conceitos básicos de gerência de redes, tais como: Objeto Gerenciado, paradigma Gerente-Agente, MIB e SMI. A seguir, a seção 1.3 comenta os princípios e a complexidade da gerência OSI. Já as propostas da IETF (SNMPv1, SNMPv2 e RMON) são detalhadas na seção 1.4.

1.2 Conceitos Básicos

1.2.1 Objeto Gerenciado e MIB

Objeto Gerenciado é um importante conceito em gerência de redes, sendo definido como a representação das características relativas à gerência de um recurso físico (uma conexão, um dispositivo, etc) ou lógico (um protocolo) da rede.

Dependendo da granularidade que se deseje na gerência dos recursos, um Objeto Gerenciado pode modelar um recurso real (um roteador, por exemplo) ou uma relação entre recursos (uma rota, por exemplo), entre outros. A definição de um Objeto apresenta dois aspectos: onde ele se situa (sua localização dentro do sistema sendo gerenciado) e sua natureza (representada por seus atributos). Qualquer elemento de rede não modelado como um Objeto Gerenciado é invisível ao sistema de gerência convencional.

A MIB (*Management Information Base*) é o repositório conceitual dos Objetos Gerenciados dentro de um sistema, não importando o meio para armazenamento físico das informações de gerência. O conceito de MIB não impõe nenhuma condição, em nível de normalização, de sua estrutura interna ou em nível de organização local de uma base de informações.

Existem dois tipos de interações entre o sistema de gerência e os Objetos Gerenciados:

- Estática: recuperação das informações de gerência representadas nos Objetos Gerenciados e contidas no interior de uma MIB. Por exemplo, a consulta do valor do atributo que identifica o administrador da rede.

- Dinâmica: corresponde à manutenção da coerência das informações de gerência no interior da MIB. Os recursos reais podem gerar eventos que são transmitidos aos respectivos Objetos Gerenciados, os quais atualizam os atributos na MIB e enviam notificações para o sistema de gerência. No outro sentido, quando o sistema de gerência atua sobre o Objeto Gerenciado, este atualiza os atributos na MIB e aplica a referida ação sobre o recurso. No sentido recurso-Objeto, um exemplo seria a atualização do status de determinada conexão. Como exemplo do outro sentido, se o valor do atributo que indica o horário da próxima reinicialização do sistema for alterado para corrente, o sistema será reinicializado.

1.2.2 Paradigma Gerente-Agente

Por natureza, a gerência de redes é uma atividade distribuída, já que o local de interesse pelas informações é diferente do local onde estas informações se encontram. Assim, a atividade de gerência é dividida em dois processos: Gerente e Agente (figura 1.1).

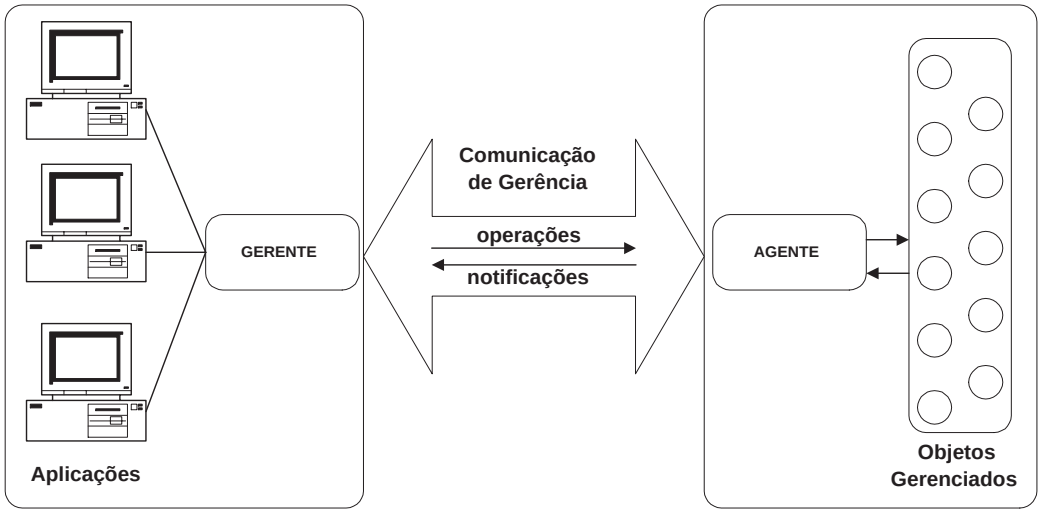

Figura 1.1: Modelo Gerente-Agente

- Processo Gerente: parte associada ao usuário da gerência, realiza operações sobre os Objetos Gerenciados, através do processo Agente. O Gerente recebe e trata as notificações enviadas pelo Agente, repassando ao usuário informações de seu interesse.
- Processo Agente: parte associada ao elemento de rede. Executa sobre o Objeto Gerenciado os comandos enviados pelo Gerente, provendo-o de uma visão dos objetos. Além disso, emite notificações ao Gerente que refletem o comportamento dos objetos.

1.2.3 SMI

Como a informação é a base de todo processo de gerência, é necessário adotar-se especificações bem claras na definição desta informação. A SMI (*Structure of Management Information*) é o conjunto de regras a serem obedecidas na identificação e na definição de objetos na MIB.

Tanto a SMI OSI como a SMI Internet determinam que todos os nomes e tipos de objetos na MIB devem ser definidos em ASN.1 (*Abstract Syntax Notation 1*), uma linguagem formal padronizada pela ISO que determina regras para a definição de sintaxes

abstratas para aplicações de dados e para suas Unidades de Dados de Protocolos (PDUs - *Protocol Data Unit*) [9].

A utilização de ASN.1 facilita a implementação dos protocolos de gerência, embora não garanta a interoperabilidade entre sistemas heterogêneos, por estes possuirem diferentes representações para os tipos considerados.

As estruturas a seguir mostram a definição de dois Objetos Gerenciados. O primeiro deles, *sysDescr*, é um objeto definido em SMI Internet. O segundo objeto é definido em SMI OSI, *pduCounterObject*.

```
sysDescr OBJECT-TYPE
    SYNTAX  DisplayString (SIZE (0..255))
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
            "A textual description of the entity.  This value
            should include the full name and version
            identification of the system's hardware type,
            software operating-system, and networking
            software.  It is mandatory that this only contain
            printable ASCII characters."
    ::= { system 1 }
```

É fácil perceber que a estrutura de um objeto OSI é bem mais complexa.

```
pduCounterObject MANAGED OBJECT CLASS
     DERIVED FROM ``CCITT REC.X.721 (1992)|ISO...;
     CHARACTERIZED BY
         basePackage PACKAGE -- in-line PACKAGE definition
               ATTRIBUTES pduCounterName
                           GET
                          pduCounter
                           INITIAL VALUE sintax.initialZero
                           GET
         ; -- End of in-line PACKAGE definition
     ; -- End of CHARACTERIZED BY construct
     CONDITIONAL PACKAGES additional Package
         PRESENT IF *enable/disable control is required*;
REGISTERED AS {object-identifier 1}
```

1.3 Gerência OSI

Os padrões de gerência OSI se desenvolveram mais lentamente que outros padrões OSI. Foi necessário pesquisar os requerimentos da gerência de redes e só a partir deles, definir um conjunto de serviços de comunicação e de funções de gerência.

Um Objeto Gerenciado OSI é definido em termos de seus atributos, operações, notificações que pode emitir e de suas relações com outros objetos. Estas relações são definidas segundo três representações hierárquicas, a saber:

- *Hierarquia de Herança.* Define relações de herança entre as classes de Objetos Gerenciados, relacionando suas propriedades (atributo, comportamento, pacotes condicionais, operações e notificações). Uma subclasse herda todas propriedades da superclasse, podendo definir propriedades adicionais (como na orientação a objetos).

- *Hierarquia de Agregação.* Define as relações entre instâncias de Objetos do tipo "está contido".

- *Hierarquia de Registros.* Identifica os Objetos Gerenciados, independentemente das hierarquias de herança e agregação.

Para a definição de Objetos Gerenciados, é necessário seguir uma série de especificações, chamada de GDMO (*Guidelines for the Definition of Managed Objets* - ISO/IEC 10165-4). GDMO provê uma ligação entre os conceitos abstratos de modelagem contidos no Modelo de Informações de Gerência (*Management Information Model* - ISO/IEC 10165-1) e os requerimentos concretos da especificação de classes de objetos gerenciados que permitam a gerência de recursos em um ambiente OSI [8].

Uma aplicação de gerência pode executar o papel de Gerente, de Agente ou ambos. O Agente OSI possui um poder de processamento bem maior que seu correspondente Internet. O Agente OSI possui uma série de serviços que suportam várias funções como geração de eventos (devidamente discriminados e enviados para os Gerentes interessados), controle de log e controle de acesso.

A OSI definiu um modelo de comunicação entre sistemas abertos baseado no protocolo CMIP (*Common Management Information Protocol*) e no serviço CMISE (*Commom Management Information Service Element*). O CMISE define um conjunto de elementos de de serviços de comunicaçáo representando as funções de base para a troca de informações ou de comandos CMIP entre as aplicações de gerência. As operações realizadas pelo CMIP podem referir-se aos atributos dos objetos ou ao objeto como um todo.

Áreas Funcionais da Gerência de Redes

A ISO definiu um modelo que visa a descrição dos requerimentos da gerência, dividindo suas atividades em cinco áreas funcionais [9]:

- Gerenciamento de Falhas: meios para permitir a detecção, isolamento e correção de operações anormais na rede. Exemplo de atividade: correlação de alarmes.
- Gerenciamento de Contabilização: meios para permitir a medição, a responsabilidade e os custos pelo uso dos Objetos Gerenciados. Exemplo: medições de utilização dos recursos.
- Gerenciamento de Configuração: meios para controlar e identificar Objetos Gerenciados, além de coletar e provê-los de dados que facilitem o fornecimento contínuo de seus serviços. Exemplo: seleção e agrupamento de recursos.
- Gerenciamento de Desempenho: meios para avaliar o comportamento dos Objetos Gerenciados e a eficiência de suas atividades. Exemplo: armazenamento de dados sobre o gerenciamento de tráfego.
- Gerenciamento de Segurança: meios que garantam o funcionamento correto da gerência e que protejam os Objetos Gerenciados. Exemplo: controle de acesso ao sistema de gerência.

1.4 Gerência Internet

1.4.1 Introdução

O SNMP foi desenvolvido como parte da família de protocolos TCP/IP (*Transmission Control Protocol/Internet Protocol*) [10], na qual está baseada a atual Internet.

Até meados da década de 80, pouca atenção era dada à gerência de redes TCP/IP propriamente dita, por exemplo, ao desempenho dos roteadores e de entidades associadas (no sentido OSI). O que havia de mais popular era o ping (*Packet Internet Groper*), um comando mais associado à administração de máquinas do que à gerência de redes. O ping permite o teste de conectividade de máquinas e de simulação do retardo da rede através de *timestamps*. Com o crescimento das redes, começou-se a pensar num protocolo formal de gerência.

Inicialmente, foi proposto que as soluções a curto e a longo prazo seriam, respectivamente, o SNMP e o CMOT (*CMIP over TCP/IP*). O CMOT foi uma tentativa de incorporar ao máximo possível na gerência internet o protocolo (CMIP), os serviços e a estrutura dos dados da gerência OSI.

Porém, logo se percebeu a dificuldade da ligação entre os dois protocolos em nível de objeto. Na gerência internet, os objetos SNMP não são considerados como tais à luz da orientação a objetos, e sim, variáveis simples com algumas características básicas, como seu tipo e seu modo de acesso.

Liberado da restrição de compatibilidade com o modelo OSI, o SNMP teve rápido desenvolvimento, tornando-se o padrão *de facto* da gerência de redes. O SNMP têm sido utilizado inclusive sobre outras tecnologias de rede que não o TCP/IP.

Várias MIBs SNMP têm sido desenvolvidas, modelando desde aspectos específicos de interfaces padrões (como *Token Ring*) a extensões privadas de fabricantes de equipamentos de rede.

Existe um limite até onde o SNMP pode ser estendido apenas pela definição de novas MIBs. O RMON (*Remote Monitoring*) é um grande avanço no funcionamento do SNMP, através da adição de semântica a uma MIB [9]. RMON permite que o gerente monitore uma sub-rede como todo e não apenas individualmente cada dispositivo.

Contudo, quando o SNMP é aplicado a redes maiores e mais sofisticadas, suas deficiências se tornam mais visíveis: sua insegurança e pouca funcionalidade. Algumas destas deficiências são tratadas na segunda versão do SNMP, o SNMPv2, lançado em 1993. Atualmente, está em curso de definição a terceira versão do SNMP. Estas deficiências e todas estas propostas serão comentadas a seguir.

1.4.2 SNMPv1

SMI

Todos os Objetos Gerenciados SNMP são arranjados como folhas numa estrutura de árvore, dentro de grupos logicamente relacionados. Cada objeto é identificado por uma seqüência de inteiros, o seu OID (*Object Identifier*).

A definição de um objeto numa MIB SNMP contém seu tipo de dado (campo SYNTAX), seu modo de acesso (campo ACCESS), seu status (campo STATUS), sua descrição (campo DESCRIPTION) e seu relacionamento com outros objetos na MIB. A notação ASN.1 é utilizada para definir cada objeto e a estrutura da MIB.

MIB-II

A MIB-II [11] (RFC 1213) define a segunda versão da MIB Internet. Seus objetos são todos obrigatórios e cobrem um largo espectro de Objetos Gerenciados. A MIB-II trata aspectos gerais dos dispositivos e do funcionamento dos protocolos TCP/IP.

A figura 1.2 mostra sua estrutura. Os grupos da MIB-II estão assim definidos:

- **system**. Informações gerais sobre o sistema.

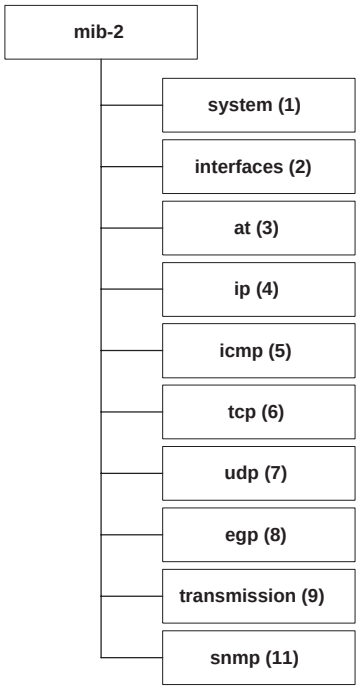

Figura 1.2: MIB-II

- **interfaces**. Informações sobre cada interfaces do sistema com a subrede.
- **at**. Informações de mapeamento de endereços. Não é mais implementada.
- **ip**. Informações relacionadas à implementação e operação do protocolo IP no sistema.
- **icmp**. Informações relacionadas à implementação e operação do protocolo ICMP (*Internet Control Message Protocol* no sistema.
- **tcp**. Informações relacionadas à implementação e operação do protocolo TCP no sistema.
- **udp**. Informações relacionadas à implementação e operação do protocolo UDP (*User Datagram Protocol*) no sistema.
- **egp**. Informações relacionadas à implementação e operação do protocolo EGP (*External Gateway Protocol* no sistema.
- **transmission**. Informações sobre os esquemas de transmissão e protocolos de acesso em cada interface do sistema.

- **snmp**. Informações relacionadas à implementação e operação do protocolo SNMP no sistema.

Protocolo

Como parte dos protocolos TCP/IP, o SNMP foi projetado para operar no nível de aplicação sobre o UDP (*User Datagram Protocol*). Assim como este, o SNMP é um protocolo não orientado à conexão, isto é, nenhuma conexão é mantida entre o Gerente e seus Agentes.

A figura 1.3 mostra as operações do SNMP. As operações são realizadas através de trocas de PDUs entre Gerente e Agente. Quando um Agente recebe uma PDU, ele testa a sintaxe da mesma, a versão do SNMP e sua autenticidade. A autenticação é feita com base no campo *community* da PDU, que funciona como uma senha. Caso algum dos testes falhe, a PDU é descartada.

A segurança baseada em *community* é fraca, pois um invasor pode observar a mensagem SNMP na rede para capturar a *community* de determinado dispositivo. Tendo a capturado, ele pode desabilitar ou mudar a configuração deste em seguida. Por isso, muitos fabricantes permitem apenas monitoramento sobre seus Objetos e não o controle.

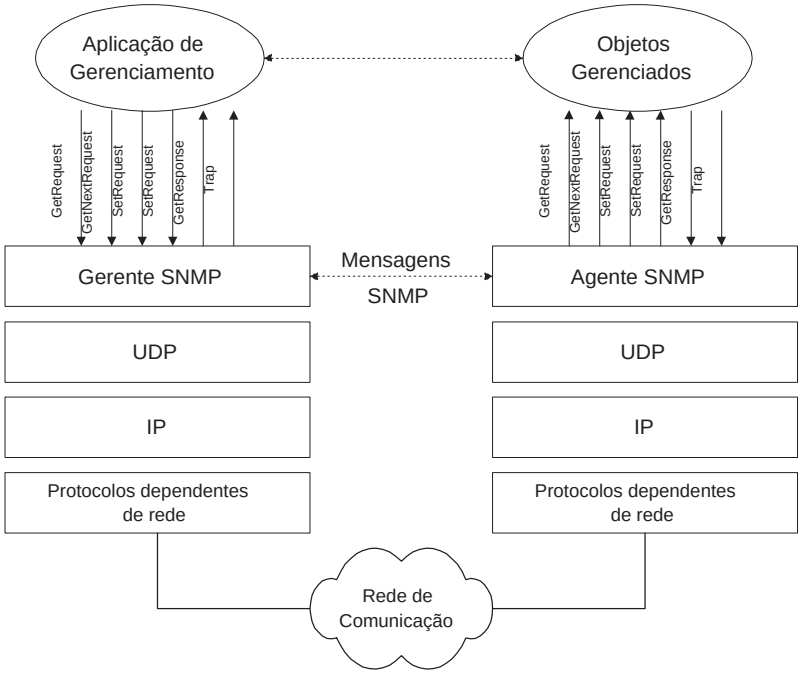

Figura 1.3: Operações do SNMPv1

As PDUs de GetRequest e GetNextRequest permitem que o Gerente obtenha o valor de Objetos Gerenciados no Agente. Para ambas, uma PDU de GetResponse é retornada. A diferença é que no GetNextRequest o valor do Objeto seguinte ao indicado na PDU na ordem lexicográfica será obtido. A PDU SetRequest permite que o Gerente altere o valor de um Objeto no Agente, causando assim alguma ação. Este retorna uma PDU GetResponse com o mesmo identificador de PDU (request-ID). Já a PDU Trap permite que o Agente notifique o Gerente de eventos importantes nos Objetos Gerenciados. A figura 1.4 mostra o formato das PDUs SNMP.

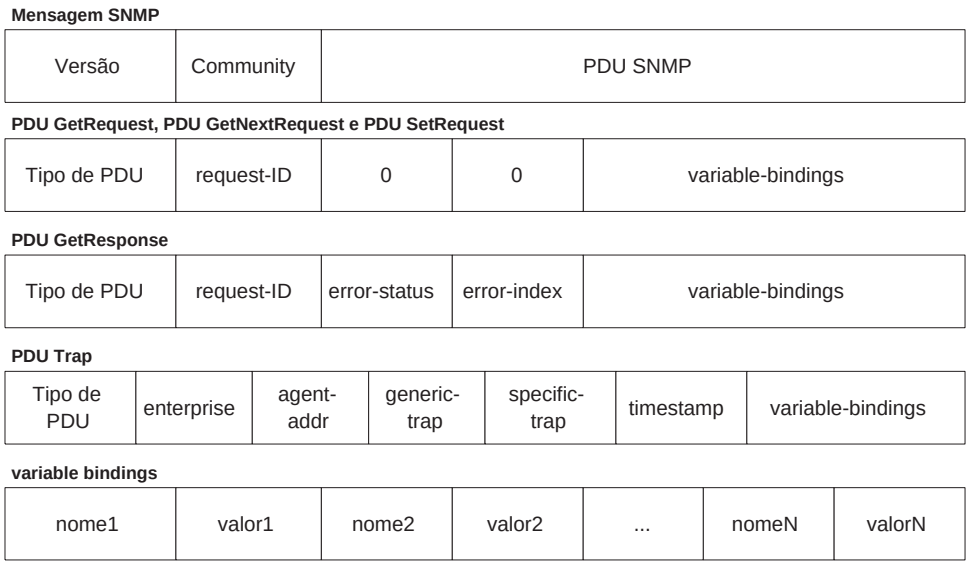

Figura 1.4: Mensagens SNMPv1

1.4.3 SNMPv2

Inicialmente, duas propostas foram apresentadas na tentativa de correção das deficiências do SNMPv1. O *secure SNMP* tratava o problema de segurança, enquanto que o SMP (*Simple Management Protocol*) propôs melhorias nas questões do desempenho e da funcionalidade do protocolo. No entanto, houve consenso em torno de uma proposta única lançada em março de 1993, referida como SNMPv2.

As principais inovações do SNMPv2 encontram-se nos seguintes pontos:

- SMI
- Operações do Protocolo
- Comunicação Gerente-Gerente

- Segurança

A SMI SNMPv2 expande a primeira versão para incluir novos tipos de dados e melhorar a documentação associada a Objetos. A SMI SNMPv2 também define uma convenção para criação e remoção de *linhas conceituais*. Estas permitem o aumento no número de linhas numa tabela, sem ser necessário sua redefinição.

A MIB SNMPv2 foi definida, sendo análoga ao grupo snmp da MIB-II. Esta MIB contém informações sobre o tráfego gerado pelo protocolo, além de informações relacionadas à configuração de Agentes e Gerentes.

Quanto ao protocolo, duas novas PDUs foram definidas: GetBulkRequest e InformRequest. A primeira permite que o Gerente obtenha grandes blocos de dados de uma vez, aumentando a eficiência do protocolo. A PDU InformRequest permite que um Gerente envie notificações a outro gerente. A figura 1.5 mostra as PDUs do SNMPv2.

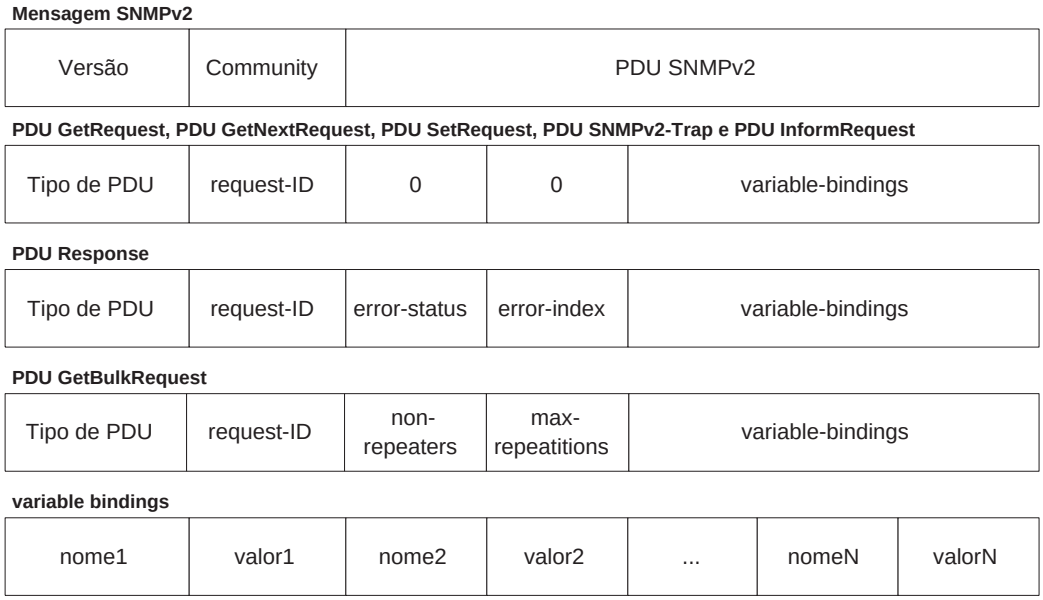

Figura 1.5: Mensagens SNMPv2

A comunicação Gerente-Gerente também é suportada por uma nova MIB, a M2M (*Manager-to-Manager*). Esta MIB pode ser utilizada para permitir que um Gerente intermediário desempenhe o papel de monitor remoto de informações de tráfego de uma sub-rede e também possa enviar notificações.

Na parte de segurança, as especificações do SNMPv2 garantem autenticidade e controle de acesso, a partir da utilização de criptografia e da introdução do conceito de *SNMPv2 party*. A troca de informações entre entidades SNMPv2 é realizada realmente entre *parties*, que garantem a segurança destas informações.

Contudo, dado o pouco entusiasmo entre os fabricantes diante do SNMPv2, a IETF decidiu lançar uma adaptação do SNMPv2, onde esta parte de segurança baseada em *parties* é retirada, voltando ao mecanismo de *community* do SNMPv1. Esta versão é conhecida como SNMPv2C, isto é, SNMPv2 baseado em *community*.

1.4.4 RMON

O RMON (*Remote Monitoring*) é empregado para estudar o tráfego em uma rede como um todo, superando as limitações do SNMPv1, onde um Gerente pode obter apenas informações localizadas em um determinado equipamento. Tipicamente, estes monitores (também chamados de *probes*) operam em uma rede de modo promíscuo, visualizando todos os pacotes que passam através dela. A figura 1.6 mostra um exemplo de configuração de uma rede com RMON.

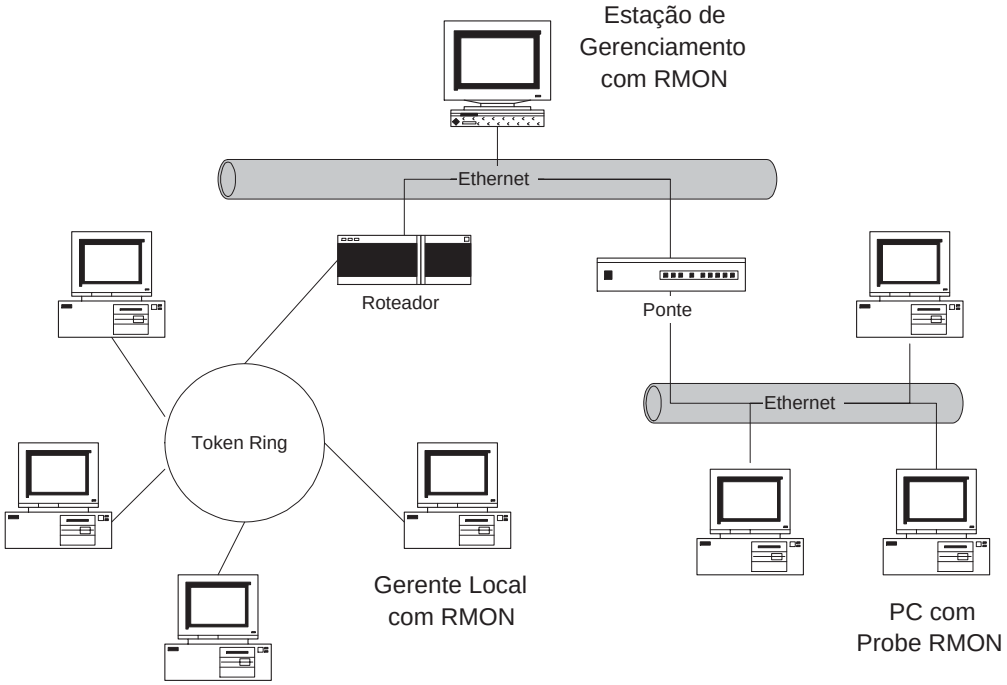

Figura 1.6: Exemplo de Configuração com RMON

A gerência de redes remotas através de Agentes remotos (Agentes que implementam a RMON MIB) possui cinco funções:

- Operações off-line: são as operações nas quais uma estação de gerenciamento não necessita estar em contato direto com seus dispositivos de monitoração remotos. Essa função na RMON MIB permite que os Agentes sejam configurados para realizar

diagnósticos e coletar estatísticas continuamente, mesmo que a comunicação entre a estação de gerenciamento não seja possível ou não seja eficiente;

- Monitoração pró-ativa: os recursos disponíveis nos monitores são potencialmente úteis para continuamente executar diagnósticos e manter logs do desempenho da rede. Essas informações são importantes para desenvolver a função *baseline*. A função *baseline* refere-se ao fato de manter um histórico da operação normal de uma rede por um tempo estendido, com o objetivo de analisar essas informacoes posteriormente para identificar problemas potenciais numa rede;

- Detecção e registro de problemas: o monitor remoto pode fazer o reconhecimento de determinadas condições, realizando constantes averiguações;

- Tratamento dos dados coletados: o monitor pode executar análises específicas nos dados coletados em suas sub-redes, aliviando o gerente desta função;

- Múltiplos gerentes: na configuração de uma rede pode haver mais de uma estação de gerenciamento como forma de oferecer maior nível de disponibilidade, para executarem diferentes funções ou ainda para gerenciar diferentes sub-redes.

1.4.5 SNMPv3

Atualmente, está em curso de definição a terceira versão do SNMP. O SNMPv3 é especificado nas seguintes RFCs:

- RFC 2271, que descreve uma arquitetura para definição de ambientes de gerenciamento SNMP [12].

- RFC 2272, que define processamento e envio de mensagens [13].

- RFC 2273, que especifica várias aplicações SNMPv3 [14].

- RFC 2274, que descreve o USM (*User-based Security Model*), um modelo de segurança que provê autenticação e privacidade de mensagens SNMP [15].

- RFC 2275, que define o VACM (*View-based Access Control Model*), que define mecanismos de controle de acesso a visões dos Objetos Gerenciados na MIB [16].

Em fevereiro de 1999, foram lançados documentos (*Internet Drafts*) que redefinem aspectos destas RFCs, inclusive tratando a coexistência com o SNMPv1 e SNMPv2. Como o SNMPv3 ainda está em processo de especificação, este não foi utilizado neste trabalho.

Capítulo 2

Agentes Inteligentes

2.1 Introdução

Os sistemas de gerência de redes não estão adaptados ao largo espectro com que estas se apresentam nas diversas configurações. Tecnologias emergentes como o CORBA (*Common Object Request Broker*) não parecem capazes de resolver os problemas referentes à complexidade, ao custo e à escalabilidade no suporte à gerência de redes. Diferentes estudos indicam que o caminho para a resolução destes problemas é a distribuição de inteligência entre os componentes da rede. Entre as propostas apresentadas, o paradigma de Agentes Inteligentes parece ser a solução mais promissora [6].

Agente Inteligente é definido em [7] como um "elemento independente de software responsável por uma tarefa. Para realizá-la, este agente contém algum nível de inteligência, variando de simples regras pré-definidas a máquinas de inferência. Ele age no lugar de um usuário ou de um processo, permitindo a automação de tarefas. Agentes Inteligentes operam de maneira autônoma sendo freqüentemente ativados por eventos ou por tempo e podem se comunicar com o usuário, com recursos do sistema e com outros agentes para realizar sua tarefa. Agentes Inteligentes mais sofisticados podem cooperar para realização de tarefas não resolvidas individualmente".

A tecnologia de Agentes Inteligentes deve ser vista como a aplicação integrada de várias tecnologias [17]. Três conceitos merecem destaque ao tratar-se de Agentes Inteligentes:

- *Inteligência Artificial.* O interesse por aplicações de Inteligência Artificial mais simples, flexíveis e reusáveis tem aumentado devido à influência cada vez maior da orientação a objeto. Este interesse facilita o desenvolvimento de agentes inteligentes.

- *Orientação a Objetos.* A popularização da orientação a objetos e o desenvolvimento das arquiteturas de Objetos Distribuídos [18] têm resultado na necessidade de objetos que também sejam autônomos e móveis. Normalmente, Agentes Inteligentes

são implementados em linguagens orientadas a objeto. Além disso, podem utilizar arquiteturas de Objetos Distribuídos.

- *Redes de Computadores.* A expansão das redes mudou a forma como os computadores são utilizados: os recursos estão distribuídos em sistemas diferentes e heterogêneos, e as informações estão dispersas na rede. Os agentes inteligentes normalmente utilizam as redes de computadores.

Este capítulo tem a seguinte organização. A seção 2.2 descreve as características dos agentes inteligentes. O modelo conceitual de um agente inteligente é apresentado na seção 2.3. A seguir, as aplicações desta tecnologia na gerência de redes são discutidas. Nesta seção, aspectos relacionados à gerência pró-ativa e à mobilidade de agentes também são comentados.

2.2 Características

Muitos trabalhos discutem a definição e a taxonomia dos agentes inteligentes, existindo muita polêmica em torno de uma definição única. A seguir, são identificadas possíveis características destes agentes [6]:

- **Autonomia**. O agente decide quando e em que condições deve agir. Em [19], um agente autônomo é definido como "um sistema situado em um ambiente, avaliando e agindo sobre ele de acordo com seus planos, de modo a mudar sua avaliação no futuro".

- **Comunicação**. O agente pode se comunicar com outro agente, com o usuário ou com um dispositivo. Para a comunicação entre os agentes, são freqüentemente utilizadas as seguintes técnicas:

  - *Blackboard*: os agente lêem e escrevem mensagem em uma área comum.

  - KQML (*Knowledge Query and Manipulation Language*) [20]: uma linguagem e um protocolo para troca de informações e de conhecimento.

  - KIF (*Knowledge Interchange Format*) [21]: um cálculo estendido de predicados de primeira ordem para comunicações agente-servidor e agente-agente.

- **Cooperação**. Agentes podem cooperar para realizar uma tarefa, podendo coordenar suas ações. Enquanto coordenação pode ocorrer sem obrigatoriamente haver colaboração, esta necessita que haja negociação entre os agentes.

- **Deliberação**. Agente deliberativo é definido em [22] como "aquele que possui um modelo simbólico do mundo e no qual decisões são tomadas através de raciocínio lógico ou pseudo-lógico, baseado em associação de padrões e manipulação de símbolos".
- **Mobilidade**. Os agentes móveis podem ser transportados (código e dados) de um sistema para outro. Freqüentemente, são baseados em Java [37].
- **Aprendizado**. Considera-se que o agente aprende quando é capaz de utilizar seu conhecimento para modificar seu comportamento. O objetivo de aprendizado é tornar o agente capaz de realizar novas tarefas, sem interromper sua execução.
- **Pró-atividade**. Ações pró-ativas pretendem causar mudanças, ao invés de apenas reagir a elas. Agentes pró-ativos geralmente seguem planos ou executam regras quando algum parâmetro do ambiente atinge determinado limite.
- **Reatividade**. O agente é capaz de reagir quando algum determinado evento ocorre.
- **Segurança**. O agente deve ser capaz de discriminar "amigos" de "inimigos" e de elementos "contaminados".
- **Planejamento**. O agente organiza por prioridade as ações tomadas durante sua existência.
- **Delegação**. O agente pode pedir a outra entidade a realização de alguma de suas tarefas, visando um equilíbrio de carga.

2.3 Modelo Conceitual

A figura 2.1 apresenta os elementos de um agente, segundo [17]. Estes elementos estão divididos em dois fatores: Inteligência e "*Agency*". Inteligência refere-se ao raciocínio, aprendizado ou a outra técnica utilizada para interpretar a informação ou o conhecimento aos quais o agente tem acesso. *Agency* é o grau de percepção que o agente possui do seu ambiente e de sua ação sobre ele.

Os elementos referentes à Inteligência são a Maquinaria e o Conteúdo. Maquinaria refere-se aos mecanismos que suportam variados níveis de inteligência presentes nos agentes. Estes mecanismos são desenvolvidos principalmente no domínio da Inteligência Artificial. Conteúdo são os dados utilizados pela Maquinaria no raciocínio e aprendizado do agente.

Segurança e Acesso são os elementos relacionados com a *Agency*. Segurança são as preocupações usuais com computação distribuída, acrescentadas de algumas específicas

Figura 2.1: Elementos de um Agente Inteligente

aos agentes inteligentes. O Acesso é o grau de interação do agente com seu ambiente. Este permite que a Maquinaria compreenda o Conteúdo, para que ações sejam realizadas de acordo com o resultado de seu raciocínio. A Maquinaria é ativada por Eventos, que são a detecção de algum novo conhecimento, indicando uma mudança no ambiente do agente.

A figura 2.2 mostra o Modelo Conceitual de um agente inteligente, apresentado em [17]. Os Eventos e o Conhecimento do agente, que constituem o Conteúdo, são combinados na Maquinaria de Raciocínio para decidir que ação tomar. O agente, de maneira autônoma, realiza a ação através de seus métodos de Acesso, sujeitos às restrições e aos testes impostos pela Segurança. Por fim, a Maquinaria de Aprendizado é capaz de aprender a partir dos Eventos, o que leva o agente à aquisição de novo Conhecimento.

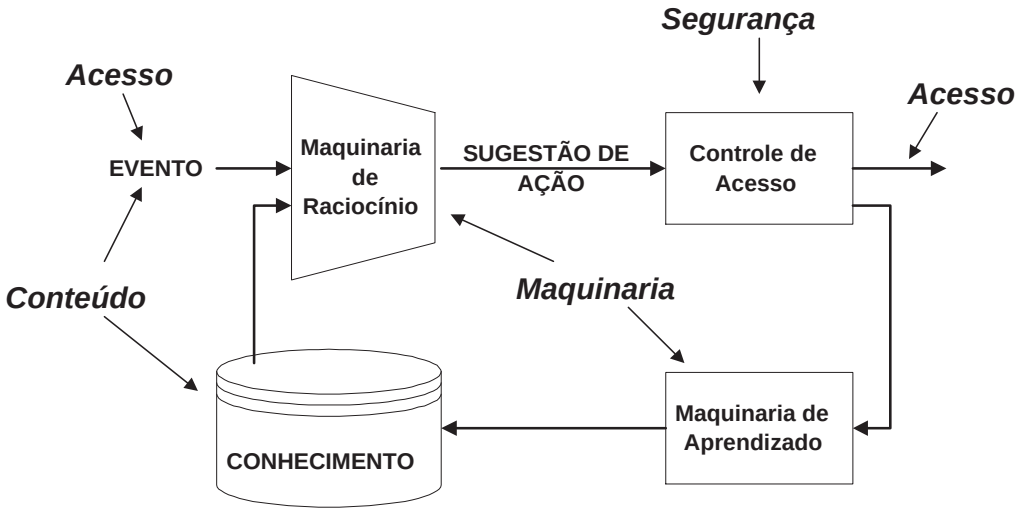

Figura 2.2: Modelo Conceitual de um Agente Inteligente

2.4 Agentes Inteligentes na Gerência de Redes

Os protocolos e modelos clássicos de gerência de redes convencionais possuem suas deficiências. Os dados estão distribuídos em agentes com capacidades computacionais limitadas e o processamento dos dados para extração de informação de gerência é centralizado nos gerentes, que assim têm grande carga. Além disso, as operações devem ser decompostas em funções primitivas, basicamente limitadas a GETs e SETs.

Para superar essas deficiências e com o intuito de prover uma arquitetura mais flexível, escalável e com capacidades de processamento em tempo real, admite-se que as funções e operações de gerência devem ser realizadas dinamicamente, próximas de onde os Objetos Gerenciados estão [6].

O conceito de Gerenciamento por Delegação (MbD - *Management by Delegation*) segue este princípio. MbD é baseado no conceito de "servidor elástico", apresentado em [23]. Um servidor elástico pode ter suas funcionalidades estendidas ou diminuídas dinamicamente em tempo de execução. Um protocolo de delegação permite que um cliente atribua, através de *scripts*, novas funções a um servidor elástico e depois pedir a execução destas funções.

Vários trabalhos têm aplicado este conceito na tecnologia de agentes inteligentes. Em [24], um ambiente baseado em uma planilha de *scripts* é apresentado. Cada célula da planilha é definida por uma expressão que calcula algum valor a partir de outros dados, como valores numa MIB. Também utilizando o SNMP, uma linguagem de *scripts* permite que o gerente delegue computações a um agente. Já em [25], uma unidade de processamento de regras é adicionada ao agente. Regras podem ser carregadas dinamicamente, atualizadas e removidas, com o intuito de tornar o agente flexível.

2.4.1 Gerência Pró-Ativa

Agentes Inteligentes podem ser utilizados na gerência pró-ativa, que tem como objetivo detectar situações anormais, utilizando um perfil de comportamento da rede que permita ao sistema evitar problemas. Uma atitude pró-ativa compreende medidas preventivas ou reativas de menor impacto, dentro de uma ação planejada. Esta abordagem pode ser associada a técnicas de Inteligência Artificial.

Em [26], é proposto o uso de modelagem e ferramentas de simulação para o desenvolvimento de *baselines* para a gerência pró-ativa. Já em [27], o uso de agentes na identificação de problemas é explorado, a partir da aplicação de um sistema especialista. Enfocando redes ATM, o trabalho [28] também utiliza agentes com regras de produção e *frames* para diagnosticar situações problemáticas antecipadamente. Na mesma linha, podemos citar ainda [29] e [30].

RENATA inscreve-se no contexto de interesse do LAR (Laboratório Multinstitucional de Redes e Sistemas Distribuídos), onde a gerência pró-ativa tem recebido atenção especial. Dentre as propostas do LAR, destacam-se a aplicação de banco de dados ativos, descrita em [31], e a implementação de um sistema especialista para apoio à gerência, apresentada em [32].

2.4.2 Agentes Móveis

Agentes Móveis são agentes que podem se transportar de um nó para outro da rede para execução remota. Diferentemente de RPC (*Remote Procedure Call*) que limita-se ao envio de dados para execução remota, os Agentes Móveis tranpostam código e dados [17].

Para o desenvolvimento de Agentes Móveis, deve-se utilizar linguagens de programação que possibilitem o transporte de código e dados, a definição do comportamento dos agentes durante a migração e testes de segurança. Algumas opções são:

- **TCL e Safe-TCL**. Linguagem baseada em *scripts* que utiliza correio eletrônico para transportar os procedimentos do agente como conteúdo destas mensagens [33]. Safe-TCL isola programas desconhecidos em um "célula de proteção", num ambiente seguro.

- **Telescript**. Linguagem baseada em *scripts*, também oferece ambiente para execução e desenvolvimento de Agentes Móveis. Possui mecanismos de autenticação. Telescript foi adaptado para também utilizar Java [17].

- **Java**. Linguagem orientada a objeto desenvolvida pela Sun, permite serialização de objetos capturando seu estado e transporte deste código através de RMI (*Remote Method Invocation*). Applets podem ser autenticadas para acessar recursos do sistema.

Em [7], foram apresentados aspectos da aplicação de Agentes Móveis na gerência de redes. Agentes Móveis podem encapsular *scripts* de gerência para que estes executem sob demanda onde for necessário. Um agente pode ser enviado para coletar dados entre os elementos de uma rede e voltar com os dados filtrados e coletados. Se o tamanho do agente permanecer pequeno, haverá economia no uso da banda. Se o agente puder tomar decisões durante suas operações, mais banda será economizada.

O trabalho [34] sugeriu a utilização de Agentes Móveis no controle de congestionamento numa rede com comutação de circuitos. Uma classe de agentes fica migrando entre os nós da rede e coletando informações sobre sua utilização. Monitorando esta informação, os agentes são capazes de perceber quando o nó está congestionado. Quando esta situação é

identificada, um agente móvel para balanceamento de carga é criado. Ele deve atualizar as tabelas de roteamento dos nós vizinhos de modo a reduzir o tráfego para aquele nó.

Outra aplicação é permitir que um administrador possa se conectar remotamente e disparar um Agente Móvel para realizar determinada tarefa. Depois ele se conecta novamente para receber os resultados da operação do agente.

Embora seja uma abordagem poderosa, a aplicação de Agentes Móveis deve ser examinada com atenção. Questões relativas à segurança, à real utilização de banda e à eficiência da solução devem ser consideradas quando da aplicação desta tecnologia [6]. Na sua primeira versão, RENATA não utiliza agentes móveis. Contudo, estes poderiam ser ativados, por exemplo, para coleta de dados pela rede. Algumas funções realizadas para gerência pró-ativa ou por agentes móveis podem ser adaptadas para utilização dos agentes RENATA.

Capítulo 3

Redes Neurais

3.1 Introdução

As redes neurais artificiais são sistemas computacionais maciçamente paralelos, consistindo de um grande número de processadores simples com muitas interconexões. As redes neurais possibilitam um novo paradigma para a solução de problemas devido às suas características de aprendizado, adaptabilidade, robustez e tolerância a falha [5][35].

As redes neurais artificiais são inspiradas nas redes neurais biológicas. Os neurônios são bem mais lentos que os computadores digitais, mas a inferência humana é bem mais rápida. O cérebro compensa a operação relativamente lenta com o enorme número de neurônios (cerca de 10^{11} a 10^{14}), extremamente interconectados (cada neurônio tem entre 10^3 a 10^4 conexões), funcionando de maneira paralela e descentralizada. Enquanto nos computadores o conhecimento é estritamente substituível, ele é adaptável no cérebro. Nova informação é adicionada pelo ajuste da força de conexão entre os neurônios. O cérebro exibe características de tolerância a falhas. Danos individuais em neurônios podem ocorrer sem maior degradação do desempenho geral. Além disso, a generalização para casos desconhecidos é permitida a partir de tarefas e exemplos conhecidos [5].

As redes neurais artificiais tentam imitar algumas destas características do cérebro. Possuindo um grande número de processadores simples com muitas interconexões, o conhecimento é adquirido pelas redes neurais através de um processo de aprendizagem, onde este é armazenado em forma de pesos nas conexões.

As redes neurais são capazes de realizar tarefas que envolvem processamento simultâneo de grande quantidade de dados e onde respostas rápidas e acuradas são necessárias. Algumas funções das redes neurais são [35]:

- *Classificação.* Capacidade de analisar dois objetos ou estados e, de acordo com suas características, estimar suas similaridades e diferenças.

- *Segmentação.* Capacidade de agrupar conceitos semelhantes.
- *Memória Associativa.* Capacidade de associar objetos ou idéias com memórias relacionadas.
- *Modelagem.* Capacidade de modelar relacionamentos entre exemplos, para possibilitar generalização diante de casos novos.
- *Predição de Séries Temporais.* Capacidade de prever acontecimentos em algum ponto do futuro, dado o acontecido no passado.
- *Satisfação de Restrições.* Capacidade de solucionar problemas com restrições conflitantes.

Cada unidade de processamento atua como um reconhecedor simples de padrões. Ele recebe entradas de outros elementos, compara estas com sua memória e produz um sinal de saída que corresponde ao grau de semelhança entre os padrões de entrada. Mais precisamente, esses sinais de entrada passam por conexões com peso (sua memória), que aumentam ou diminuem os sinais. Dentro de cada processador, os sinais de entradas são totalizados. Esse valor total de entrada é submetido a uma função para produzir a saída deste processador, variando de 0 a 1.

O modelo mais utilizado de unidade de processamento é o contínuo. A figura 3.1 mostra sua estrutura. Neste modelo, a unidade é definida por uma *regra de propagação* e uma *função de ativação.* Os dados dos pesos das conexões (w_n) e das entradas (x_n) de cada unidade são agregados pela regra de propagação $\sum$ (geralmente por uma função somatório com pesos ou da distância euclidiana) e o resultado é processado pela função de ativação f (geralmente sigmoidal, limiar ou linear *piecewise*), produzindo a saída da unidade de processamento (o).

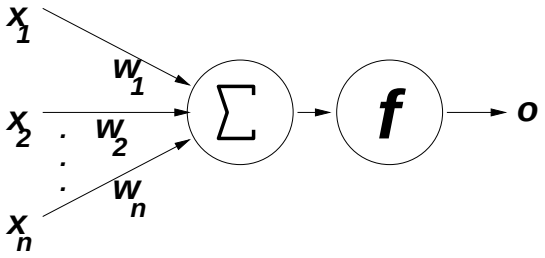

Figura 3.1: Estrutura de uma Unidade de Processamento

A maneira como as unidades de processamento estão arranjadas numa rede neural está diretamente relacionada com a forma de aprendizado desta.

Este capítulo têm a seguinte organização. A seção 3.2 apresenta os paradigmas de aprendizado das redes neurais. As topologias e modelos de redes neurais são descritas nas seções 3.3 e 3.4, respectivamente. A seção 3.5 comenta aspectos relacionados ao seu treinamento. Por fim, a seção 3.6 discute a aplicação de redes neurais como componente dos agentes inteligentes.

3.2 Paradigmas de Aprendizado das Redes Neurais

Entre as características mais relevantes das redes neurais, estão sua capacidade em aprender informações do seu ambiente e sua habilidade em modificar seu desempenho e forma de atuação em função do aprendizado [36].

A etapa de aprendizado, o treinamento da rede neural, trata-se de um processo iterativo de ajuste dos pesos das conexões entre as unidades de processamento. Estes pesos guardam no final do processo o conhecimento aprendido, consistindo uma representação própria do problema.

Os três maiores paradigmas de aprendizado são o *supervisionado*, *não supervisionado* e *por reforço*; descritos a seguir.

3.2.1 Aprendizado Supervisionado

Aprendizado supervisionado é útil no treinamento de redes neurais, que visam realizar classificações, modelagem e predição de séries temporais.

Nesta abordagem, um caso é apresentado à rede neural, que faz sua predição. O algoritmo de aprendizado obtém a diferença entre a saída esperada e a realizada. Esta diferença é utilizada no ajuste do pesos, de modo que na próxima vez a predição seja mais próxima da esperada. Deste modo, a rede neural aprende a relação entre as entradas e as saídas.

Existem problemas reais altamente não lineares, que possuem relações complexas entre as variáveis e para os quais nenhuma função matemática é conhecida ou não pode ser facilmente derivada. Para estes casos e para aqueles em que o problema em si pode mudar com o tempo, as redes neurais com aprendizado supervisionado podem ser utilizadas.

3.2.2 Aprendizado Não Supervisionado

O aprendizado não supervisionado é utilizado quando se possui grandes quantidades de dados, mas não se sabe a saída desejada. Neste caso, deseja-se que a rede neural descubra o relacionamento entre os dados, dividindo-os em grupos de elementos similares entre

si. Por isso, este tipo de aprendizado é utilizado para redes neurais que vão realizar segmentação.

As redes neurais treinadas sob este paradigma possuem a capacidade de se organizar. Quando apresentados a uma série de entradas, as unidades de saída se organizam inicialmente pela competição no reconhecimento do padrão e depois cooperando para o ajuste do peso de suas conexões.

3.2.3 Aprendizado por Reforço

Neste paradigma, possui-se os exemplos do problema, mas não a resposta, pelo menos não de maneira imediata. A informação sobre como os pesos devem ser alterados é buscada no ambiente, através da observação de como o sistema está se saindo no ambiente. Se uma ação tomada é seguida por um estado satisfatório, então a tendência do sistema em produzir aquela ação em particular deve ser reforçada. Caso contrário, a tendência deve ser reduzida.

Redes neurais treinadas sob este paradigma permitem a resolução de problemas que dependem de elementos temporais. Além disso, elas podem ser aplicadas quando a informação de *feedback* não está disponível e apenas sinais secundários da situação são visíveis.

3.3 Topologias de Redes Neurais

O modo como as unidades de processamento e suas interconexões estão arranjadas possuem grande impacto no poder computacional da rede neural.

Em geral, todas as redes neurais possuem um conjunto de unidades de processamento que recebem os estímulos exteriores, sendo chamados como "camada de entrada". Muitos modelos de redes neurais também possuem uma ou mais "camadas intermediárias", constituídas por unidades que só recebem estímulos de outras. Por fim, o conjunto de unidades que representam a computação da rede neural é denominado como "camada de saída".

Existem três grandes tipos de topologia, que definem como os dados fluem entre as unidades de entrada, intermediárias e de saída: *feedforward*, *parcialmente recorrentes* e *totalmente recorrentes*.

3.3.1 Redes Neurais Feedforward

Neste tipo de rede neural (figura 3.2), os dados fluem em apenas uma direção e a resposta é baseada somente no presente conjunto de entradas. Na maioria dos casos, as unidades

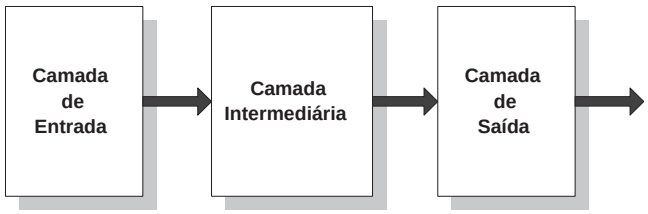

Figura 3.2: Redes Neurais Feedforward

de uma camada são totalmente conectadas às unidades da próxima camada.

3.3.2 Redes Neurais Parcialmente Recorrentes

Redes neurais recorrentes (figura 3.3) são utilizadas quando se tem as informações a serem submetidas, mas a seqüência das entradas é importante. É necessário que a rede neural registre as entradas passadas e que as considere com a entrada corrente para produzir uma resposta. Para isto, alguns modelos sugerem uma "camada de contexto" para prover as camadas intermediárias de *feedback*.

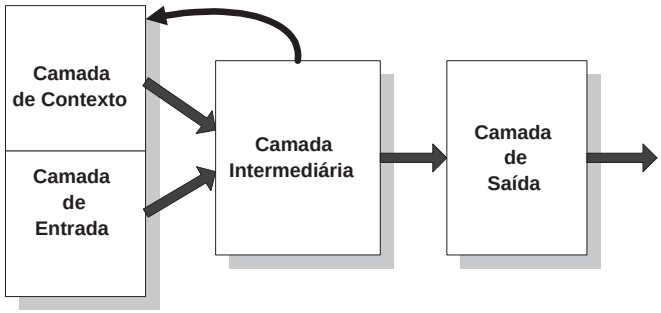

Figura 3.3: Redes Neurais Parcialmente Recorrentes

3.3.3 Redes Neurais Totalmente Recorrentes

Redes neurais totalmente recorrentes (figura 3.4) apresentam conexões nos dois sentidos para cada unidade de processamento. A camada de entrada recebe os estímulos exteriores. Então, os dados fluem para todas as unidades adjacentes e ficam circulando até que a ativação das unidades se estabilize. Neste ponto, a saída da rede neural pode ser lida na devida camada.

Redes neurais totalmente recorrentes são sistemas complexos e dinâmicos, que podem exibir comportamento caótico. Diferentemente das redes *feedforward*, as totalmente

recorrentes podem levar um tempo indeterminado para produzir uma saída. Estas redes neurais são utilizadas principalmente para problemas de otimização e de memória associativa.

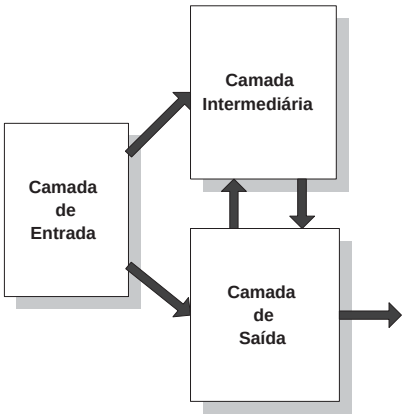

Figura 3.4: Redes Neurais Totalmente Recorrentes

3.4 Modelos de Redes Neurais

A combinação da topologia, do paradigma de aprendizado e de seu algoritmo definem um modelo de rede neural. Existe uma grande variedade de modelos. Por exemplo, alguns são otimizados para ter um treinamento rápido e outros para ter memória de entradas passadas. Contudo, o melhor modelo para uma certa aplicação depende dos dados e das funcionalidades requeridas. Detalhes sobre diversos modelos de redes neurais podem ser encontrados em [5]. A tabela 3.1 relaciona alguns modelos com seu paradigma de aprendizado, topologia e funções.

3.5 Treinamento de Redes Neurais

É na fase de treinamento que a rede neural adquire o conhecimento que vai permitir a realização de funções como classificação e predição de série temporais, entre outras. Este conhecimento é representado pelos pesos das conexões entre as unidades de processamento, que são ajustados durante o treinamento. O processo é extremamente dependente de fatores como a disponibilidade e características dos dados, do modelo da rede neural e de sua função.

Na maioria dos modelos, os pesos são inicializados com valores aleatórios. À medida que os exemplos são apresentados à rede neural, os pesos são ajustados de acordo com o

Modelo	Paradigma de Aprendizado	Topologia	Funções Primárias
Back Propagation	Supervisionado	*Feedforward*	Classificação
Recurrent Back Propagation	Supervisionado	P. Recorrente	Modelagem, Séries Temporais
Adaptative Resonance Theory	Não supervisionado	P. Recorrente	Segmentação
ARTMAP	Supervisionado	P. Recurrente	Classificação
Kohonen Feature Maps	Não supervisionado	*Feedforward*	Segmentação
Radial Basis Function	Supervisionado	*Feedforward*	Classificação, Modelagem, Séries Temporais

Tabela 3.1: Modelos e Funções de Redes Neurais

algoritmo de aprendizado. Este é dependente do modelo; por exemplo, um dos mais utilizados é o algoritmo *back propagation* (também conhecido como regra delta generalizada).

Deve-se monitorar o treinamento da rede neural para determinar se ela está realmente aprendendo. O método de "tentativa e erro" pode consumir muito tempo e recursos. Em [35], é proposta uma abordagem para o desenvolvimento de redes neurais em projetos de aplicações ou para suporte a decisões. A figura 3.5 mostra as fases propostas do processo iterativo de treinamento de uma rede neural.

Seleção dos Dados

Definido o problema, deve-se obter o banco de exemplos que será utilizado no treinamento. É necessário determinar quais são os parâmetros importantes no processo de decisão. A etapa de seleção exige conhecimento profundo sobre o domínio do problema e dos dados envolvidos.

Os dados devem ser divididos em dois ou três conjuntos: dados de treinamento, dados para teste e para validação, se for necessário. A rede neural aprende através dos dados de treinamento. Depois desta fase, deve-se verificar se ela realmente aprendeu, a partir da observação do seu desempenho nos padrões de testes, aos quais não foi apresentada anteriormente. Opcionalmente, os dados de validação podem ser utilizados por outra pessoa para comprovar o aprendizado da rede neural.

Representação dos Dados

Os dados devem ser tratados antes de serem submetidos à rede neural. Alguns parâmetros podem ser combinados em um só ou serem eliminados em caso de redundância. Freqüen-

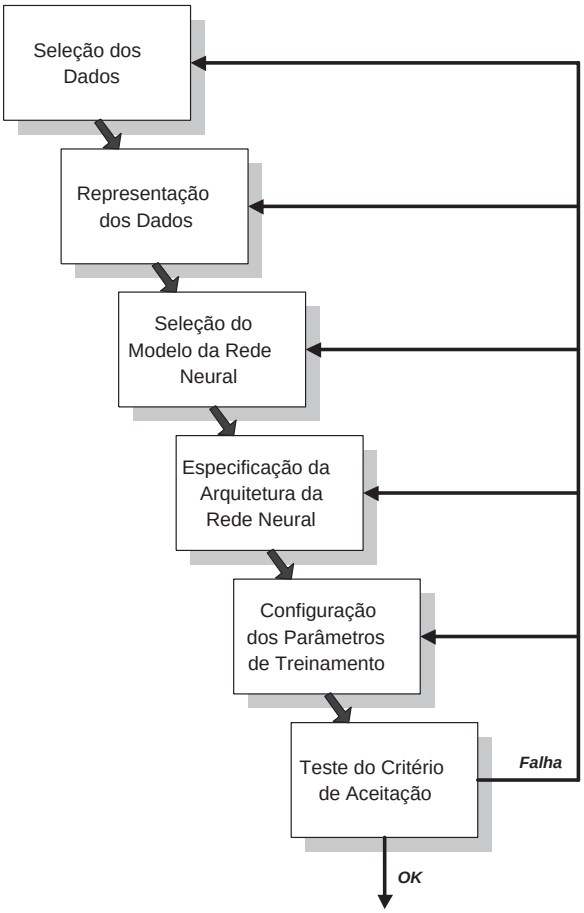

Figura 3.5: Processo de Treinamento de Redes Neurais

temente, os dados devem ser transformados para uma forma que seja aceitável para a rede neural. Neste casos, várias técnicas podem ser utilizadas:

- *Escalonamento* é aplicado quando a rede neural só aceita entradas reais no intervalo $[0, 1]$ ou $[-1, 1]$. Assim, os dados devem ser escalonados para um destes intervalos.
- *Normalização* é utilizada quando deseja-se tratar vetores de dados numéricos como um grupo. Nestes casos, pode ser necessário normalizá-los ou escaloná-los agrupadamente.
- No caso de valores simbólicos, funções *hash* ou de *mapeamento* geram dados numéricos correspondentes. Estes podem ser escalonados para o intervalo desejado ou passarem por uma *Codificação* para valores binários.

A forma de representação dos dados é importante. Se escolhas erradas forem feitas, é possível que a rede neural não consiga aprender a relação entre os dados. Além disso, a

forma de representação afeta diretamente o tempo de treinamento e a precisão obtida.

Seleção do Modelo da Rede Neural

Antes de selecionar o modelo de rede neural aplicado, deve-se determinar a função a ser executada. Então, deve-se examinar os dados de entradas. Se estes forem todos binários ou se contiverem valores reais, alguns modelos podem ser desconsiderados. A seguir, a quantidade de dados e a velocidade desejada de treinamento devem ser avaliadas. Por exemplo, redes neurais *Back Propagation* treinam muito lentamente, não devendo ser utilizadas em situação de aprendizado on-line.

Outro aspecto pode ser a influência do tempo nos dados. Se estes tiverem forte elemento temporal, uma rede neural *Recurrent Back Propagation* ou *Radial Basis Function* terá melhor desempenho que uma simples *Back Propagation*.

Especificação da Arquitetura da Rede Neural

O número de camadas intermediárias é determinado na especificação da arquitetura da rede neural. Durante o treinamento, é possível que a rede neural não esteja convergindo por não poder tratar a complexidade do problema. A adição de mais unidades intermediárias (ou mesmo de outra camada destas) aumenta o poder computacional da rede neural. Contudo, um equilíbrio nesse número deve ser obtido para evitar longo tempo de treinamento ou a perda de poder de generalização da rede neural, no caso desta apenas memorizar os padrões de treinamento sem extrair nenhum relacionamento dos dados.

Configuração dos Parâmetros de Treinamento

Entre outros fatores, estes parâmetros determinam a velocidade do treinamento e o grau de generalização da rede neural. Alguns desses parâmetros são gerais, como a função de ativação, e vários são específicos a cada modelo. A taxa de aprendizado é um parâmetro geral que determina o grau das mudanças realizadas nos pesos visando a saída desejada. Quando se aplica uma alta taxa de aprendizado, os pesos oscilam muito, mas o treinamento é mais rápido. A tabela 3.2 mostra alguns destes parâmetros.

Teste do Critério de Aceitação

O Critério de Aceitação indica se a rede neural realmente aprendeu o desejado. Seu desempenho é testado diante dos padrões de teste. Se a rede neural atingir um valor aceitável de erros, ela está treinada.

Este critério depende do modelo e da função que a rede neural deve executar. Para problemas de predição, os erros RMS (*Root Mean Squared*) e o MSE (*Mean Squared Error*)

Parâmetro	Modelo	Descrição
Taxa de Aprendizado	Todos	Controla o tamanho do passo para ajustes dos pesos. Diminui com o tempo em alguns modelos.
Função de Ativação	Todos	Seleciona a função de ativação utilizada na unidade de processamento.
Momentum	*Back Propagation*	Suaviza os efeitos dos ajustes no peso em função do tempo.
Error Tolerance	*Back Propagation*	Especifica quão próximo valor da saída deve estar antes que o erro seja considerado zero.
Vigilance	ART	Especifica quão similares os padrões de entrada devem ser para serem classificados na mesma categoria.
Neighborhood	*Kohonen Maps*	Define o tamanho da área ao redor da unidade de saída vencedora que será atualizada. Diminue com o tempo.
Número de Épocas	*Kohonen Maps* e outros	Determina o número fixo de vezes que as redes neurais de alguns modelos passa pelos dados de treinamento.

Tabela 3.2: Parâmetros de Treinamento de Redes Neurais

são boas medidas da precisão da rede neural.

3.6 Agentes Inteligentes e Redes Neurais

A incorporação na Maquinaria do agente inteligente de tecnologias como redes neurais possibilita o diagnóstico de certos tipos de problemas e a solução destes de maneira mais rápida e mais acurada, quando comparada com soluções apenas algorítmicas [37].

Outras opções de Maquinaria (regras de produção, sistemas especialistas, lógica fuzzy ou algoritmos genéticos) são possíveis. No entanto, o foco deste trabalho está na aplicação de redes neurais e na criação de facilidades para o desenvolvimento de agentes inteligentes, baseados em redes neurais, para a gerência pró-ativa de redes ATM.

Capítulo 4

Redes ATM

4.1 Introdução

As aplicações em redes de computadores têm exigido maior largura de banda, bem como requisitos mais acurados em relação ao atraso e à taxa de perda nos meios utilizados. As tecnologias convencionais utilizadas na comunicação de dados têm se mostrado incapazes de garantir Qualidade de Serviço (QoS) ao integrar numa mesma infra-estrutura serviços de voz, vídeo e dados textuais.

Além de prover mais banda, uma tecnologia que satisfizesse esses requisitos também deveria fazê-lo de maneira mais eficiente. Como as aplicações multimídia produzem tráfego à taxa variável, deve-se fazer uma alocação otimizada de recursos. Além disso, essa tecnologia deveria ter padrões abertos; de modo que a interoperabilidade entre produtos de fabricantes diferentes fosse garantida, possibilitando uma maior competição entre os fornecedores e promovendo uma maior acessibilidade da tecnologia. Se essa tecnologia possuísse, ainda, a mesma arquitetura para LANs e WANs, a complexidade de integração seria reduzida e uma solução universal seria possível.

ATM (*Asynchronous Transfer Mode*) se apresenta como uma solução viável para satisfação desses requisitos. Trata-se de uma tecnologia de comutação e multiplexação usada para transportar pequenos pacotes de tamanho fixo, chamados células, sobre uma rede de alta velocidade. ATM suporta a integração e transporte de dados, voz e vídeo com diferentes níveis de QoS sobre a mesma rede, além de possuir funções similares para LANs e WANs [38].

Neste capítulo, os conceitos básicos da tecnologia ATM serão apresentados. Além disso, serão discutidos aspectos como: interfaces, conexões, célula, arquitetura, endereçamento, *LAN Emulation* e IP sobre ATM. A última seção apresenta algumas aplicações de redes neurais em ATM.

4.2 RDSI-FL

O conceito de integração das tecnologias de comutação e transmissão surgiu no final da década de 50, no entanto, o primeiro conjunto de recomendações do ITU-T saiu em 1984. A idéia por trás das RDSI (Redes Digitais de Serviços Integrados) é a chamada "Tomada de Informações", a exemplo da tomada elétrica, uma interface universal para o acesso a informações. Portanto, as RDSI devem fornecer uma interface comum para a transferência de dados.

De acordo com o ITU-T, uma RDSI é "uma rede, em geral evoluída da Rede Digital Integrada (RDI) de telefonia, que proporciona conectividade digital fim-a-fim, para suportar uma variedade de serviços vocais e não-vocais, aos quais os usuários têm acesso através de um conjunto limitado de interfaces usuário-rede padronizadas". Na sua fase inicial, as RDSI (denominada posteriomente de Faixa Estreita - FE) consistem na integração de serviços, mas dependendo ainda de redes dedicadas para atendê-los (figura 4.1).

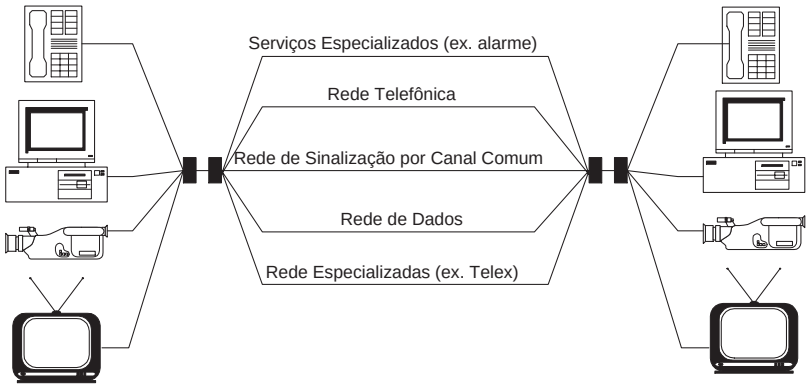

Figura 4.1: RDSI-FE

Os padrões atuais para RDSI-FE definem um acesso básico à taxa de 144 Kbps (dois canais B de 64 Kbps e um canal D de sinalização de 16 Kbps) e um acesso primário com taxas correspondentes às dos canais T-1 ou E-1 (1.5 ou 2 Mbps, respectivamente).

Percebeu-se, logo, que taxas maiores de transferências são necessárias para aplicações como interconexão de LANs, vídeo e imagem, levando o processo de padronização à introdução dos conceitos de RDSI-FL (Redes Digitais de Serviços Integrados de Faixa Larga).

Os conceitos de RDSI-FL são apresentados na Recomendação I.121 do ITU-T desta forma:

"RDSI-FL suporta conexões comutadas, semi-permanentes e permanentes, ponto-a-ponto e ponto-a-multiponto; devem prover serviços em demanda, reservados e permanentes. Conexões em RDSI-FL suportam tanto serviços modo circuito ou modo pacote, de

uma única ou várias mídias, orientada à conexão, em configuração uni ou bi-direcional. Uma RDSI-FL conterá capacidades inteligentes para prover serviços avançados que suportem ferramentas poderosas de operação e manutenção, controle e gerência da rede".

Diferentemente da RDSI-FE, não apenas o acesso será integrado como também haverá uma única rede de transporte, ATM (figura 4.2).

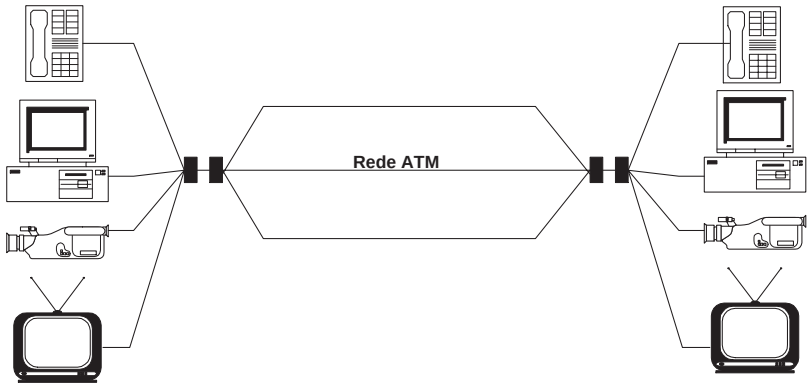

Figura 4.2: RDSI-FL

Apesar de RDSI-FL e ATM serem termos usados indistintamente, rigorosamente RDSI-FL é o conjunto de padrões do ITU-T que definem sinalização e transporte em banda larga, além da gerência de serviços integrados sobre uma WAN. ATM é o modo de transporte para RDSI-FL. O modo de transporte define como a informação suprida pelos usuários da rede é eventualmente mapeada na rede física. Segundo o ITU-T, o modo de transporte é a técnica usada para tratar os aspectos de transmissão, multiplexação e comutação de redes de comunicação.

Apesar de ter sido concebida para redes públicas de altas velocidades, ATM também apresenta-se como boa alternativa para redes locais/corporativas. Já existem tecnologias de alta velocidade para LANs, como: FDDI (*Fiber Distributed Data Interface*), DQDB (*Distributed Queue Dual Bus*) e *Fast Ethernet*. Mas para a interconexão destas redes através da RDSI-FL ou acesso a outras redes, é necessário prover modos de interfuncionamento. Por outro lado, se a tecnologia da rede local for a mesma da utilizada pela rede pública, a compatibilidade é total. Isto, mais as garantias de QoS, são a motivação para a aplicação da tecnologia ATM também em ambientes locais/corporativos.

4.3 Características Gerais de ATM

ATM provê um serviço orientado à conexão. Isso significa que uma rota e uma conexão devem ser configuradas entre duas estações ATM antes que os dados dos usuários se-

jam transmitidos. Durante a conexão, os dados percorrerão a mesma rota. Serviço não orientado à conexão é suportado, mas o fluxo dos dados passa por uma ou mais rotas pré-estabelecidas.

Conexões virtuais são o mecanismo utilizado para conectar e transportar dados em ATM. A conexão virtual é dedicada a uma origem e a um destino, não podendo ser compartilhada por mais nenhuma outra. Uma ou mais conexões virtuais podem executar sobre o mesmo enlace físico.

ATM usa um pacote de tamanho fixo, chamado célula. A célula ATM tem 53 bytes, com 48 bytes de dados (*payload*) e 5 bytes de cabeçalho. Manter pequeno o tamanho da célula proporciona flexibilidade ao modo em que a largura de banda pode ser usada, além de favorecer o uso eficiente de buffers e banda através da multiplexação estatística.

Comutadores são utilizados para interconectar estações e redes ATM. O comutador contém uma tabela de rotas, que consiste das portas do comutador (de entrada e saída) e o identificador de conexão (presente no cabeçalho de célula) para cada conexão que passa pelo comutador.

ATM suporta QoS. Isso significa que uma rede ATM proverá ou reservará recursos que vão garantir *throughput* mínimo, atraso e perda de dados máximos para a duração de determinada conexão. O suporte à QoS por conexão permite que ATM atenda concorrentemente qualquer tipo de tráfego (dados, voz e vídeo) sobre uma mesma rede.

O assincronismo em ATM vem da natureza das aplicações que fazem as células chegarem em intervalos irregulares à camada Física, podendo ser transmitidas a um destino a qualquer momento.

A figura 4.3 mostra como é feita a integração das aplicações em uma rede ATM. Para cada aplicação, seja um canal de voz a 64 Kb ou uma vídeo-conferência, é alocada uma conexão virtual da estação fonte ao destino. As informações destas aplicações são segmentadas em células e multiplexadas sobre um único canal físico até o destino. Lá, as células são remontadas em quadros ou *streams* digitais e repassadas para a aplicação.

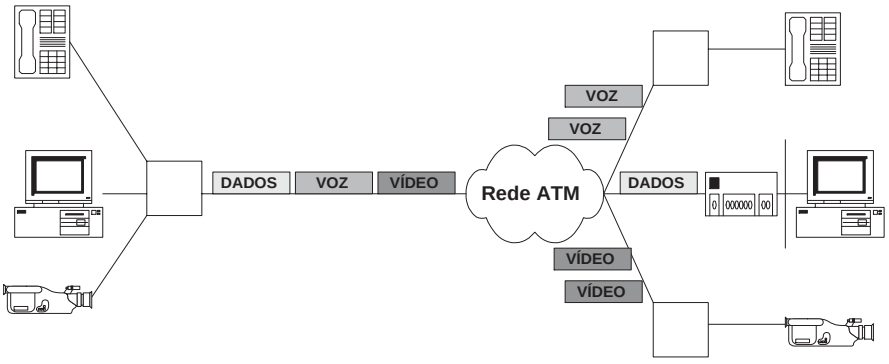

Figura 4.3: Integração de Serviços no ATM

4.4 Interfaces ATM

Os padrões ATM são baseados em diferentes tipos de interfaces, que tratam de questões de conectividade e interoperabilidade entre duas entidades ATM (comutadores, estações ou redes ATM). Na verdade, interfaces são associações lógicas e funções que são executadas entre duas entidades.

A figura 4.4 mostra a estrutura de um ambiente ATM genérico, com suas interfaces e componentes [38].

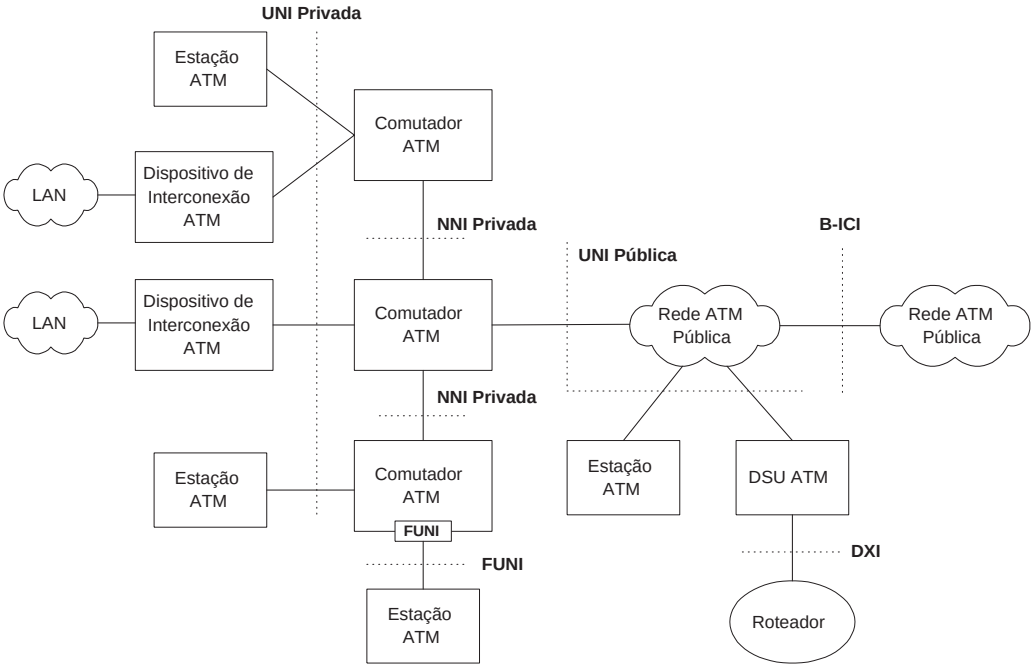

Figura 4.4: Interfaces ATM

UNI

A UNI (*User Network Interface*) define procedimentos e protocolos que permitem que uma estação ATM se conecte e se comunique com um comutador ATM. Existem dois tipos de UNI: a UNI Privada, que é a interface entre uma estação e um comutador numa rede privada; e a UNI Pública, que trata da comunicação entre uma estação ou comutador de rede privada com um comutador de rede pública.

A especificação da UNI inclui funções para:

- Conexão física entre estação e comutador

- Multiplexação/Demultiplexação sobre a conexão física
- Sinalização a um comutador para estabelecimento de conexão permanente ou comutada
- Gerenciamento de tráfego entre estação e comutador
- Endereçamento de estações ATM

NNI

A NNI (*Network-to-Network Interface*) é a interface entre dois comutadores adjacentes. Será uma NNI Privada (PNNI - *Private NNI*) se ambos pertecerem à rede privada e será uma NNI Pública se ambos pertecerem à rede pública.

O ITU-T tem se concentrado na definição da NNI e da UNI pública, tratando aspectos como: camada Física e ATM; planos de Usuário, Controle e Gerenciamento; e sinalização de rede pública. Já o ATM Forum[1] fez recomendações sobre a UNI Privada e a NNI Privada.

B-ICI

A B-ICI (*Broadband Intercarrier Interface*) é uma especificação que permite dois comutadores de rede pública se comunicarem. A B-ICI também provê uma série de serviços fim-a-fim:

- Cell Relay Service (CRS)
- Circuit Emulation Service (CES)
- Frame Relay Service (FRS)
- Switched Multimegabit Data Service (SMDS)

DXI

A DXI (*Data Exchange Interface*) é a interface que permite que um DTE (*Data Terminal Equipment*), como um roteador, e um DCE (*Data Communication Equipment*), como um DSU (*Data Service Unit*), cooperem para prover uma UNI. O DTE manda quadros no

[1] O ATM Forum é uma organização internacional formada com o objetivo de acelerar o uso de serviços e produtos baseados na tecnologia ATM, através da rápida convergência das especificações de interoperabilidade.

formato DXI para o DSU, que os segmenta em células ATM. Assim, o DXI provê acesso a rede ATM sem caras atualizações de hardware.

DXI define um protocolo de enlace e camadas Físicas que tratam das transferências de dados entre o DTE e o DCE [3]. A especificação da DXI também define a DXI LMI (*DXI Local Management Interface*) para tratar a troca de informação de gerência, juntamente com uma MIB.

FUNI

A FUNI (*Frame UNI*) permite que uma estação ATM transmita quadros FUNI de tamanho variável na rede ATM. O comutador ATM FUNI realiza todas as segmentações em célula e remontagens necessárias (figura 4.4). Por causa da maior eficiência de transmissão (maior *payload* em relação ao cabeçalho do quadro), a largura de banda é melhor utilizada. FUNI foi projetada para que os equipamentos ATM utilizassem mais eficientemente enlaces de baixa velocidade, como o T-1 [2].

4.5 Conexões ATM

ATM é orientado à conexão, portanto, requer que uma conexão virtual fim-a-fim seja estabelecida antes que o tráfego seja realmente iniciado. As conexões ATM podem ser permanentes ou comutadas. Conexões permanentes (também conhecidas como PVC - *Permanent Virtual Connections*) são configuradas por um operador via funções de gerência e geralmente permanecem ativas por um longo período de tempo. Conexões comutadas (SVC - *Switched Virtual Connections*) são estabelecidas dinamicamente via sinalização. Normalmente, SVC ficam ativas até um sinal indicar o fim da conexão.

Durante a fase de estabelecimento de conexão, é necessário especificar os parâmetros de QoS da conexão e os endereços completos da origem e do(s) destino(s), dado que as conexões ATM podem ser ponto-a-ponto ou ponto-a-multiponto. A esta conexão, é associado um identificador.

Em cada interface, o identificador tem apenas significado local. A rede se encarrega de fazer o mapeamento adequado entre os vários identificadores que a conexão recebe a cada nó que passa.

O identificador da conexão é composto por dois campos hierárquicos: o identificador de caminho virtual (VPI - *Virtual Path Identifier*) e o identificador de canal virtual (VCI - *Virtual Channel Identifier*). Os dois identificadores, juntos com o enlace físico por onde as células chegam, identificam unicamente a conexão em cada comutador.

[2] Alguns autores não consideram a DXI e a FUNI como interfaces ATM verdadeiras, mas como mecanismos de interoperabilidade.

Um canal virtual (VC) descreve o transporte unidirecional de células ATM com um identificador comum. Um VCC (*Virtual Channel Connection*) é a concatenação de um ou mais VCs. Em outras palavras, VCC é a conexão lógica fim-a-fim entre dois usuários finais ATM.

Como ilustrado na figura 4.5, um caminho virtual (VP) é um grupo de VCs. Cada VC está associado a um VP. Por sua vez, o VP está sempre contido dentro de uma conexão de camada Física. Um VPC (*Virtual Path Connection*) é a concatenação de VPs desde o ponto onde são atribuídos até o ponto onde eles são traduzidos ou removidos.

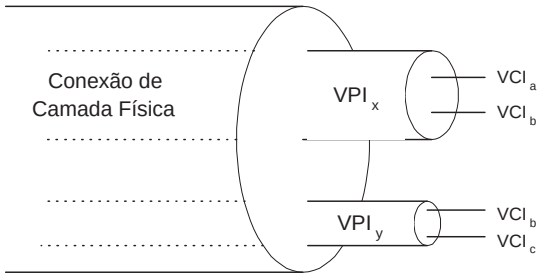

Figura 4.5: Identificadores de Conexão ATM

A Figura 4.6 [39] exemplifica a comutação de canais e caminhos virtuais. Além disso, a figura mostra o conceito de comutador de VP e de VC. Em um comutador de VP, apenas o VPI é alterado. Enquanto que o VPI e o VCI são modificados quando a conexão passa por um comutador de VC.

Durante a fase de estabelecimento de conexão (VPC ou VCC), o usuário especifica as características e exigências do tráfego que deverá gerar. As conexões podem ser comutadas ou permanentes. A comunicação pode ser simétrica (mesma capacidade nos dois sentidos), assimétrica (capacidades diferentes) ou unidirecional (capacidade zero ou mínima para suportar tráfego de gerência em um sentido) [39]. Assim, os conceitos de caminhos e canais virtuais oferecem um mecanismo flexível e robusto para o estabelecimento e comutação de conexões dentro de uma rede ATM.

4.6 Célula ATM

A célula ATM tem um tamanho fixo de 53 bytes, sendo 5 bytes de cabeçalho e 48 de *payload*. A escolha por 48 bytes foi resultado de um acordo dentro dos órgãos de padronização. Para aplicações de voz, um célula menor seria melhor pois levaria menos tempo para ser preenchida. Para aplicações de dados, uma célula maior aumentaria a eficiência de transmissão. Foi feita uma média aritmética das duas propostas (32 e 64 bytes) e 48 foi o resultado.

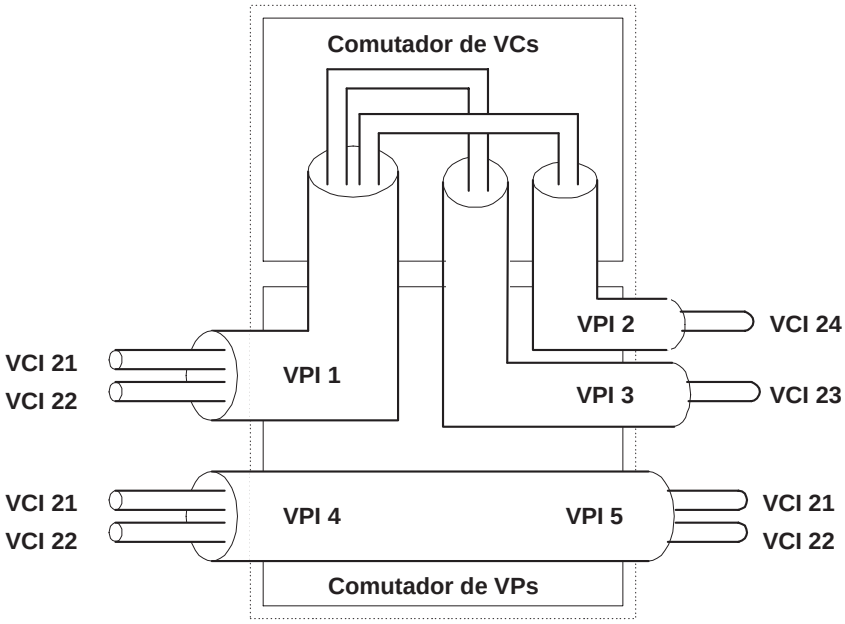

Figura 4.6: Comutação de Canais e Caminhos Virtuais

O formato do cabeçalho depende da interface ATM em questão, UNI (figura 4.7) ou NNI. A diferença é que os quatro bits usados para GFC (*Generic Flow Control*) na célula para UNI são acrescentados para o campo VPI na célula para NNI. Isso porque, por definição, controle de fluxo não é executado na NNI e, assim, grandes redes interconectadas podem suportar um número maior de caminhos virtuais.

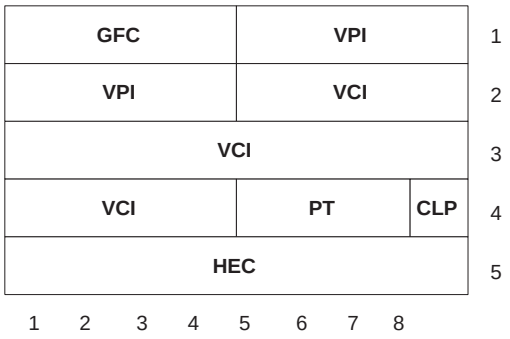

Figura 4.7: Estrutura da Célula ATM (UNI)

A seguir, cada campo do cabeçalho é explicado:

- **GFC**: Campo de 4 bits projetado para possibilitar mecanismos de controle de fluxo entre um usuário ATM e um comutador. A UNI 3.0/3.1 especifica que o GFC tem

valor "0000".

- **VPI** e **VCI**: Identificadores de conexão.
- **PT** (*Payload Type*): Campo de 3 bits que indica se a célula leva informação de usuário, de gerência (OAM - vide capítulo 5) ou gerenciamento de tráfego. Quando a célula contém dados de usuário, a camada ATM ignora o *payload*, apenas o repassando para a camada de Adaptação.
- **CLP** (*Cell Loss Priority*): Campo de 1 bit que indica a prioridade de descarte da célula em caso de congestionamento. Células de baixa prioridade ou de conexões que estejam desrespeitando o contrato de tráfego (CLP=1) serão descartadas primeiro.
- **HEC** (*Header Error Check*): Campo de 8 bits usado pela camada Física na detecção/correção de erros de bits no cabeçalho. Dependendo do meio de transmissão, o HEC também pode ser usado na delimitação da célula.

4.7 Arquitetura ATM

A arquitetura ATM é baseada no Modelo de Referência de Protocolos RDSI-FL (figura 4.8), desenvolvido pelo ITU-T na Recomendação I.321. Trata-se de um modelo multidimensional, consistindo de três planos (Usuário, Controle e Gerenciamento) para três camadas (camada Física, ATM e Adaptação ATM - AAL[3]).

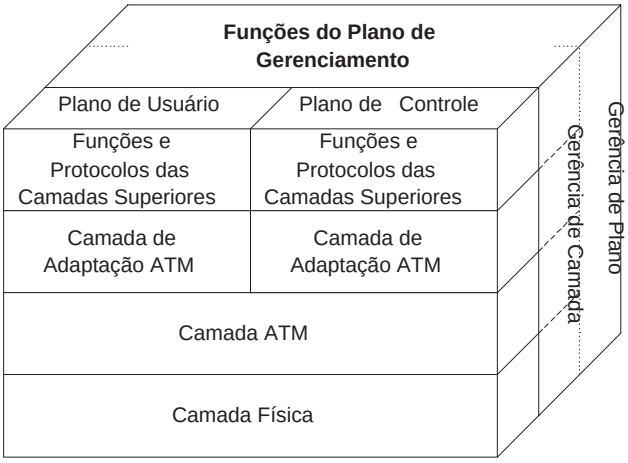

Figura 4.8: Modelo de Referência de Protocolos da RDSI-FL

[3] *AAL - ATM Adaptation Layer*

O plano do Usuário provê o transporte de dados do usuário e de controle associados a este transporte, tais como controle de fluxo e recuperação de erros. O plano de Controle provê funções de controle e sinalização necessárias para o estabelecimento de uma conexão comutada, além de roteamento e endereçamento. Os planos de Usuário e Controle compartilham as camadas Física e ATM, mas utilizam diferentes AALs (*Adaptation ATM Layer*) e funções de alto nível (figura 4.9).

Figura 4.9: Plano do Usuário e de Controle

O plano de Gerenciamento possui funções de gerência de camadas e de planos. As funções de gerência de planos são relativas ao sistema como um todo e à coordenação entre os planos. Já as funções de gerência de camadas tratam dos fluxos de informação de Operação e Manutenção (OAM - *Operation and Maintenance*) específicas de cada camada. O plano de Gerenciamento e as funções OAM serão melhor detalhados no capítulo 5.

4.7.1 Camada Física

A camada Física é responsável pela transmissão de células entre duas camadas ATM adjacentes por um enlace físico específico. A camada ATM é independente do tipo de camada Física, podendo operar sobre uma grande variedade de enlaces físicos. A tabela 4.1 mostra algumas das interfaces da camada Física que foram padronizadas.

Alguma das interfaces, como a SONET (*Synchronous Optical NETwork*), utilizam transmissão de quadros. O princípio de transmissão de quadros foi desenvolvido com sistemas de multiplexação com divisão por tempo. Blocos de bits são transmitidos a cada 125 μs. O número de bits depende da velocidade do enlace de transmissão.

Um quadro consiste de um *payload* e de bits de overhead. O *payload* leva informações do usuário. Já os bits de *overhead* são usados para determinar o início e o fim de um quadro e para carregar informações de sinalização e de manutenção.

Formato	**Taxa**	**Meio Físico**	**Distância**	**Sistema**	**UNI**
DS-1 (T-1)	1.544	Par Trançado	1km	PDH	Pública
DS-3 (T-3)	44.736	Coaxial	300m	PDH	Pública
STS-1	51.840	SMF	15km	SONET	Pública
	51.840	SMF,MMF	2km,2km	SONET	Privada
	51.840	Coaxial,UTP-3	300m,100 m	SONET	Privada
STS-3c (STM-1)	155.520	SMF	15km	SONET/SDH	Pública
	155.520	SMF,MMF	2km,2km	SONET/SDH	Privada
	155.520	UTP-3, UTP-5	100m	SONET/SDH	Privada
STS-12 (STM-4c)	622.080	SMF	15km	SONET/SDH	Pública
	622.080	SMF,MMF	2km,300m	SONET/SDH	Privada
Células	25.6	UTP-3	100m	Clear Channel	Privada
	155.52	SMF	2 km	Clear Channel	Privada

Tabela 4.1: Interfaces da Camada Física

Como um quadro é transmitido a cada 125 μs, mesmo que não exista informações do usuário, a transmissão de quadros é também conhecida como síncrona. Mesmo parecendo contrário às características do ATM, o sincronismo na camada Física não impõe nenhuma restrição à natureza assíncrona do ATM, que está relacionada ao comportamento das aplicações que fazem as células chegarem em intervalos irregulares à camada Física.

Outra estrutura de transmissão utilizada em ATM é a *Clear Channel Transmission*. Nesse mecanismo, não há estruturas de quadros. Células são transmitidas assincronamente no meio físico. A delimitação da célula é feita pelo HEC.

Como definido na Recomendação I.321 do ITU-T, a camada Física está dividida em duas sub-camadas:

- **PMD** (*Physical Medium Dependent*): A sub-camada PMD vê as células ATM como um fluxo contínuo de bits, com informação de sincronização. A PMD insere e extrai essas informações, além de realizar as codificações necessárias para a transmissão da célula no meio físico. As funções de transmissão e recepção na PMD são, respectivamente, a inserção e reconhecimento de sinais elétricos (num cabo) ou ópticos (numa fibra).

- **TC** (*Transmission Convergence*): A TC fica sobre a PMD e sob a camada ATM. Na transmissão, a TC recebe células ATM, empacota-as em um quadro (caso a transmissão síncrona seja usada), calcula o campo HEC e envia o quadro para a PMD. Já na recepção, as células são extraídas do quadro e checadas usando o HEC, sendo corrigidas quando possível (erros em um único bit). Se não for possível corrigir o cabeçalho, a célula é descartada. Ainda no caso de transmissão síncrona, a TC

também realiza a desassociação da taxa de células. Isto é, a TC insere células vazias para que o fluxo de células entre a camada Física e a ATM se iguale com a velocidade do meio de transmissão.

A figura 4.10 resume as funções da camada Física.

Camada Física	**TC**	Desassociação da Taxa de Células Geração e Verificação do HEC Delimitação das Células no Frame de Transmissão Adaptação do Frame de Transmissão Geração e Recuperação do Frame de Transmissão
	PMD	Sincronização dos Bits Acesso ao Meio Físico

Figura 4.10: Funções da Camada Física

4.7.2 Camada ATM

A camada ATM, independente da camada Física abaixo e da AAL acima, é responsável pelas funções que envolvem o cabeçalho da célula, com exceção do campo HEC que é tratado na camada Física. No lado transmissor, 48 bytes vindos da AAL são recebidos pela camada ATM que gera o cabeçalho apropriado. Depois, a célula é passada para a camada Física para transmissão. Nos nós intermediários até o destino, a camada ATM é responsável pela comutação dessas células, realizando a tradução do VPI e/ou VCI, onde for estabelecido. Na camada ATM do destino, o cabeçalho é removido e o *payload* é repassado para a AAL.

Outras funções executadas pela camada ATM são a multiplexação e a demultiplexação das células. Na direção de transmissão, células de caminhos ou circuitos virtuas individuais são combinadas em um fluxo não contínuo de células, que é passado para a camada Física. Isso possibilita a multiplexação sobre um único enlace físico. Na direção de recepção, o fluxo de células é demultiplexado em caminhos ou circuitos vituais individuais, a partir do conteúdo dos campos VPI e VCI.

A camada ATM tem a capacidade de discriminar células a partir de alguns campos no cabeçalho e dar tratamento especial a essas células. O PTI permite que a camada ATM diferencie células de usuários e de gerência. Alguns determinados valores de VPI-VCI têm significado especial. Por exemplo, células com VPI=0 e VCI=5 levam informação de sinalização da UNI.

Controle de tráfego e de congestionamento também são realizados pela camada ATM. O objetivo é suportar a QoS de cada conexão por toda sua duração e, ao mesmo tempo, proteger a rede de congestionamento. Algumas funções de controle de tráfego são:

- CAC (*Connection Admission Control*): Ações realizadas durante o estabelecimento de conexão para determinar se um pedido de conexão deve ser aceito ou rejeitado.
- UPC (*Usage Parameter Control*): Ações realizadas pela rede para monitorar e controlar o tráfego submetido pelo usuário.
- PC (*Priority Control*): Opção para habilitar o descarte de células com CLP=1, caso haja congestionamento.
- Descarte de Quadros: Opção para redes congestionadas descartarem todas as células associadas a um quadro, ao invés de células individuais.

A camada ATM também possui funções de gerência que serão detalhadas no capítulo 5. A figura 4.11 resume as funções da camada ATM.

Camada ATM	Controle de Fluxo Genérico Geração e Extração do Cabeçalho da Célula Tradução do VPI/VCI da Célula Multiplexação e Demultiplexação de Células

Figura 4.11: Funções da Camada ATM

4.7.3 Camada AAL

A camada ATM lida com funções relacionadas ao cabeçalho da célula, independentemente do tipo de informação carregada no *payload*. Essa simplificação é necessária para seus protocolos se manterem no mesmo nível dos enlaces de alta velocidade. Assim, vários serviços necessários pelas aplicações são deixados de fora. Na camada ATM, por exemplo, não existe detecção de perda de células ou tratamento da variação do retardo das células (CDV - *Cell Delay Variation*).

A principal razão para essas funções estarem fora da camada ATM é porque nem todos os serviços são necessários para todas as aplicações. Por exemplo, para o tráfego de dados, o CDV é irrelevante mas erros não podem acontecer; ao contrário do tráfego de voz no qual erros em bits são admissíveis, mas variação no atraso pode causar problemas de compreensão.

Assim, a camada AAL é usada para incrementar os serviços fornecidos pela camada ATM, suportando as funções necessárias pelas camadas superiores que variam de acordo com os requisitos das aplicações. Então, a AAL é dependente de serviço. Na AAL, as PDUs das camadas superiores são mapeadas em *payloads* de células ATM e vice-versa.

Contudo, como ATM foi projetado para suportar diferentes tipos de serviços, uma única AAL não é suficiente para atender os requerimentos de todas aplicações. Portanto, uma AAL diferente foi projetada para cada classe de serviço. O ITU-T classifica os serviços RDSI-FL baseando-se em três parâmetros:

- Sincronização entre origem e destino (necessária ou não)
- Taxa de transferência (constante ou variável)
- Modo de conexão (orientado à conexão ou não).

A tabela 4.2 apresenta as classes de serviços relacionadas com suas características e as respectivas AALs. Além destas quatro classes, foram definidas mais duas: a classe X e a Y. Na classe X, a AAL (normalmente proprietária), o tipo de transmissão e a necessidade de sincronização são definidas pelo usuário. Essa classe também pode ser usada por aplicações que acessem diretamente a camada ATM, sem nenhum serviço de AAL. A classe Y foi projetada para aplicações de dados sem restrições de atraso, de modo que as características da conexão (largura de banda máxima ou mínima, por exemplo) possam ser alteradas depois do seu estabelecimento.

Parâmetro	**Classe A**	**Classe B**	**Classe C**	**Classe D**
Sincronização entre Origem e Destino	Necessária	Necessária	Não Necessária	Não Necessária
Taxa de Transmissão	Constante	Variável	Variável	Variável
Modo de Transmissão	Orientado a Conexões	Orientado a Conexões	Orientado a Conexões	Não Orientado a Conexões
Tipo de AAL	AAL1	AAL2	AAL3/4, AA5	AAL3/4

Tabela 4.2: Classificação dos Serviços AAL

A estrutura da AAL, independente do seu tipo, é apresentada na figura 4.12. A AAL é dividida em duas sub-camadas: a SAR (*Segmentation and Reassembly*) e a CS (*Convergence Sublayer*). A SAR lida com a segmentação em *payloads* de células ATM e a remontagem das unidades de dados das camadas superiores. A CS fornece o serviço AAL no SAP (*Service Access Point*) e contém funções específicas a um determinado serviço. Cada AAL-SAP está associada a um conjunto de parâmetros de QoS. A CS é ainda

subdividida em duas: a CPSP (*Common Part Convergence Sublayer*), sempre presente; e a SSCS (*Service Specific Convergence Sublayer*), específica a cada classe de serviço e podendo nem estar presente.

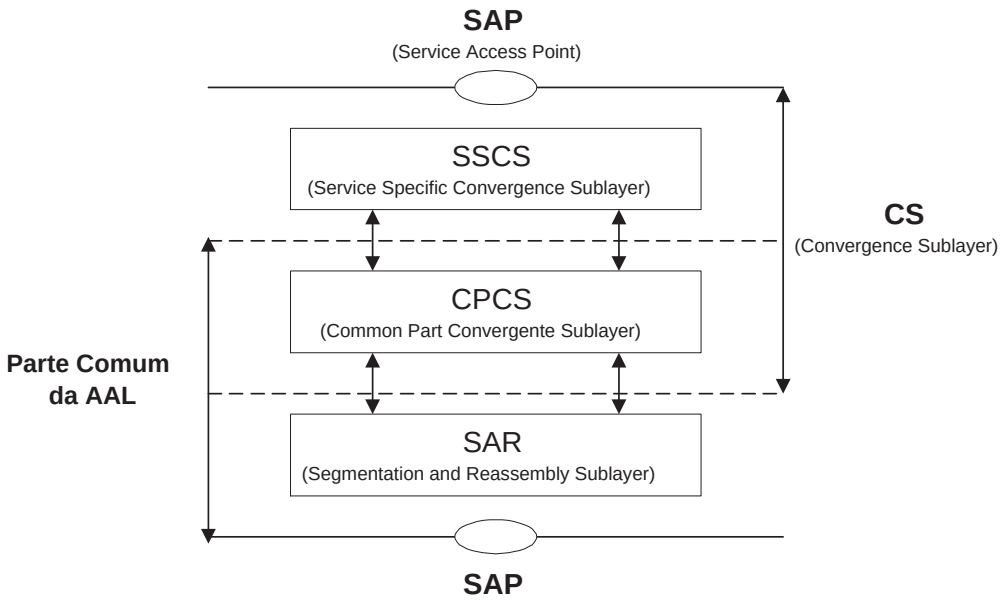

Figura 4.12: Estrutura da Camada AAL

SAAL

A SAAL (*Signaling AAL*) oferece mecanismos para o transporte de tráfego de sinalização entre usuários ATM. Como parte do Plano de Controle, a SAAL serve como interface entre as funções de sinalização de alto nível (como a UNI 3.1/Q.2931) e a camada ATM.

A SAAL utiliza a CPCS e a SAR oferecidas pela AAL 5, como mostrado na figura 4.13. A diferença está na SSCS da SAAL, que contém duas funções:

- **SSCOP** (*Service Specific Connection-Oriented Protocol*): é um protocolo de enlace que oferece transporte de mensagens de sinalização. A SSCOP suporta detecção e correção de erros fim-a-fim, senquenciação e recuperação seletiva de quadros.
- **SSCF** (*Service Specific Coordination Function*): responsável pelo mapeamento da aplicação da camada superior para a SSCOP.

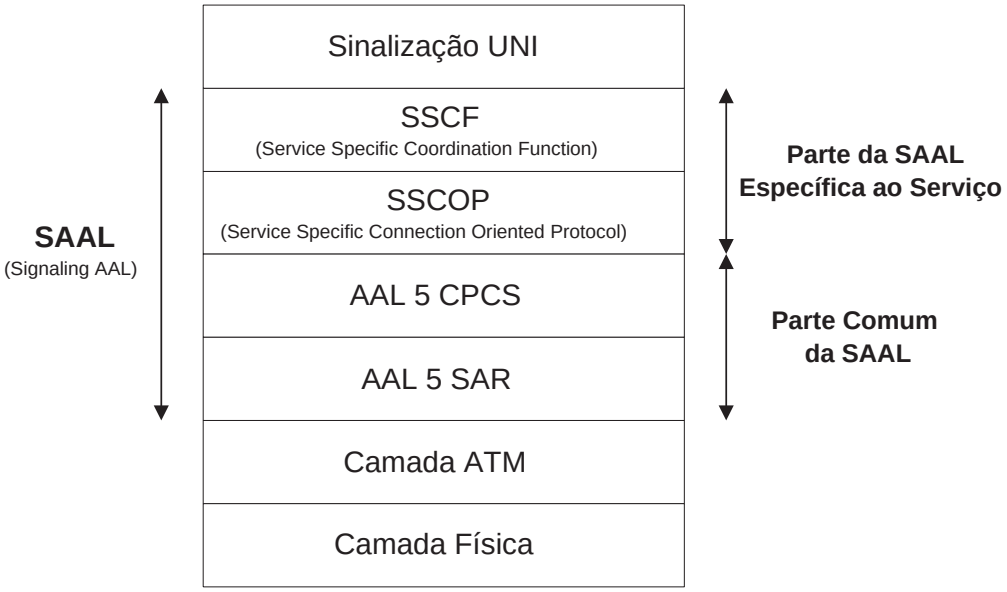

Figura 4.13: Estrutura da SAAL

4.8 Endereçamento ATM

Cada estação ATM necessita de um endereço ATM que a identifique unicamente. Endereços ATM são necessários para que as redes ATM possam localizar os nós origem e destino de uma conexão.

As redes públicas e privadas utilizam diferentes formatos de endereço. As redes públicas fazem uso de endereços no formato E.164 do ITU-T. Já as redes privadas utilizam o formato NSAP (*Network Service Access Point*) da OSI. [3].

Endereços NSAP são baseados no conceito de domínios hierárquicos de endereçamento. Cada endereço tem 20 bytes e possui duas partes, uma chamada IDP (*Initial Domain Part*) e outra chamada DSP (*Domain Specific Part*). O IDP especifica um sub-domínio do espaço de endereçamento global (indicada pelo campo IDI - *Initial Domain Identifier*) e a autoridade responsável por atribuir endereços naquele sub-domínio (indicada pelo campo AFI - *Authority and Format Identifier*).

A parte da rede tem 13 bytes e deve ser configurada pelo administrador da rede em cada porta do comutador. A parte da estação é dividida entre 6 bytes para o identificador da estação (ESI - *End System Identifier*) e 1 byte para o campo *Selector*, que indica cada dispositivo ligado à estação, caso haja algum. O ESI pode ser o endereço MAC (*Medium Access Control*) da estação.

4.9 LAN Emulation

LAN Emulation (LANE) é um serviço desenvolvido no ATM Forum para permitir que aplicações LAN já existentes executem sobre uma rede ATM. Para isso, esse serviço deve emular as características e o comportamento de redes locais convencionais. Assim, um serviço não orientado à conexão deve ser suportado. Tráfego *broadcast* e *multicast* também devem ser permitidos. A LANE deve possibilitar a interconexão de LANs tradicionais com LANs emuladas, mantendo o endereço MAC de cada dispositivo da LAN, para deste modo, preservar a imensa base de aplicações LAN existentes permitindo que funcionem sem alterações sobre uma rede ATM. Esta é a maior vantagem da LANE e sua maior desvantagem pois as aplicações não utilizam às características de garantia de Qos do ATM.

A LAN emulada pode ser Ethernet (IEEE 802.3) ou Token Ring (IEEE 802.5). A LANE define para cada instância de LAN emulada um conjunto de serviços: configuração, resolução de endereços (MAC para ATM) e *broadcast*.

A participação numa LAN emulada não é determinada pela localização física do dispositivo e sim pela associação lógica com o conjunto de serviços. Por isso, LANE é ideal para a construção das chamadas *LANs Virtuais* (VLANs).

Numa LAN emulada, também é possível o acesso direto à rede ATM permitindo que duas estações estabeleçam um caminho virtual e tirem proveito da largura de banda e garantias de QoS oferecidas pelo ATM. Contudo, a maioria das aplicações devem executar utlizando os serviços da LANE, sem garantias de QoS.

Arquitetura da LANE

A figura 4.14 mostra as camadas funcionais da arquitetura da LANE. As camadas Física e ATM são as usuais. LANE usa serviços AAL tradicionais para as transferências de dados de usuário (AAL 5) e de sinalização (SAAL). A camada LANE apresenta para a camada superior uma interface ao nível de MAC e um esquema de endereçamento, com o objetivo de parecer às aplicações estarem, para todos os efeitos, executando realmente sobre uma LAN.

Clientes e servidores LANE interagem sobre a LUNI (*LANE User-Network Interface*). As funções executadas sobre a LUNI são: inicialização, transferências de dados, registro e resolução de endereços.

Componentes da LANE

Os componentes de um LAN emulada são definidos assim:

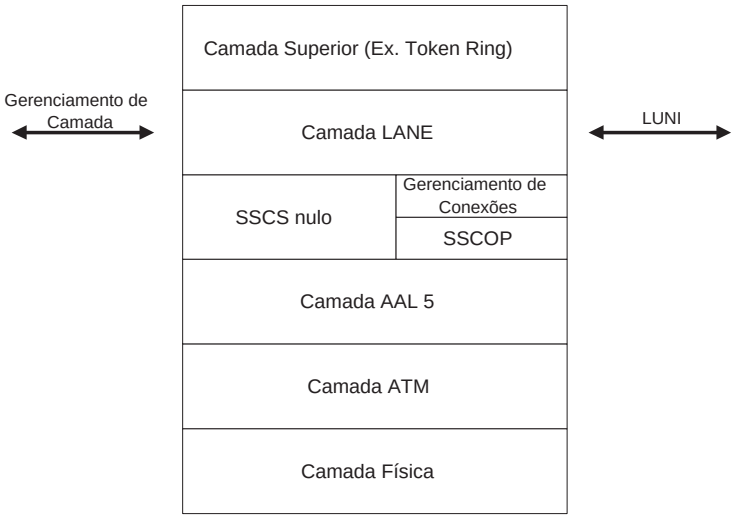

Figura 4.14: Estrutura da LANE

- **LEC** (*LAN Emulation Client*): Um LEC pode ser uma estação, uma ponte ou um roteador. O LEC faz resolução de endereços, repasse de dados para outros LECs e outras funções de controle. O LEC apresenta uma interface de nível MAC e implementa numa LUNI comunicação com outros componentes de uma LAN emulada.

- **LES** (*LAN Emulation Server*): O LES age como servidor de registro e resolução de endereços, sendo dedicado a uma LAN emulada. O LES oferece meios para que LECs registrem seus endereços MAC e ATM. Um LEC pode pedir ao LES a resolução de um endereço MAC para ATM. O LES pode responder diretamente esse pedido ou repassá-lo para outros LECs.

- **LECS** (*LAN Emulation Configuration Server*): O LECS é usado para inicializar um LEC com informações específicas da LAN emulada para qual o LEC está entrando. Por exemplo, o LECS fornece o endereço ATM do LES para o LEC (de acordo com seu endereço MAC ou ATM).

- **BUS** (*Broadcast and Unknown Server*): O BUS trata quadros para o endereço de *broadcast* MAC, todo tráfego *multicast* e quadros *unicast* enviados por um LEC antes que o endereço ATM do destino seja resolvido. Todos LECs mantêm uma conexão com o BUS e são destinos de um VCC ponto-a-multiponto que tem o BUS como origem. Isso permite que os LECs enviem quadros sem estabelecer uma conexão primeiro.

Os servidores (LES, LECS e BUS) podem executar numa única máquina ou em máquinas separadas. Uma LAN emulada só precisa de um LES e de um BUS, mas

que só podem executar um tipo de LANE (Ethernet ou Token Ring).

Conexões da LANE

Mensagens de controle e quadros de dados trafegam em VCCs diferentes. VCCs de Controle levam mensagens de um LEC para outro LEC, para um LES ou um LECS. Já os VCCs de Dados são estabelecidos entre LECs ou de um LEC para o BUS.

VCCs de Controle:

- VCC de Configuração Direta: VCC bidirecional ponto-a-ponto entre o LEC e o LECS. É usado pelo LEC para obter informações de configuração, como o endereço ATM do LES.
- VCC de Controle Direto: VCC bidirecional ponto-a-ponto entre o LEC e o LES para o envio de informações de controle.
- VCC de Controle Distribuído: VCC unidirecional ponto-a-ponto ou ponto-a-multiponto que é opcionalmente estabelecido do LEC para um ou mais LECs, durante a fase de inicialização.

VCCs de Dados:

- VCC de Dados Direto: VCC bidirecional ponto-a-ponto estabelecido entre dois LECs. Quando um LEC (origem) deseja se comunicar com outro (destino), mas não sabe seu endereço ATM, o LEC origem faz um pedido de resolução de endereços para o LES que é mandado pelo VCC de Configuração Direta. Assim que a resposta é recebida, o LEC origem pode estabelecer um VCC de Dados Direto com o LEC destino.
- VCC de Envio Multicast: VCC bidirecional ponto-a-ponto estabelecido entre o LEC e o BUS para o envio de quadros de dados *multicast* e *unicast* com endereço ATM do destino desconhecido pelo LEC. Um LEC pode receber quadros de dados sobre esse VCC.
- VCC de Repasse Multicast: Pode ser um VCC unidirecional ponto-a-ponto ou ponto-a-multiponto que é estabelecido do BUS para um ou mais LECs. Esse VCC é usado para o repasse de quadros de dados *multicast* para os membros da LAN emulada.

4.10 IP sobre ATM

Visando permitir o suporte direto do IP sobre a AAL5, o IETF especificou um mecanismo conhecido como IP sobre ATM, definido na RFC 1577 [40]. A imensa base de estações utilizando IP deve ter meios de migração para o ATM. Para isso, o IP deve ser adaptado para suportar aplicações multimídia.

A política de dividir as redes em subredes de acordo com domínios administrativos e de trabalho é utilizada nesta abordagem. Assim, uma rede ATM pode ser vista como várias LIS (*Logical IP Subnetwork*). Uma LIS é um conjunto de estações e roteadores ATM conectados, dentro de uma subrede IP comum. A figura 4.15 [41] mostra uma rede ATM dividida em diversas LIS.

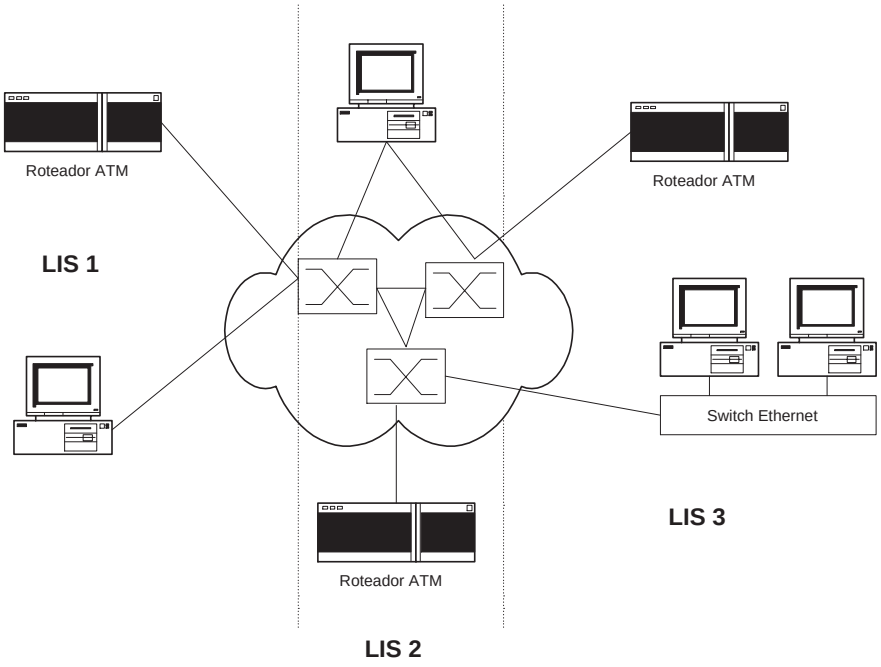

Figura 4.15: IP Sobre ATM

Cada LIS contém um servidor de ARP (*Address Resolution Protocol*), responsável pelo mapeamento de endereços IP em ATM e vice-versa. Todos os membros da LIS são registrados com o servidor de ARP. Todos os pedidos de resolução de endereços a partir de membros da LIS são tratados pelo servidor, que pode ser um processo executando no roteador.

Os pedidos de resolução de endereços IP são passados para o servidor de ARP. Quando a estação que originou o pedido recebe o endereço ATM correspondente, ela pode estabelecer uma conexão direta com esta máquina. Duas grandes mudanças foram realizadas

no protocolo tradicional ARP: a criação da mensagem ATMARP (para pedir resolução de endereços) e da mensagem InATMARP (para registro inverso de endereços).

O modelo IP sobre ATM é mais simples que o modelo LANE, contudo, menos funcional. No IP sobre ATM, cada estação deve ser configurada manualmente com o endereço do servidor de ARP, diferentemente da LANE onde há descoberta dinâmica. Uma desvantagem é a possibilidade que duas estações na mesma rede ATM física, tenham que se conectar através de um roteador por estarem em diferentes LIS. Esta restrição pode diminuir o *throughput* da rede e aumentar o retardo. Uma vantagem do modelo IP sobre ATM é o tamanho da MTU (*Maximum Transfer Unit*) ser de 9180 bytes. Uma MTU grande pode aumentar o desempenho de estações ligadas a uma rede ATM [41].

4.11 Aplicações de Redes Neurais em ATM

Além das características comentadas no capítulo 3, as redes neurais podem ser implementadas em hardware, que incorpora estruturas altamente paralelas ou analógicas utilizando técnicas VLSI (*Very Large Scale Integration*). Tais características têm motivado a investigação do uso de redes neurais em redes ATM.

Vários trabalhos aplicam redes neurais ao problema de Gerenciamento de Recursos (RM - *Resource Management*) de uma rede ATM. RM é realizado em dois níveis. No nível mais baixo, existe a necessidade de mecanismos de controle on-line que evitem congestionamentos na rede a curto e longo prazo, dada a topologia da rede virtual. Em um nível mais alto, um conjunto de mecanismos é necessário para determinar a topologia da rede virtual a partir da física [42].

No nível mais baixo, a tarefa de admitir conexões garantindo a Qualidade de Serviço (tarefa também conhecida como CAC - *Connection Admission Control*) tem recebido atenção especial. No trabalho [43], foi proposto um controlador ATM que usa uma rede neural *back-propagation* para aprender as relações entre o tráfego oferecido e a QoS. Já o trabalho [44] propõe um esquema de unidades hierárquicas, onde cada unidade é uma rede *feed-forward*. A rede neural calcula a largura de banda necessária através de medidas on-line do tráfego. O método neural obteve resultados melhores que as técnicas convencionais. Já no nível mais alto do RM, podemos citar o trabalho [45], que trata do problema de otimização do dimensionamento de sub-redes virtuais atráves de uma CPNN (*Convex Programming Neural Network*).

Entre outras aplicações de redes neurais em ATM, podemos ainda citar: predição de tráfego multimídia [46][47][48], roteamento *multicast* [49] e controle de fluxo [50].

Capítulo 5

Gerência de Redes ATM

A tecnologia ATM é caracterizada pela integração, desde seu surgimento, de técnicas e infra-estrutura destinadas ao monitoramento e controle de seus componentes. Quanto à infra-estrutura, trata-se do Plano de Gerência do Modelo de Arquitetura nos Protocolos RDSI-FL. Provavelmente, esta característica adicional na arquitetura ATM, que privilegia a questão da gerência, deve-se a uma maior complexidade desta tecnologia, necessária ao tratamento eficiente de vários tipos de mídia.

Devido às suas especificidades, gerência ATM exige uma organização diferente daquela utilizada na gerência de pacotes. Em [1], é apresentada uma proposta que melhor organiza esses conceitos:

1. Elementos de Interesse à Gerência (EIGs) semelhantes a Objetos Gerenciados SNMP
2. EIG resultado da interação de dois blocos funcionais em uma rede ATM (ex.: comutador-comutador, comutador-estação)
3. EIG intrínseco à arquitetura de protocolos RDSI-FL, portanto sem similaridades com Objetos Gerenciados SNMP

A opção 1 acima pode ser tratada, com limitações, pelo modelo de gerência convencional. Afinal, existem EIGs numa rede ATM que podem ser modelados como Objetos Gerenciados em uma MIB (*Management Information Base*) e acessados pelo protocolo de gerenciamento, utilizando o paradigma Gerente-Agente. Esta opção será vista na seção 5.1 denominada *MIBs SNMP Aplicadas ao ATM*.

Já a opção 2 refere-se a EIGs associados às diversas interfaces existentes, em particular à interface UNI. Ela também se baseia no modelo Gerente-Agente, com um diferencial importante: uma série de procedimentos definidos e que funcionam independentemente dos mecanismos de gerência existentes na rede em questão. Esta opção será ilustrada na seção 5.2 denominada *ILMI*.

A opção 3 é um mecanismo de gerência intrínseco à arquitetura ATM. Trata-se de um conjunto de funções de Operação e Manutenção recomendadas pelo ITU-T [51]. Esta opção é abordada na seção 5.3 denominada *Funções OAM* (*Operations and Maintenance*).

Visando classificar as especificações para gerência ATM, o ATM Forum especifica um modelo de referência que define cinco interfaces de gerenciamento, denominadas de M1 a M5, como ilustrado na figura 5.1. A interface M1 suporta várias funções de gerenciamento de dispositivos ATM. M2 define funções de gerência de rede ATM privada, enquanto M4 é especificada para redes públicas. A interface M3 permite que dois sistemas de gerenciamento, um de rede pública e outro de privada, se comuniquem. A interface M3 permite ainda que o cliente de uma rede pública supervisione sua porção da rede ATM, através de um agente *Customer Network Management* (CNM), que se baseia no protocolo SNMP e suporta as mesmas MIBs (*Management Information Base*) das interfaces M1 e M2. Similarmente à M3, a interface M5 especifica a comunicação entre dois sistemas de gerência de rede pública [3].

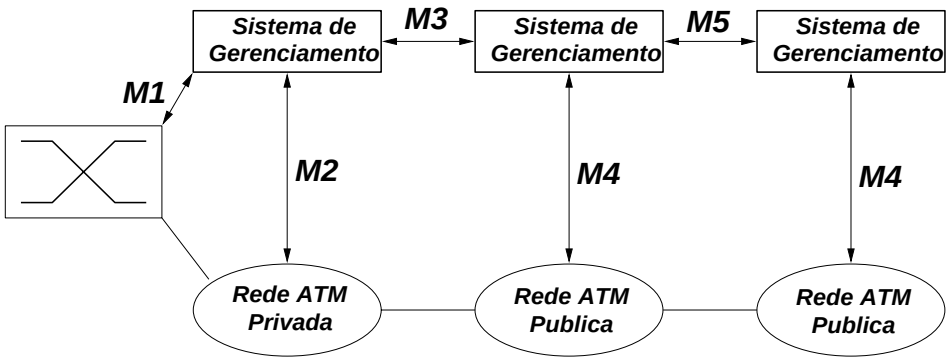

Figura 5.1: Modelo das 5 Interfaces

5.1 MIBs SNMP Aplicadas ao ATM

Com a popularização da Internet e, por conseqüência, de todas os protocolos da família TCP/IP, é natural que o SNMP tenha sido a primeira tentativa de se integrar toda a infra-estrutura disponibilizada pelo Plano de Gerência em um sistema de gerência adequado ao ambiente ATM. A despeito de suas limitações, o SNMP tem sido adaptado e estendido para suportar produtos baseados na tecnologia ATM.

Várias organizações, como o ATM Forum e o IETF, têm proposto MIBs SNMP que modelam desde aspectos gerais a específicos de determidado meio de transmissão (ex: SONET MIB) ou de determinados ambientes (ex: LANE MIB). Dentre as mais importantes, destacam-se a AToM MIB e as extensões da RMON MIB para ATM [52].

5.1.1 AToM MIB

O IETF definiu a AToM MIB [53], que especifica objetos gerenciados para as interfaces M1, M2 e partes da M3. A AToM MIB permite o gerenciamento de interfaces ATM, circuitos virtuais, cruzamento de conexões, entidades e conexões AAL5, dando enfoque maior aos PVCs (*Permanent Virtual Connections*). O gerente obtém informações de gerenciamento através dos agentes utilizando SNMPv2 sobre UDP/IP. Tentando suprir algumas deficiências da AToM MIB, a IETF lançou posteriormente a MIB Suplementar. Tal MIB contém objetos específicos a SVCs (*Switched Virtual Connections*), endereçamento, sinalização, portas lógicas e testes. Estes últimos permitem a ativação de fluxos de células OAM (seção 5.3) para realizar testes de loopback em conexões [54].

A figura 5.2 mostra a estrutura da AToM MIB. Suas informações podem ser divididas em três categorias: informações sobre interfaces, informações sobre circuitos virtuais e informações sobre cruzamento de conexões. Cada uma será detalhada a seguir.

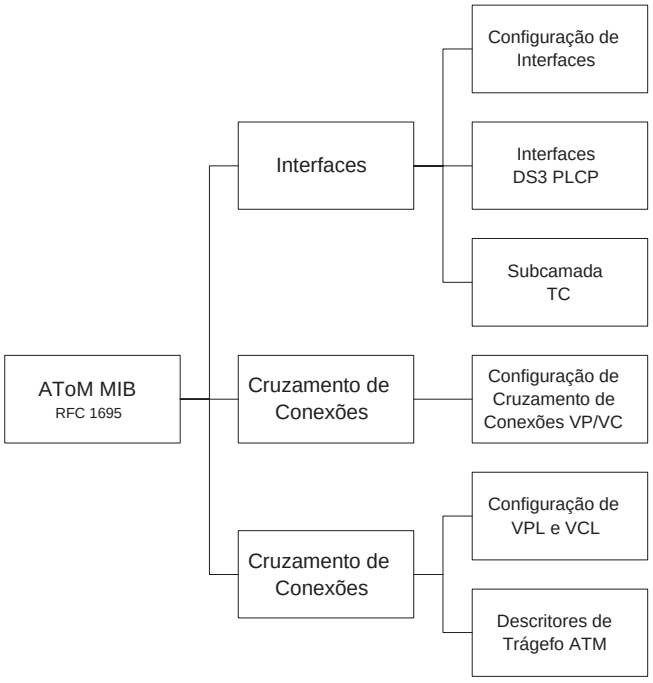

Figura 5.2: Estrutura da AToM MIB

Informações sobre Interfaces

Como visto na figura 5.2, as informações sobre interfaces estão divididas em três grupos: informações sobre configuração de interfaces, informações sobre interfaces DS3 e sobre a

subcamada TC (*Transmission Convergence*) da camada Física.

- **Configuração de Interfaces (atmInterfaceConfTable)**

  Este grupo contém informações de configuração sobre as interfaces ATM, além das encontradas na tabela de interfaces ifTable da MIB II (seção 1.4.2). Cada entrada se relaciona com uma interface ATM presente no dispositivo e é composta de parâmetros de configuração como: endereço ATM da interface, número máximo de VPCs e VCCs suportados e número de VPCs e VCCs configurados, entre outras informações.

- **Interfaces DS3 PLCP (atmInterfaceDs3PlcpTable)**

  Este grupo é também uma extensão da tabela ifTable (MIB-II) e contém parâmetros de configuração e de estado do DS3 PLCP. DS3 é um tipo de interface de camada Física (tabela 4.1) e o PLCP (*Physical Layer Convergence Procedure*) é um método para transportar células sobre PDH.

- **Subcamada TC (atmInterfaceTCTable)**

  Este grupo possui parâmetros de estado e de configuração da subcamada TC. Também é uma extensão da tabela ifTable. Cada entrada se relaciona com uma interface que utiliza a subcamada TC para transportar células. As entradas possuem dois atributos que contém informações sobre a existência de problemas na delineação de células. Exemplos destas interfaces são aquelas que possuem como camada física SONET ou DS3.

Informações sobre Circuitos Virtuais

As informações sobre circuitos virtuais estão divididas naquelas referentes à configuração de VPLs e VCLs e aos descritores de tráfego ATM.

- **Configuração de VPLs e VCLs (atmVplTable e atmVclTable)**

  Estas tabelas são definidas em terminais e comutadores e podem ser utilizadas para se criar, atualizar ou liberar um VPL/VCL. Os parâmetros comuns à estas tabelas são: identificador de caminho virtual, que é o VPI para conexões VPL e a combinação de VPI e VCI para VCL; atributos que caracterizam o estado do circuito (administrativo e operacional). Em geral, os outros parâmetros são índices para a tabela de descrição de tráfego, e um índice para a tabela de cruzamento de conexões.

- **Descritores de Tráfego ATM (atmTrafficDescrParamTable)**

  Este grupo possui um conjunto de parâmetros que caracterizam o tráfego ATM, incluindo a classe da Qualidade de Serviço (QoS). Esta tabela é definida tanto em terminais como em comutadores. Cada entrada descreve o tráfego que é transportado sobre um circuito virtual. Os descritores de tráfego estão de acordo com os resultados da negociação, quando do estabelecimento da conexão. Quando é criada uma nova entrada nesta tabela, seus parâmetros são checados para garantir a consistência.

Informações sobre Cruzamento de Conexões

As informações de cruzamento de conexões estão em duas tabelas, comentadas a seguir.

- **Configuração de Cruzamento de Conexões VP/VC (atmVpCrossConnectTable e atmVcCrossConnectTable)**

  Estes grupos descrevem informações sobre estado e configuração de todos os cruzamentos de conexão VP/VC relacionados com PVC (Permanent Virtual Circuit), sejam eles ponto a ponto, ponto a multiponto e multiponto a multiponto. Estas informações são disponibilizadas somente em comutadores, onde temos a funcionalidade de cruzamento de conexões.

  Um conjunto de entradas nesta tabela representa o cruzamento de conexões VPC/VCC bidirecionais. Para uma conexão ponto a ponto temos uma entrada; ponto a mutiponto com 'N' nós folhas temos 'N' entradas; e para conexões multiponto a multiponto entre 'N' nós temos, N(N-1)/2 entradas. Cada uma dessas entradas referencia duas entradas nas tabelas de circuitos virtuais (VPL e VCL).

5.1.2 MIBs SNMP e o Modelo das Cinco Interfaces

A tabela 5.1 mostra o mapeamento de diversas MIBs ATM (inclusive MIBS CMIP) para o modelo das Cinco Interfaces, apresentado no trabalho [55].

As interfaces M1 e M2 (figura 5.1) possuem suas especificações baseadas no SNMP, em virtude deste protocolo ser amplamente empregado atualmente. Gerentes de uma rede privada estarão particularmente interessados nas interfaces M1 e M2. Estas interfaces utilizam MIBs padronizadas pelo IETF e ATM Forum, tais como: MIB II, MIBs padrões para conexões (DS-1, DS-3 e SONET), AToM MIB e ILMI MIB.

Interface de Gerenciamento	MIBs Aplicáveis
M1	AToM MIB, LANE MIB, DXI MIB, MIBs Proprietárias
M2 (SNMP)	AToM MIB, LANE MIB, PNNI MIB, MIBs de Transmissão (RFC 1406, RFC 1595), IMA, RMON MIB, MIBs Proprietárias
M2 (CMIP)	M4 Network View MIB, M4 Network Element MIB, M2 SVC MIB, ITU-T SONET MIB e E1/E3 MIBs
M3	M3 MIB e AToM MIB
M4 (SNMP)	ILMI MIB, LANE MIB, CES MIB, M4 SNMP MIB, ATM AAL MIB, MIBs de Transmissão (RFC 1406, RFC 1595), ATM IMA MIB e RMON MIB
M4 (CMIP)	M4 NE MIB, ITU-T I.751 MIB, M4 SVC MIB, ATM AAL MIB, MIBs de Transmissão (ITU-T G.704, G.706, G.774, G.826, G.882), Bellcore G.I114 e NM Forum MIB
M5 (CMIP)	ATM Forum Network-to-Network MIB e ETSI NAS-2212 carrier-to-carrier MIB

Tabela 5.1: Mapeamento das MIBs para o Modelo das Cinco Interfaces

5.2 ILMI

A ILMI (*Integrated Local Management Interface*) [56] foi definida pelo ATM Forum, com o propósito de possibilitar a configuração da interface UNI (*User-Network Interface*) e de fornecer informações de status e controle. A UNI conecta um equipamento ATM, seja um comutador ou um terminal, a uma rede ATM pública ou privada.

A ILMI também define um mecanismo de registro de endereços e serviços pela UNI, além dos procedimentos de conectividade ILMI. Cada UNI de dispositivo possui uma IME (*Interface Management Entity*) e uma instância da ATM Interface MIB (também conhecida como ILMI MIB). As informações de gerência são trocadas entre IMEs adjacentes, que atuam como agente e gerente em relação ao seu vizinho, através do SNMP diretamente sobre a AAL5.

Essas especificações antes se chamavam *Interim Local Management Interface*, porque o ATM Forum tinha a expectativa que essa fosse a solução temporária de gerenciamento enquanto o ITU-T não definisse padrões para a gerência de redes ATM. Como essas expectativas não se concretizaram, a proposta foi confirmada na versão 4.0, como dita *Integrated*.

ILMI define a sua MIB separadamente do seu mecanismo de acesso (figura 5.3). Esta abordagem não requer que existam agentes nos dois lados da UNI. Por outro lado, foram definidas IMEs que acessam as informações da MIB, diretamente sobre um VCC pré-

definido (normalmente, VPI=0 e VCI=16), através de comandos SNMP encapsulados na AAL5. Outro mecanismo de acesso a esta MIB se dá através de sistemas de gerenciamento usando SNMP sobre UDP/IP/AAL5. Como também ilustrado na figura 5.3, ILMI pode ser usado tanto sobre UNIs públicas e privadas.

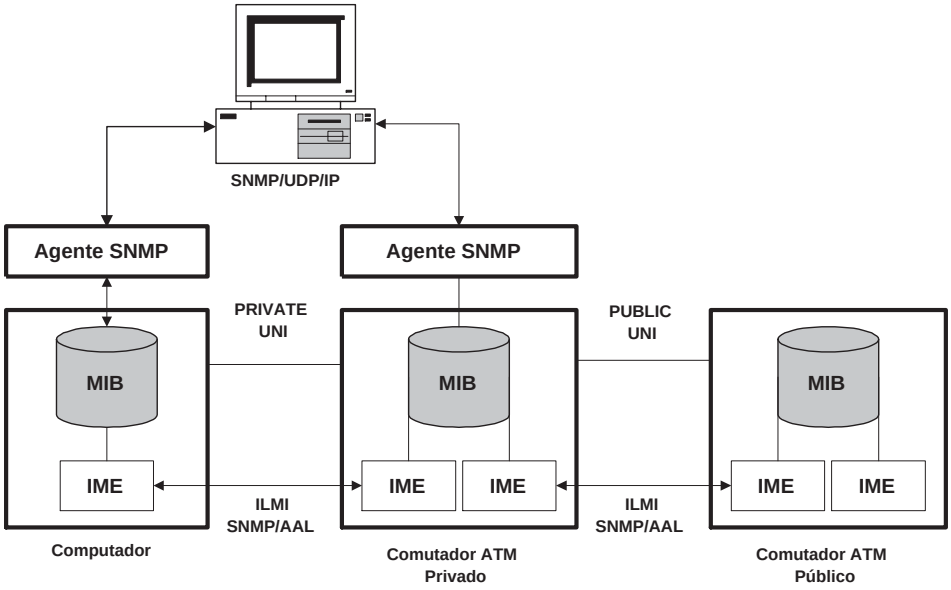

Figura 5.3: Integrated Local Management Interface (ILMI)

5.2.1 ILMI MIB

Também conhecida como ILMI MIB, a ATM Interface MIB possui 4 módulos: *Textual Conventions MIB*, *Link Management MIB*, *Address Registration MIB* e *Service Registry MIB*.

A Textual Conventions MIB define um número de Convenções de Texto (definição de diversos tipos de dados) e Identificadores de Objetos em um único módulo, de modo que outros módulos da ILMI MIB possam importar essas informações de modo consistente. A Link Management MIB provê a facilidade de gerenciamento de interface para todas as interfaces ATM. A Address Registration MIB provê um mecanismo para o registro de endereços em uma UNI. A Service Registry MIB provê um registro geral de serviços para a localização de servidores ATM, como o LECS (*LAN Emulation Configuration Server*) ou o ANS (*ATM Name Server*). A figura 5.4 mostra os tipos de informações existentes na ATM Interface MIB, correspondendo às tabelas da MIB.

A não ser que seja declarado ao contrário para porções específicas da ILMI, cada IME

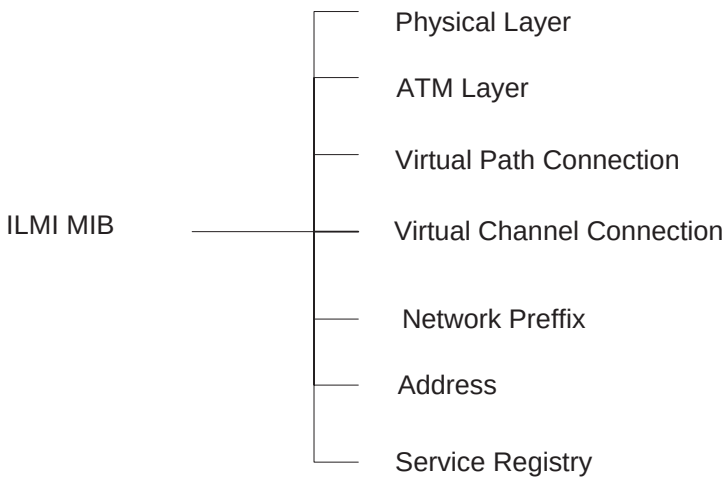

Figura 5.4: Estrutura da ILMI MIB

contém a mesma ATM Interface MIB. Contudo, a semântica de alguns objetos da MIB pode mudar dependendo do papel de cada IME (se rede ou usuário).

Obrigatoriamente, um dispositivo que implemente a ATM Interface MIB deve também implementar o grupo "system" da MIB-II. Deste modo, um IME deve prover acesso ao grupo "system" via ILMI.

Os módulos mais importantes da ILMI MIB são apresentados a seguir: a Link Management MIB, a Address Registration MIB e a Service Registry MIB.

Link Management MIB

- Grupos da Link Management MIB

  atmfPhysicalGroup: Esse grupo contém informações que identificam a interface física gerenciada pela IME e endereços do dispositivo (IP ou NSAP). Alguns objetos dessa MIB (como tipo do meio físico da interface, por exemplo) se tornaram obsoletos na versão 4.0. O ATM Forum considera que esses objetos devem ficar em outras MIBs padrões, como a AToM MIB.

  atmfAtmLayerGroup: Esse grupo contém informações de configuração da camada ATM do dispositivo, presentes em todas as conexões virtuais da interface. Por exemplo: o número máximo de VPCs e VCCS permitidos na interface, o tipo do dispositivo (usuário ou rede), o tipo da UNI (pública ou privada), a versão da UNI e da ILMI, entre outras.

  atmfAtmStatsGroup: Esse grupo se tornou obsoleto na versão 4.0. Ele con-

tinha estatísticas como número de células recebidas, transmitidas e descartadas. Essas informações devem ficar em outras MIBs padrões.

atmfVpcGroup: Esse grupo contém informações de status e parâmetros dos VPCs que passam por essa interface. Por isso, a tabela é indexada pelo ifIndex da MIB-II e pelo VPI da conexão. As informações do grupo descrevem o estado operacional da conexão (desconhecido, *up* fim-a-fim, *up* localmente, *up* local e desconhecido fim-a-fim, *down* fim-a-fim e *down* local), os descritores de tráfego relativos à transmissão e recepção do VPC e à categoria de serviço do VPC.

atmfVpcAbrGroup: Esse grupo de objetos é uma extensão opcional ao atmfVpcGroup, indexada também pelo VPI e pelo ifIndex. Ele contém informações a respeito dos parâmetros operacionais de conexões VPC com tráfego ABR (Available Bit Rate)

atmfVccGroup: Esse grupo contém informações de status e parâmetros dos VCCs que passam por essa interface. Por isso, a tabela é indexada pelo ifIndex e pelo par VPI/VCI da conexão. As informações do grupo são análogas àquelas para VPCs. Adicionalmente, o grupo traz indicadores de descarte de quadros, para o caso de congestionamentos.

atmfVccAbrGroup: Esse grupo de objetos é uma extensão opcional ao atmfVccGroup, indexada também pelo par VPI/VCI e pelo ifIndex. Ele contém informações a respeito dos parâmetros operacionais de conexões VPC com tráfego ABR, análogos aos contidos na atmfVpcAbrGroup.

Procedimentos da Link Management MIB

Juntamente com as MIBs, foram definidos procedimentos de operação na ILMI. Os procedimentos de Conectividade ILMI são usados para detectar a perda e o estabelecimento de conectividade ILMI. Esses procedimentos são obrigatórios para nós ATM (tipo rede) e opcionais para um host ATM (tipo usuário). Esse eventos são utilizados para autoconfiguração e registro de endereços. Para testar ou estabelecer a conectividade, a IME envia uma PDU SNMP GetRequest para os objetos atmfPortMyIfIdentifier, atmfMySystemIdentifier e sysUpTime.

Procedimentos de mudança do ponto de conexão (*attachment point*) podem ser implementados opcionalmente em qualquer IME. Tais procedimentos tratam dos problemas que podem ocorrer durante a sinalização. Os mecanismos de sinalização podem suportar falhas na camada física por um curto período de tempo, antes de liberar as conexões. Durante esse breve espaço de tempo, existe uma possibilidade que dois links sejam trocados, sem a detecção dos mecanismos de sinalização.

Se um ponto de conexão não executa ILMI, o IME local pode interpretar uma mudança no ponto de conexão como uma perda da conectividade ILMI, o que pode ser inaceitável para links seguros. Assim, quando uma perda de conectividade ILMI é detectada, a IME de um link seguro deve informar as entidades de sinalização para liberar todos SVCs controlados por essa entidade, além dos procedimentos de Conectividade ILMI descritos acima. Esses procedimentos de links seguros são opcionais.

O VCC de VPI=0 e VCI=5 é utilizado para sinalização. A ILMI também define procedimentos para configurar automaticamente os parâmetros de transmissão desse VCC. Outro procedimento da ILMI, implementado obrigatoriamente, trata da modificação de atributos locais. Se o valor de alguns atributos, como aqueles da interface da camada ATM, são modificados; o IME deve ser reinicializado para efetuar as mudanças.

Address Registration MIB

Para se estabelecer uma conexão ATM na UNI, o usuário e a rede devem saber os endereços ATM ativos naquela UNI. Esses endereços ATM podem ser usados nas informações do "*Calling Party Number*" das mensagens de sinalização. Os procedimentos de registro de endereços possibilitam a troca dinâmica de informação de endereçamento entre o usuário e a rede, quando da inicialização e em outros momentos. Tanto equipamentos com UNI privada como pública devem suportar mecanismos de registro de endereços.

Como especificado na UNI 4.0, um endereço ATM Privado consiste de diversos campos. Dois desses, *End System Identifier* (ESI) e o *Selector* (SEL), formam a "parte do usuário" e são fornecidos pelo IME do lado do usuário. Os outros campos formam o "lado da rede", normalmente únicos em uma UNI, e são atriibuídos pelo lado da rede. O endereço de um terminal ATM de uma UNI Privada seria obtido pela junção dos valores ESI e SEL com um prefixo da rede para aquela UNI.

Os procedimentos estabelecidos para o registro de endereços consistem em:

- Troca de informações de endereçamento em momento de inicialização
- Restrições das combinações de prefixo da rede e parte do usuário
- Aceitação de prefixos de redes livres
- Rejeição de valores inaceitáveis, tanto rejeição de um prefixo de rede por parte do usuário, como a rejeição da parte do usuário por parte da rede.
- Adição/Remoção dinâmica de prefixos da rede ou parte do usuário
- Remoção de um endereço quando a conectividade ILMI é perdida

- Indicação de suporte ou falta de suporte para o registro de endereços em uma interface.

Duas tabelas na MIB foram definidas para tratar do registro de endereços: uma contendo prefixos de rede (atmfNetPrefixTable) e outra contendo os endereços ATM registrados (atmfAddressTable).

A tabela de prefixos da rede fica no IME do lado do usuário, mas é mantida pelo IME do lado da rede, que envia PDUs SNMP de SetRequest para registrar ou remover prefixos. Analogamente, a tabela de endereços ATM fica no IME do lado da rede, mas é mantida pelo IME do lado do usuário, que envia PDUs SNMP de SetRequest para registrar ou remover endereços ATM.

Na inicialização, o registro de prefixos de rede ocorre primeiro. A seguir, o IME do lado do usuário combina a partes do usuário com o prefixo da rede registrado para formar um endereço ATM completo. Então, o IME do usuário registra esse endereço na atmfAdressTable. A figura 5.5 mostra o processo de registro de endereços, quando da inicialização do sistema.

O grupo atmfAddressRegistrationAdminGroup, implementado em todos IMEs, contém um indicador que informa se os mecanismos de registro de endereço, descritos acima, são suportados ou não naquela interface.

Service Registry MIB

A Service Registry MIB provê um registro geral de serviços para a localização de servidores ATM, como o LECS (*LAN Emulation Configuration Server*) ou o ANS (*ATM Name Server*). Essa MIB é implementada em IMEs que representam o "lado da rede". Para localizar um serviço, um IME de usuário consulta o IME da rede por informações contidas na tabela atmfSrvcRegTable. Cada entrada dessa tabela contém o endereço do servidor para que o IME do usuário possa estabelecer conexão com o servidor.

5.2.2 Integração ILMI e AToM MIB

A figura 5.6 mostra a proposta de integração da ILMI com a AToM MIB, feita pelo ATM Forum. Um agente proxy recebe todas as solicitações do gerente. A partir dos dados na *community* da PDU SNMP, o agente decide para onde enviar a solicitação, se para a IME (local ou remota) ou para a AToM MIB. Caso receba pedidos SNMPv2, o proxy traduz o pedido de SNMPv2 para SNMP, como especificado na RFC 1908.

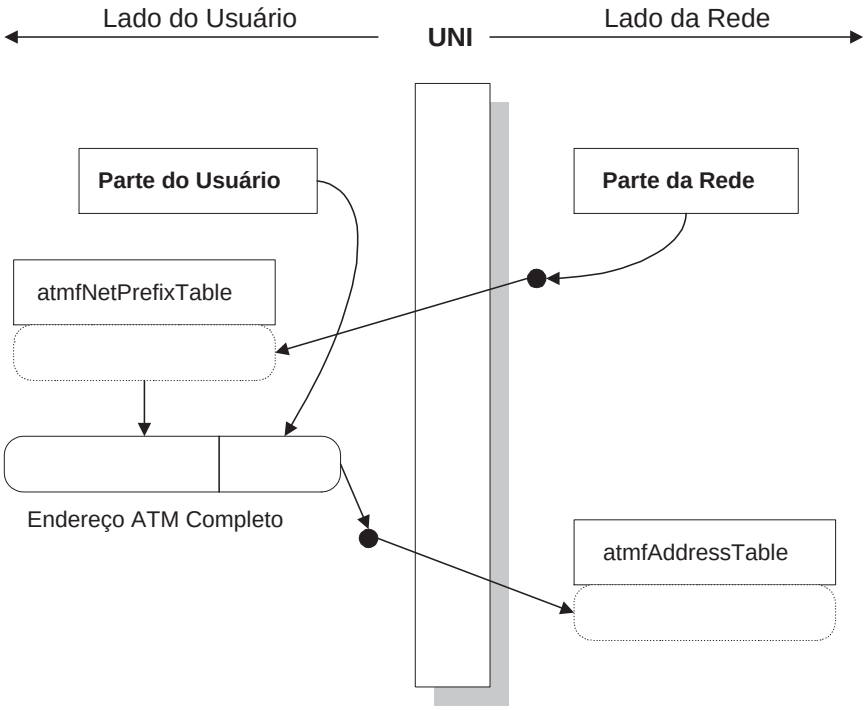

Figura 5.5: Mecanismo de Registros de Endereços na ILMI

5.3 Funções OAM

O OAM é um conjunto de funções de Operação, Administração e Manutenção, recomendadas pela ITU-T no contexto das RDSIs. Esta infra-estrutura foi, posteriormente, ratificada pelo ATM Forum, em sua recomendação da interface UNI 3.1. Atualmente, cinco funções de gerenciamento OAM estão definidas. São elas:

- Monitoramento de Desempenho
- Detecção de Falhas e Defeitos
- Proteção de Sistemas
- Informações de Falha e Desempenho
- Localização de Falhas

Para realizar as operações acima citadas, as funções OAM são aplicadas em 5 níveis hierárquicos de fluxos: F1, F2, F3, F4 e F5. Os fluxos OAM que dizem respeito à Camada Física são F1, F2 e F3. A implementação destes fluxos varia de acordo com a estrutura

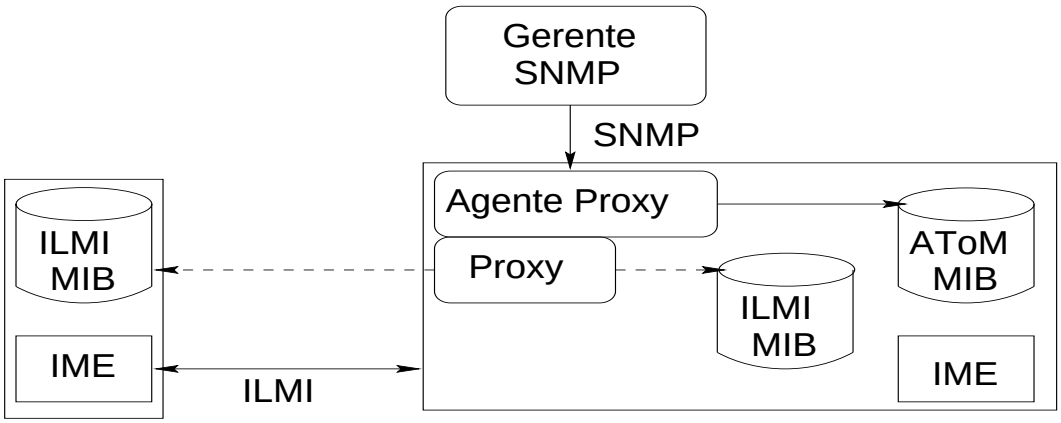

Figura 5.6: Integração ILMI - AToM MIB

de transmissão utilizada. Por exemplo, os fluxos F1, F2 e F3 utilizam os bits de *overhead* dos quadros em sistemas baseados em SDH.

Já os fluxos F4 e F5 têm como finalidade o monitoramento de funções na camada ATM, sendo implementadas como células especiais denominadas células OAM. Tais células permitem monitoramento de desempenho, detecção e localização de falhas e proteção do sistema ao nível de caminhos virtuais (F4) e canais virtuais (F5). Assim, os dispositivos ATM terão a capacidade de conseguir informações sobre conexões fim-a-fim, reduzindo a necessidade de se distribuir MIBs por toda a rede, diminuindo o tráfego de gerência [4].

5.3.1 Células OAM

Como dito anteriormente, OAM é implementado em nível de camada ATM como células especiais, denominadas células OAM. Os fluxos F4 e F5 se utilizam de vários tipos de células OAM para realizar determinadas funções. A definição dessas funções é feita através dos campos da células OAM, apresentada na figura 5.7.

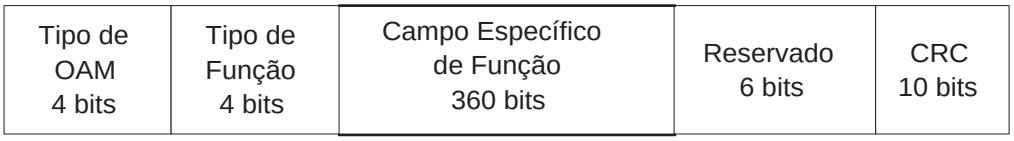

Figura 5.7: Payload da Célula OAM

Os valores possíveis para os campos Tipo de OAM e Tipo de Função do *payload* da célula OAM são mostrados na tabela 5.2. O campo específico, mostrado na figura, transporta informações relativas a cada função OAM. O campo CRC (*Cyclic Redundancy Check*) é utilizado para a detecção de erros no *payload* desta célula.

Tipo de Célula OAM	Valor	Tipo de Função OAM	Valor
Gerenciamento de Falhas	0001	*Alarm Information Signal*	0000
		Far-End Receive Failure	0001
		OAM cell loopback	0010
		Continuity Check	0100
Gerenciamento de Desempenho	0010	*Forward Monitoring*	0000
		Backward Monitoring	0001
		Monitoring/Reporting	0010
Ativação/Desativação	1000	*Performance Monitoring*	0000
		Continuity Check	0001

Tabela 5.2: Campos da Célula OAM

Os fluxos F4 e F5 podem ser aplicados tanto a nível de conexão fim-a-fim quanto a nível de segmento de conexão. No caso do fluxo F4, sua identificação é feita através da aplicação de valores de VCI 3 e 4 para conexões fim-a-fim e segmentos de conexões, respectivamente. O fluxo F5 possui os mesmos valores de VPI/VCI das células de usuários, sendo identificado pelos valores do campo PTI (*Payload Type Identifier*) 101 e 100 para conexões fim-a-fim e segmentos de conexões, respectivamente.

O funcionamento dos fluxos F4 e F5 segue algumas regras, que dependem dos componentes de uma conexão, os quais podem ser: ponto de conexão, nó terminal ou nó intermediário. Um ponto de conexão é representado pelo equipamento onde as conexões são solicitadas (por exemplo, uma estação). Chamamos de nós terminais os comutadores aos quais os dois pontos de conexão estão diretamente ligados. Um nó intermediário é um comutador localizado entre nós terminais ao longo da conexão.

Assim, a aplicação dos fluxos F4 e F5 em uma rede ATM é controlada pelas seguintes regras:

- Os fluxos OAM são finalizados nos nós terminais ou nos nós intermediários das bordas de segmentos de uma conexão (VC ou VP). Cada fluxo é bidirecional.
- As células OAM devem seguir o mesmo caminho físico de nós intermediários nas duas direções.
- Os nós intermediários podem inspecionar as células OAM que chegam, ou inserir novas células OAM, mas não podem terminar um fluxo OAM.
- O controle das células OAM deve se certificar de que um fluxo foi propriamente terminado.

Entre as funções da células OAM, destacamos a indicação de falhas e os mecanimos de loopback.

Indicação de Falhas

Os Mecanismos de Indicação de Falhas têm como função propagar aos elementos de rede ou a toda a rede a existência de problemas, sejam estes da camada Física ou ATM.

Existem mecanismos específicos de indicação de alarmes para VC e VP em redes ATM. São eles:

- **Sinais de Indicação de Alarmes (AIS - Alarm Indication Signals)**

  Os VP-AIS e VC-AIS são gerados por um nó que detectou algum problema em nós anteriores, com a função de sinalizar o problema para os nós seguintes. Estes sinais são propagados através de células OAM (Operation and Mantainance) do tipo de Gerenciamento Falhas, que contêm a semântica do sinal AIS em um de seus campos.

- **Indicações de Defeito Remoto (RDI - Remote Defect Indication)**

  Os alarmes do tipo VP-RDI e VC-RDI são gerados por um dos nós terminais à conexão virtual defeituosa, com a função de notificar o problema aos nós seguintes. Estes sinais são propagados através de células OAM do tipo de Gerenciamento Falhas, que contêm a semântica do sinal AIS.

Uma falha de VPC na camada ATM é verificada quando da recepção de uma notificação a partir da camada Física ou a partir de algum elemento de rede. O procedimento de geração de notificações varia, dependendo de onde o sinal é percebido. Uma notificação pode ser detectada nos nós intermediários, de origem ou de destino da conexão.

Mecanismo de Loopback

O mecanismo de loopback OAM pode ser utilizado para verificar conectividade, isolar falhas e realizar testes de conformidade de serviço. Estes testes são realizados pela inserção de células OAM especiais para lookback de Gerenciamento de Falhas. Células deste tipo são lançadas na rede a partir de um ponto arbitrário, contendo instruções para cada nó intermediário enviar mensagem de recebimento da célula. Desta forma, é possível não só testar conectividade, mas também verificar o tempo de resposta do teste, através de *timestamps*.

Na figura 5.8, o nó terminal T1 gera célula loopback com destino ao nó terminal T2. Ao longo do caminho, cada nó intermediário emite uma célula de reconhecimento (ACK) do recebimento da célula loopback. Como o defeito se encontra entre os nós intermediários

N2 e N3, a célula não pode ser propagada para os nós N3 e T2, que por sua vez não emitem o ACK do recebimento de loopback OAM. Desta forma o defeito é localizado.

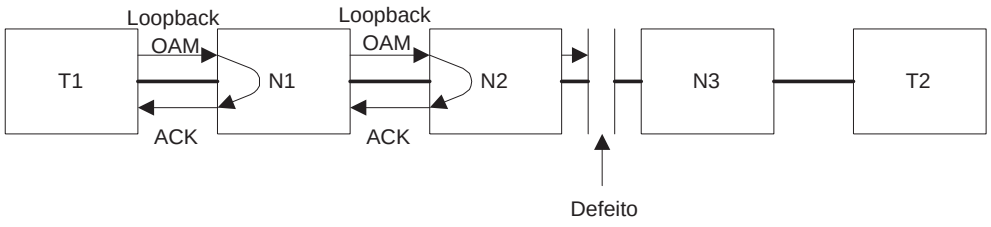

Figura 5.8: Mecanismo de Loopback das Células OAM

Várias aplicações para loopback são especificadas pela ITU-T, na Recomendação I.610 [51]. Entre elas estão:

- **Loopback Fim-a-Fim**

  Células OAM loopback são geradas em um dos nós terminais com destino à outra extremidade da conexão.

- **Loopback de Linha de Acesso**

  Células OAM loopback para segmentos de conexão são produzidos por um cliente de uma concessionária. O primeiro equipamento da rede pública deve responder à célula. Tem como função principal testar o acesso de um cliente à rede pública, através de uma concessionária.

- **Interdomain Loopback**

  Células OAM loopback são lançadas à rede a partir de um domínio e são retornadas pelo primeiro equipamento de comutação do domínio destino da conexão.

- **Network-to-End Loopback**

  Células OAM loopback são geradas em um domínio e respondidas pelo nó terminal da conexão em outro domínio.

- **Intradomain Loopback**

  Células OAM loopback para segmentos de conexão são lançadas por um nó terminal e retornado por um nó intermediário da conexão.

Parte II

RENATA

Capítulo 6

Arquitetura da RENATA

6.1 Redes Neurais na Gerência ATM

A gerência Internet tornou-se um padrão *de facto* nas redes de computadores. Portanto, é natural que a adoção de seu modelo e protocolos tenha sido a primeira tentativa no gerenciamento de ambientes baseados na tecnologia ATM. Contudo, o SNMP não tem suportado suficientemente as especificidades do ATM. A diversidade de serviços oferecidos, o ambiente de alta velocidade e a necessidade de uma solução integrada exigem novos requisitos tanto nos protocolos quanto no modelo de gerência a ser adotado.

Dados estão distribuídos em agentes SNMP com capacidades computacionais limitadas e o processamento destes dados para extração de informação de gerência é centralizado nos gerentes, que assim têm grande carga operacional. Uma alternativa é a aplicação de Agentes Inteligentes, que possuem maior autonomia e capacidade computacional. Este agente poderia, assim, atuar sem necessariamente consultar o gerente. Essa maior autonomia diminui o tráfego de gerência, agilizando o gerenciamento da rede. Se o agente apresentar também comportamento pró-ativo, uma maior otimização é possível.

Entre os mecanismos inteligentes que permitem estas características, as redes neurais têm motivado muitas pesquisas, devido suas características de aprendizado, generalização, adaptabilidade, robustez e tolerância a falhas. Vários trabalhos têm aplicado redes neurais no contexto das redes ATM.

Deste modo, surge o interesse pela pesquisa da aplicação das redes neurais na gerência pró-ativa de redes ATM. Para isso, devem ser criadas facilidades para o desenvolvimento de Agentes Inteligentes baseados em redes neurais. Assim, nasce RENATA (**RE**des **N**eurais **A**plicadas ao **T**ráfego **A**TM), um ambiente que visa a permitir gerência pró-ativa de redes ATM, através do desenvolvimento de agentes baseados em redes neurais.

Neste capítulo, a arquitetura da RENATA é apresentada. A seção 6.2 descreve sua arquitetura funcional. Já a seção 6.3 especifica sua arquitetura física, que identifica os

produtos utilizados e as ferramentas implementadas.

6.2 Arquitetura Funcional

Devido às suas características, já citadas, de aprendizado, generalização e adaptabilidade, as redes neurais dotam os agentes gerados pela RENATA do conhecimento para detectar antecipadamente situações problemáticas em uma rede ATM através da análise de seus estados. Os dados para treinamento da rede neural são obtidos na RENATA através de simulações da rede ATM a ser gerenciada.

A Figura 6.1 mostra a Arquitetura Funcional da RENATA, que possui, basicamente, três módulos: *Módulo de Treinamento*, *Módulo Neural* e *Módulo de Gerência*. A arquitetura define a funcionalidade de cada módulo e como estes interagem para o desenvolvimento de Agentes Inteligentes.

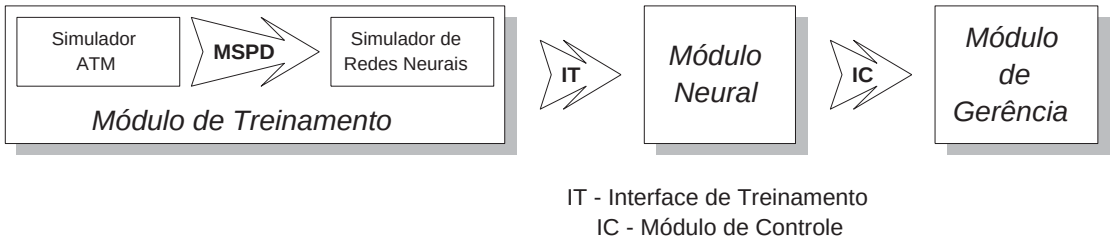

Figura 6.1: Arquitetura Funcional da RENATA

No Módulo de Treinamento, a rede neural é projetada, treinada e validada. Para isto, este módulo é dividido em três componentes: o Simulador ATM, o MSPD (Módulo de Seleção e Preparação de Dados) e o Simulador de Redes Neurais. O Módulo Neural consiste na rede neural resultante da saída do Módulo de Treinamento e de informações sobre sua arquitetura e objetivo. O Modulo de Gerência é responsável pela integração e ativação do Módulo Neural, através do desenvolvimento de um agente que fornece dados para a predição da rede neural e, de acordo com sua saída, toma determinadas ações.

O treinamento da rede neural é realizado *off-line* para que esta tenha melhor desempenho em um ambiente de alta velocidade durante a fase de operação. Por isso, o Módulo de Treinamento é separado do Módulo de Gerência. Considerando o tipo de aprendizado, RENATA pode aceitar vários modelos de rede neural.

6.2.1 Módulo de Treinamento - Simulador ATM

Normalmente, as redes neurais precisam de uma grande quantidade de dados de treinamento para aprenderem. Em situações reais de telecomunicações, a aplicabilidade de redes neurais depende da disponibilidade de padrões de teste, tanto de funcionamento normal quanto anormal da rede. Se não houver registros históricos, torna-se difícil a obtenção de dados, pois não se pode parar uma rede em produção apenas para testá-la com problemas. Assim, o uso de um simulador ATM na RENATA se justifica por essa dificuldade na obtenção de dados e pela flexibilidade de se simular uma rede nas mais diversas configurações.

Definido o problema a ser diagnosticado pela rede neural, são feitas simulações visando à obtenção dos dados necessários para o aprendizado da rede neural. Inicialmente, é descrita uma topologia de rede ATM no Simulador: comutadores, enlaces e terminais, com suas características. A partir da topologia, simulam-se aplicações executando pela rede, descrevendo carga e tipo de tráfego. Pode-se também simular falha em algum segmento ou grande carga sobre um nó para observação do comportamento da rede e, assim, obter parâmetros que caracterizem estas situações.

De acordo com o problema, determinadas opções de log no Simulador ATM são habilitadas de modo que este gere os dados que servirão como base para o treinamento da rede neural. Estes dados devem corresponder aos parâmetros que serão posteriormente monitorados por um agente numa rede ATM real.

6.2.2 Módulo de Treinamento - MSPD

Antes de serem submetidos à rede neural, os dados precisam ser selecionados, divididos e tratados. Essas funções são realizadas pelo MSPD (Módulo de Seleção e Preparação de Dados).

Segundo [35], aproximadamente 98% destes dados devem ser de funcionamento normal e 2% devem caracterizar as situações que a rede neural deve detectar. A seguir, os dados são separados entre treinamento e validação da rede neural, na proporção de 80% para treinar, 10% para testar e o restante para validar a rede neural.

Depois de selecionados, os dados devem ser tratados para que possam ser submetidos à rede neural. Os dados devem ser traduzidos do formato de log do Simulador ATM para o formato do Simulador de Redes Neurais. Além disso, alguns modelos de redes neurais só aceitam entradas binárias, enquanto outros aceitam reais na escala de 0 a 1 ou -1 a +1. Nestes casos, técnicas como normalização, escalonamento e codificação binária 1-N são utilizadas. Por exemplo, o número de conexões ativas passando por um comutador é um número inteiro que deve ser escalonado para real no intervalo de 0 a 1 para ser aceito como uma entrada válida de uma determinada rede neural.

Aspectos como o tipo de treinamento devem ser também considerados na preparação dos dados. Por exemplo, se o treinamento for supervisionado, os dados devem conter também a saída desejada da rede neural para cada entrada.

6.2.3 Módulo de Treinamento - Simulador de Redes Neurais

Com o tipo de problema definido e os dados selecionados e preparados, o modelo da rede neural deve ser escolhido. A partir dessa escolha, o Simulador de Redes Neurais é utilizado para modelar a rede neural e controlar seu treinamento. Parâmetros gerais como a taxa de aprendizado e função de ativação, além de alguns específicos a cada modelo devem ser configurados. Tais parâmetros determinam a velocidade do treinamento e o grau de generalização da rede neural, entre outros fatores. O erro da rede neural deve ser monitorado durante o treinamento para determinar se a rede neural está convergindo ou se algum parâmetro foi mal escolhido. Em [35] e [57], encontram-se discussões sobre o treinamento de vários modelos de redes neurais. A meta é otimizar o desempenho da rede neural sobre os padrões de teste e validação. Para isso, recomenda-se testar seu desempenho sobre os dados de teste, enquanto o treinamento é feito. O critério de aprovação, parâmetro que determina quando a rede neural está treinada, depende do seu modelo, podendo ser a simples proporção de acertos sobre tentativas.

Por fim, a saída do Módulo de Treinamento é a rede neural treinada e testada. A Interface de Treinamento comunica o Módulo de Treinamento ao Módulo Neural, transmitindo os dados resultantes do simulador de Redes Neurais (a matriz de pesos da rede neural). Estes pesos representam o conhecimento adquirido durante o treinamento.

6.2.4 Módulo Neural

O Módulo Neural consiste na rede neural resultante da saída do Módulo de Treinamento e de informações sobre sua arquitetura e objetivo. A rede neural treinada e devidamente testada é um módulo de aplicação, que deve receber as entradas (normalizadas e codificadas) para realizar a predição sobre o estado da rede. Algum processamento sobre a saída pode ser também necessário.

O desempenho da rede neural deve ser monitorado. Se os resultados estiverem abaixo do critério de aprovação, é possível que a dinâmica da rede ATM tenha mudado: talvez por alterações na topologia ou pela introdução de novos serviços. Então, a rede neural deve ser novamente treinada. Contudo, pequenas mudanças na rede ATM devem ser absorvidas pela capacidade de generalização e adaptabilidade da rede neural.

A Interface de Controle comunica o Módulo Neural ao Módulo de Gerência. Esta interface repassa ao Módulo de Gerência as informações relativas à arquitetura da rede neural e ao seu propósito. Estas informações serão usadas no desenvolvimento do agente.

6.2.5 Módulo de Gerência

O Módulo de Gerência é responsável pela integração e ativação do Módulo Neural, através do desenvolvimento de um Agente Inteligente que acionará a rede neural e tomará as devidas ações de acordo com suas predições, visando a ter uma atitude pró-ativa.

Antes do desenvolvimento do agente, é necessário que seja configurado no Módulo de Gerência o ambiente de gerência ATM da rede em questão: os dispositivos que serão monitorados e seus respectivos mecanismos (MIBs SNMP, ILMI e células OAM).

Quando do desenvolvimento de um agente, as informações da rede neural providas pela Interface de Controle são processadas. Para cada entrada, é definido o dado que o agente consultará para alimentá-la. Assim como, para cada saída, é determinada a ação que deve ser tomada, utilizando os mecanismos de gerência ATM previamente configurados. A seguir, o agente é gerado, devendo ser depois instalado e ativado.

O Módulo de Gerência também é responsável pelo monitoramento dos agentes, além de possibilitar o gerenciamento direto da rede. As ações tomadas pelo agente devem ser registradas para posterior análise. Além disso, o agente deve alertar o Módulo de Gerência quando a rede neural estiver abaixo de seu critério de aprovação.

A tabela 6.1 resume os passos do desenvolvimento de um Agente Inteligente na RENATA.

Passo 1:	Definição do Problema
Passo 2:	Simulação da Rede ATM
Passo 3:	Preparação e Seleção dos Dados Gerados pelo Simulador ATM
Passo 4:	Projeto, Treinamento e Validação da Rede Neural
Passo 5:	Documentação da Rede Neural
Passo 6:	Configuração dos Mecanismos de Gerência da Rede
Passo 7:	Prototipação do Agente
Passo 8:	Instalação e Ativação do Agente
Passo 9:	Monitoramento do Agente

Tabela 6.1: Passos do Desenvolvimento de um Agente na RENATA

6.3 Arquitetura Física

A figura 6.2 apresenta a Arquitetura Física da RENATA, os seus módulos e como são realizadas as trocas de informações. Ela introduz um novo elemento no Módulo Treinamento: o Documentador de Redes Neurais (DocRN). Esta ferramenta é responsável pela criação do arquivo com as informações sobre a rede neural presente no Módulo Neural. Este módulo contém também a rede neural produzida pelo Módulo de Treinamento.

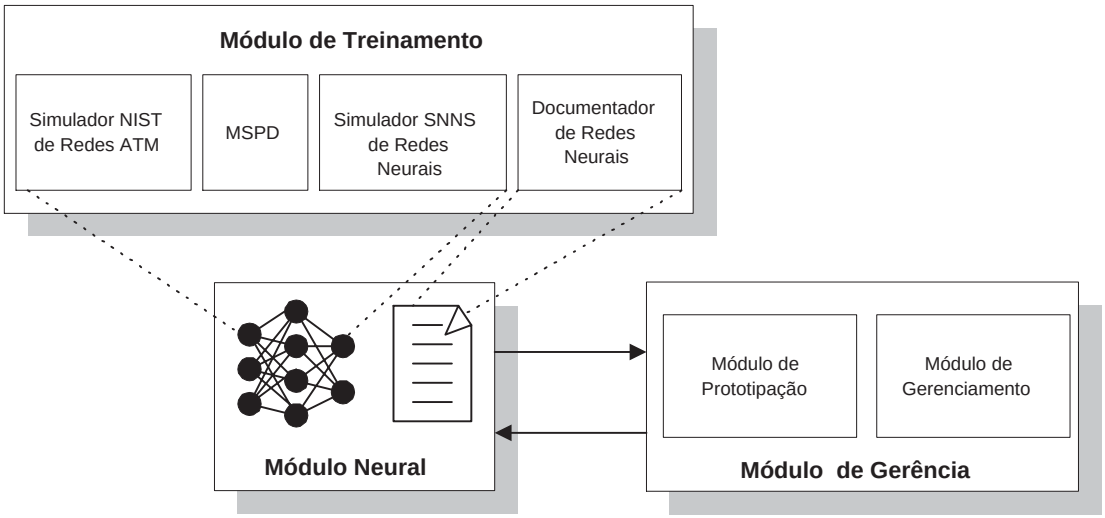

Figura 6.2: Arquitetura Física da RENATA

Fisicamente, o Módulo de Gerência é dividido em dois: o Módulo de Prototipação e o Módulo de Gerenciamento. O primeiro é responsável pelo criação do agente a partir das informações da rede neural e da configuração dos mecanismos de gerência da rede. O outro módulo é responsável pelo monitoramento dos agentes e ainda permite acesso direto à gerência da rede.

A parte mais complexa do desenvolvimento de uma aplicação na RENATA se dá realmente no Módulo de Treinamento: toda modelagem e treinamento da rede neural para diagnosticar determinado problema. A partir do Módulo Neural, o Módulo de Gerência permite o desenvolvimento dos agentes.

O agente resultante deste processo consistirá basicamente dos mecanismos de gerência para monitorar e controlar a rede ATM (determinados no Módulo de Gerência) e do conhecimento para predizer determinada situação (Módulo Neural).

A seguir, as opções para implementação de cada módulo da RENATA serão apresentadas. No caso do simulador ATM e do simulador de redes neurais, dada a complexidade de suas funções, produtos prontos são utilizados. Estes produtos são integrados visando à criação da rede neural. Para o restante do processo, foram implementadas as seguintes ferramentas: o MSPD, o Documentador de Redes Neurais e o Módulo de Gerência.

6.3.1 Produtos Utilizados

Simulador ATM

O Simulador NIST de redes ATM foi desenvolvido no *National Institute of Standards and Technology* (NIST) para fornecer um ambiente de estudo e determinação de desempenho de redes ATM. Dentro de uma ambiente gráfico interativo, o Simulador ATM permite que o usuário crie diferentes topologias e configure parâmetros da operação de cada componente. Enquanto a simulação é executada, várias medidas de desempenho podem ser mostradas na tela ou salvas em arquivo para análise posterior [58].

Simulador de Redes Neurais

O SNNS (*Stuttgart Neural Network Simulator*) é um simulador de redes neurais desenvolvido na Universidade de Stuttgart desde 1989. A meta do projeto é a criação de um ambiente flexível e eficiente de simulação para a pesquisa e aplicação de redes neurais [57]. O simulador pode ser usado para criar, modelar, treinar, testar, realizar podar, analisar e visualizar redes neurais de diversos modelos.

Módulo Neural

O Módulo Neural consiste da rede neural produzida pelo Módulo de Treinamento e pela sua documentação gerada pelo DocRN. A partir da rede neural treinada e validada no SNNS, é utilizada uma ferramenta do pacote do SNNS, o compilador *snns2c*, que gera código C com a representação da rede neural.

6.3.2 Ferramentas Implementadas

MSPD

O MSPD é o módulo responsável pela seleção e preparação dos dados que serão submetidos à rede neural. O MSPD realiza a tradução do arquivo do formato do simulador NIST para o formato do SNNS, selecionando os dados que são relevantes para o aprendizado da rede neural. Além disso, o tratamento necessário do dado bruto para que este possa ser uma entrada da rede neural também é realizado no MSPD.

Documentador de Redes Neurais

Para que o agente possa alimentar a rede neural e interpretar suas saídas é necessário que o Módulo de Gerência obtenha estas informações quando da criação do agente. Estes dados são gerados a partir do Documentador de Redes Neurais (DocRN).

No DocRN, são documentadas informações sobre o objetivo da rede neural, sua arquitetura (entradas e saídas), como esta deve ser ativada e avaliada, e alguns parâmetros do seu treinamento, entre outros dados. As informações sobre as entradas definem que dados a rede neural deve receber para fazer a predição e como esses dados brutos devem ser tratados para poderem ser submetidos à rede neural. Para cada saída, devem ser fornecidos a interpretação de tal predição e uma sugestão de ação a ser tomada. O DocRN foi implementado na linguagem Java, principalmente por suas características de portabilidade.

Módulo de Gerência

As funcionalidades do Módulo de Gerência estão divididas entre seus dois sub-módulos. O Módulo de Prototipação é responsável pela geração do código do agente a partir das informações geradas pelo DocRN e da configuração dos mecanismos de gerência da rede ATM em questão; isto é, a configuração das fontes para as entradas da rede neural e do comportamento do agente mediante as predições da rede neural. O Módulo de Gerenciamento oferece acesso à gerência da rede ATM, além de ser responsável pelo monitoramento dos agentes gerados pelo Módulo de Prototipação.

Ambos foram implementados em Java, com auxílio da API Java SNMP da *Advent Network Management, Inc.* Esta biblioteca foi criada para permitir a implementação de *applets* e aplicações Java que utilizem o protocolo SNMP para se comunicar com nós gerenciados. Foi utilizada a versão 1.3.1 da API para SNMPv2C, dado que o agente desenvolvido na RENATA pode consultar MIBs SNMP para obter as informações necessárias à rede neural, além de também permitir seu monitoramento.

A tabela 6.2 descreve as principais classes da API Java SNMP utilizadas na implementação do protótipo.

Outra opção considerada para implementação da parte de gerência SNMP foi o JMAPI (*Java Management Application Programming Interface*), desenvolvido pela Sun Mycrosystems. O JMAPI é "uma coleção de classes Java que permite que programadores construam mais facilmente ferramentas para prover soluções integradas de gerenciamento, usando as vantagens do ambiente computacional proporcionado pelo Java" [59]. Uma das vantagens do JMAPI é a integração nativa dos conceitos de administração e gerência de redes, que poderia ser útil na RENATA. Porém, as questões da distribuição (via RMI - *Remote Method Invocation*) e do modo próprio de tratamento eventos (o JMAPI EMS - *Event Management Service*) do JMAPI acarretariam em mais complexidade para os agentes gerados pela RENATA.

O Módulo de Prototipação gera código Java para o agente. A rede neural é acessada através de chamadas a métodos nativos. O código do agente deve ser compilado e depois instalado no dispositivo preestabelecido, que deve possuir a máquina virtual Java.

Classes	Descrição
SnmpApi	Gerencia as sessões criadas pela aplicação, gerencia os módulos de MIBs carregados e armazena parâmetros importantes da comunicação SNMP, como as portas utilizadas. Para usar as outras classes, é necessário primeiro instanciar esta.
SnmpSession	É utilizada para gerenciar uma sessão de comunicação entre duas entidades SNMP. Provê métodos para abrir sessões (em portas determinadas, se desejado), envio síncrono e assíncrono de pedidos SNMP e teste de respostas e *timeouts*.
SnmpPDU	Provê métodos para criação e utilização de PDUs SNMP. Os métodos incluem a adição de *variable bindings* nulos (para o envio) e impressão do conteúdo dos *variable bindings* (para a recepção).
SnmpVar	Desta classe herdam sub-classes que modelam as variáveis SNMP, como a *SnmpInt*, *SnmpString*, *SnmpCounter*, etc.
MibModule	Possibilita a utilização dos dados contidos em um arquivo de um módulo de MIB, a partir do *parse* desta MIB.
MibNode	Representa um nó da arvore da MIB depois de seu *parse*. Possibilita acesso às informações específicas sobre a sintaxe, a descrição e as possíveis folhas do nó.
LeafSyntax	Usada para representar uma sintaxe. Por exemplo, o SnmpInt.

Tabela 6.2: Principais Classes da API Java SNMP

6.3.3 Utilizando RENATA

Acesso aos Mecanismos de Gerência ATM

Após uma fase inicial de configuração (utilizando a ferramenta RenataSetup), onde os mecanismos de gerência da rede em questão são incorporados à RENATA, os agentes e o usuário através do Módulo de Gerenciamento poderão iniciar um fluxo de células OAM e consultar as MIBs SNMP da rede através de um Navegador de MIBs. Desse modo, RENATA concilia os mecanismos de gerência ATM com a inteligência das redes neurais, visando a dotar os agentes desenvolvidos de uma atitude pró-ativa.

Além de oferecer acesso direto aos mecanismos de gerência ATM (MIBs SNMP Aplicadas ao ATM, ILMI e células OAM), RENATA permite a seleção através desses mecanismos das informações que serão monitoradas pelo agente. Este, por sua vez, repassará essas informações devidamente tratadas para a rede neural que realizará com esses dados a predição sobre o estado da rede.

Para o caso da detecção de uma situação anormal, deve-se também configurar as ações que o agente deve tomar entre aquelas permitidas pelos mecanismo de gerência ATM. Por

exemplo, se o agente poderá atuar no recurso (via SNMP, ILMI ou OAM) ou se deve apenas notificar o usuário.

Interface com Usuário

RENATA tem dois tipos de usuários: o *desenvolvedor*, que utiliza RENATA para criar aplicações de redes neurais para a gerência ATM; e o *administrador de redes*, que configura essas aplicações para executarem na sua rede, através da prototipação de agentes.

O desenvolvedor se preocupa com todo processo, principalmente com a modelagem e treinamento da rede neural para detectar situações anormais numa rede ATM ou para realizar tarefas como as descritas na seção 4.11. Enquanto que o administrador se preocupa com a configuração, instalação e monitoramento dos agentes. Cada usuário tem sua própria interface, diferenciadas por suas ferramentas e tarefas. A figura 6.3 mostra o protótipo com as suas interfaces.

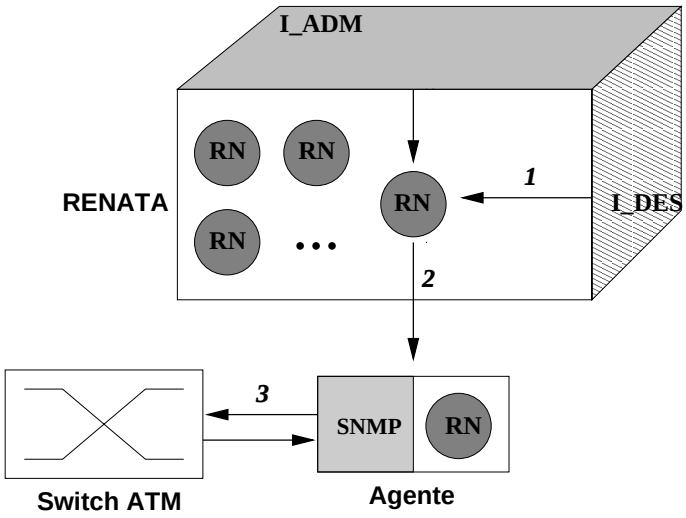

Figura 6.3: Protótipo da RENATA

A *I_DES* (Interface do Desenvolvedor) permite acesso às facilidades do Módulo de Treinamento: o Simulador ATM, o MSPD, o Simulador de Redes Neurais e o Documentador de Redes Neurais. A partir da I_DES, o desenvolvedor cria as redes neurais que, posteriormente, serão executadas a partir de agentes configurados pelo administrador. A *I_ADM* (Interface do Administrador) integra mecanismos de gerência ATM com as ferramentas para prototipação dos agentes, isto é, o Módulo de Gerenciamento e o Módulo de Prototipação. O elemento comum entre as interfaces é o Módulo Neural.

A figura 6.3 também mostra o funcionamento geral do protótipo dividido em três grandes etapas. Na etapa 1, o desenvolvedor cria um Módulo Neural, a partir das ferramentas do Módulo de Treinamento. Na etapa 2, o administrador configura o agente que conterá

a rede neural. Já a etapa 3 exemplifica o funcionamento do agente desenvolvido atuando no recurso ATM através do SNMP.

A tabela 6.3 detalha em passos as etapas do desenvolvimento de um agente na RENATA, relacionando as ferramentas e interfaces utilizadas.

I_DES	
Passo 1:	Definição do Problema
Passo 2:	Simulação da Rede ATM
	Ferramenta: *Simulador NIST*
Passo 3:	Preparação e Seleção dos Dados Gerados pelo Simulador ATM
	Ferramenta: *MSPD*
Passo 4:	Projeto, Treinamento e Validação da Rede Neural
	Ferramenta: *Simulador SNNS*
Passo 5:	Documentação da Rede Neural
	Ferramenta: *Documentador de Redes Neurais*
I_ADM	
Passo 6:	Configuração dos Mecanismos de Gerência da Rede
	Ferramenta: *RENATA_SETUP*
Passo 7:	Prototipação do Agente
	Ferramenta: *Módulo de Prototipação*
Passo 8:	Instalação e Ativação do Agente
Passo 9:	Monitoramento do Agente
	Ferramenta: *Módulo de Gerenciamento*

Tabela 6.3: Passos e Ferramentas do Desenvolvimento de um Agente na RENATA

Capítulo 7

Descrição do Protótipo

O objetivo deste trabalho é a proposição e a implementação de um ambiente genérico de desenvolvimento de agentes inteligentes baseados em redes neurais. A figura 7.1 mostra a arquitetura física da RENATA, destacando as ferramentas implementadas. Neste capítulo, as seguintes ferramentas são descritas com mais detalhes:

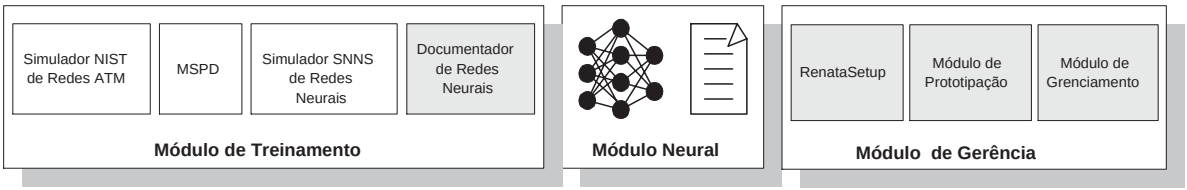

Figura 7.1: Ferramentas Implementadas da RENATA

- **DocRN**: usada pelo desenvolvedor para descrever as características e funcionalidades da rede neural criada no Módulo de Treinamento.
- **RenataSetup**: ferramenta onde o administrador informa quais são as máquinas que serão gerenciadas pela RENATA e quais são os mecanismos de gerência nelas presentes.
- **Módulo de Gerenciamento**: possibilita acesso direto aos mecanismos e informações de gerência ATM, além de também permitir o monitoramento dos agentes gerados pela RENATA.
- **Módulo de Prototipação**: gera os agentes, a partir das informações de gerência da rede geradas pelo RenataSetup e das informações da rede neural geradas pelo DocRN.

O MSPD (Módulo de Seleção e Preparação de Dados) não foi implementado por ser intrinsicamente específico a cada aplicação. Por isso, cabe ao desenvolvedor implementar o MSPD adequado a sua rede neural.

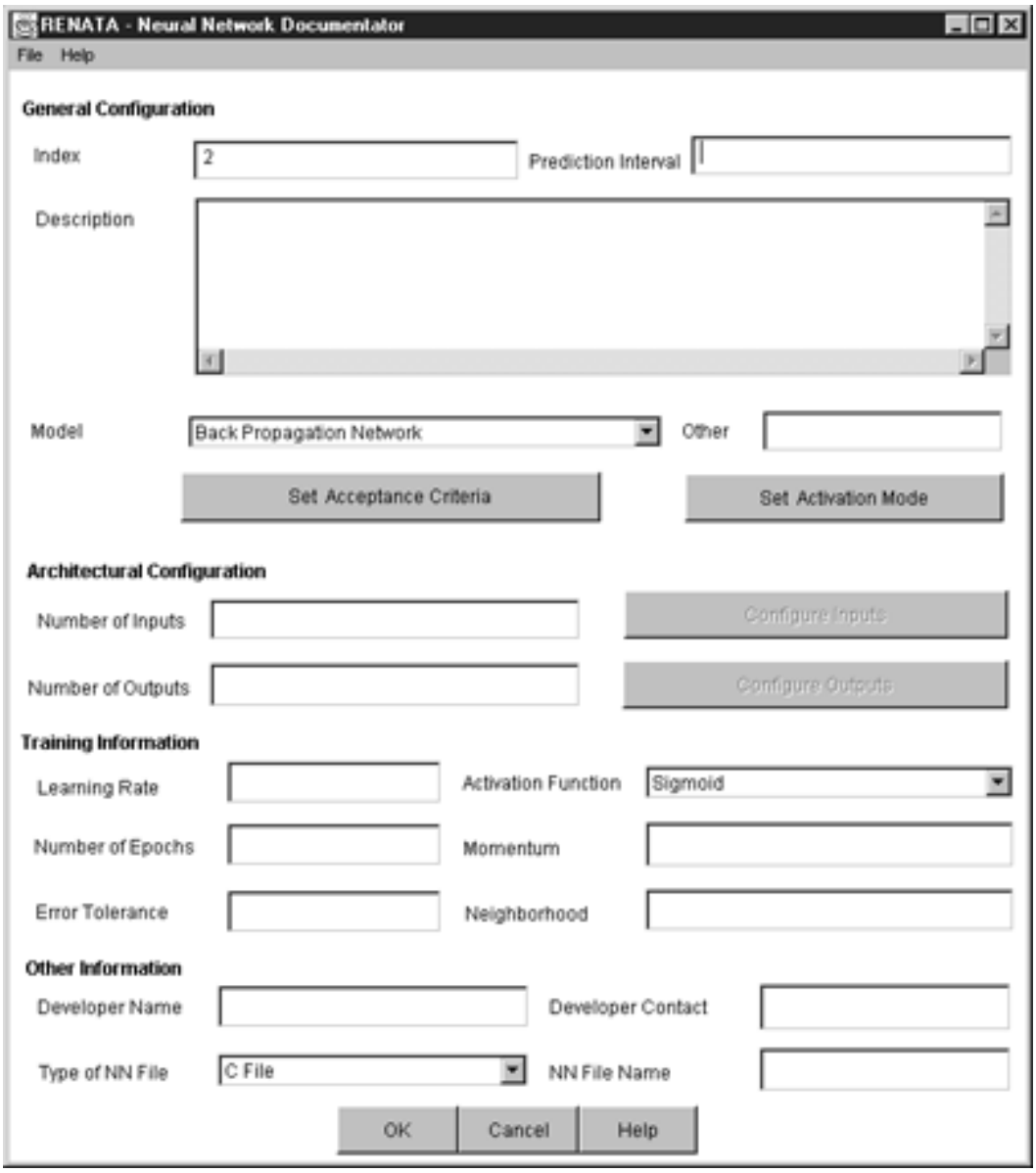

Figura 7.2: DocRN

Estas ferramentas foram implementadas em Java, devido aos motivos já citados no capítulo 6. Foi utilizada a versão 1.1.7 do JDK (*Java Development Kit*), fornecido pela SUN. No entanto, como a API SNMP da AdventNet está disponível apenas para Solaris, Linux e Microsoft Windows, RENATA executa sobre estas plataformas. Para o projeto

da interface visual das ferramentas, foi utilizada a versão 1.2 do *Borland JBuilder*. O código gerado com os componentes é modificado para não ficar dependente de nenhuma classe da Borland, utilizando apenas as classes da java.awt.

7.1 DocRN

Na janela principal do DocRN (figura 7.2), o desenvolvedor fornece informações gerais e arquiteturais sobre a rede neural desenvolvida no Módulo de Treinamento, além de alguns parâmetros do seu treinamento. No DocRN, o desenvolvedor descreve que predições podem ser realizadas pela rede neural e que informações são necessárias para essas predições.

Na *Configuração Geral*, o desenvolvedor informa o Intervalo de Predição (período de tempo que o agente deve esperar para ativar a rede neural, em milissegundos), a descrição do problema ou situação que será prevista pela rede neural e seu modelo. Desta janela, são chamadas outras que especificam o Critério de Aceitação e o Modo de Ativação da rede neural. Estas janelas serão descritas a seguir (seções 7.1.1 e 7.1.2, respectivamente).

Na parte de *Configuração Arquitetural*, o desenvolvedor inicialmente fornece o número de entradas e saídas. A partir dos botões, são abertas janelas para descrição e tratamento de cada entrada (seção 7.1.3) e saída (seção 7.1.4).