ÁTILA - Uma Arquitetura Inteligente e Distribuída para o Gerenciamento de Redes ATM

Marcelo Vidal Vasconcelos
mvidal@mcc.ufc.br
Departamento de Computação
Universidade Federal do Ceará – UFC
CEP: 60.455-760 Fortaleza-Ce

Mauro Oliveira
mauro@benfica.etfce.br
Curso de Informática Industrial
Centro Federal de Educação Tecnológica do Ceará
Av. 13 de Maio, 2081 - 60040-531 Fortaleza-Ce

Cidcley Teixeira de Souza
cidcley@mcc.ufc.br
Departamento de Computação
Universidade Federal do Ceará – UFC
CEP: 60.455-760 Fortaleza-Ce

Mardônio França
franca@mcc.ufc.br
Departamento de Estatística e Computação
Universidade Estadual do Ceará - UECe
Av. Paranjana, 1700 - 60740-000 Fortaleza-Ce

LAR- Laboratório Multiinstitucional de Redes e Sistemas Distribuídos
Processo CNPq 522198/96-1
UFC / CEFET-Ce / UECE

Resumo

Dada a complexidade e flexibilidade inerentes à redes ATM, um sistema de gerência de redes eficiente se torna cada vez mais indispensável. Entretanto, soluções de gerência que eram adequados para outras tecnologias são insuficientes para redes ATM.

Este trabalho mostra a complexidade do gerenciamento de redes ATM e apresenta o ÁTILA, uma arquitetura inteligente e distribuída para o gerenciamento pró-ativo neste ambiente. São especificadas as arquiteturas física e funcional do ÁTILA, e é descrito um protótipo que representa uma implementação da arquitetura proposta.

Abstract

Due to the complexity and flexibility inherent to ATM networks, an efficient network management system becomes really indispensable. However, management solutions ideal to other technologies become insufficient to ATM networks.

This work demonstrates the complexity of the ATM network management and presents the ATILA, an intelligent and distributed architecture for the pro-active management in this environment. The ATILA's physical and functional architectures are specified and a prototype, that represents an implementation of the proposed architecture here described.

1. Introdução

ATM é uma tecnologia apropriada para o tráfego multimídia (voz, vídeo e dados). Desenvolvida como suporte para RDSI (Redes Digitais de Serviços Integrados), ATM tem se revelado uma tecnologia também interessante para ambientes locais.

Novas tecnologias na comunicação de dados, como a ATM, ao acrescentarem novos parâmetros no atendimento à novas exigências de mercado (tráfego isócrono) exigem novos critérios de monitoração e controle, de modo a oferecerem a qualidade de serviço (QoS) esperada. Soluções que eram adequadas para outras tecnologias não são necessariamente eficazes na gerência de redes ATM, o que sugere, uma redefinição de paradigmas e o acréscimo de novas

funcionalidades a cada uma das áreas de gerência especificadas pela ISO (falha, desempenho, contabilização, segurança e configuração).

O ambiente de gerenciamento pró-ativo para redes ATM proposto em [Vasconcelos97], define uma estrutura onde várias questões complexas são tratadas através de um Banco de Dados Ativo. Este trabalho propõe o ÁTILA, uma arquitetura inteligente e distribuída que surge como uma evolução ao modelo citado anteriormente. Esta nova arquitetura se utiliza da distribuição de inteligência e de processamento das informações para otimizar a gerência pró-ativa e possibilitar a solução de novos problemas.

A seção 2 deste trabalho mostra as possíveis abordagens disponíveis para se gerenciar uma rede ATM e as dificuldades em se gerenciar tal ambiente. Na seção 3 é apresentado o ÁTILA, onde são descritas suas arquiteturas física e funcional e o fluxo dinâmico de informações. A implementação do ÁTILA é descrita na seção 4, onde é justificada a escolha de cada uma das tecnologias empregadas, juntamente com o detalhamento das ferramentas utilizadas na implementação das referidas tecnologias. Para finalizar, é descrito o protótipo, onde é apresentado a sua interface gráfica, e as peculiaridades da implementação de cada um dos elementos que compõe o ÁTILA.

2. Gerência de Redes ATM

2.1 Complexidade da Gerência ATM

O modelo de gerência para redes de pacotes é baseado no paradigma Gerente/Agente/MIB [Stallings93]. Este modelo, uma espécie de Cliente/Servidor/Base de Dados, possui restrições quando utilizado da mesma forma em ambientes ATM.

Estas restrições, que exigem tempos de resposta impossíveis de serem atendidos pelo modelo tradicional de gerência, existem devido às novas características de uma rede ATM, tais como :

- habilidade da rede de prover qualidade de serviço diferenciada à cada aplicação;
- proteger a rede e as estações de congestionamento, de maneira a garantir os níveis de desempenho desejado;
- promover a utilização eficiente dos recursos da rede;
- demandas de tráfego são estocásticas e não podem ser previstas;
- mecanismos de controle de tráfego para evitar certas situações, tais como: taxa total de entrada maior que a capacidade do enlace de saída (congestionamento), aumento da utilização de "*buffer*" rapidamente, gerando "*buffer overflow*" e perdas de células, etc.

2.2 Modelos de Gerência ATM

São apresentadas nesta seção quatro abordagens, que formam a base para um sistema de gerência de redes voltado ao ambiente ATM : MIBs SNMP desenvolvidas pelo IETF (Internet Engineering Task Force) e ATM Forum que tem como objetivo habilitar a visão do ambiente ATM para as aplicações baseadas em SNMP; modelo de referência para gerência de redes ATM e ILMI (Integrated Local Management Interface) especificados pelo ATM Forum e funções OAM desenvolvidas pelo ITU-T e ATM Forum.

- **SNMP aplicado à ATM** : O IETF e o ATM Forum desenvolvem MIBs específicas para a tecnologia ATM. Estas MIBs, que modelam a estrutura de uma rede ATM, dão ao Sistema de Gerência uma visão ampla de uma rede ATM [Minoli97].

- **Modelo de Referência de Gerenciamento de Redes ATM** : Criado pelo ATM Forum, descreve os vários tipos de gerência de redes necessários para suportar redes privadas, públicas e redes híbridas. Para isto foram criadas cinco interfaces, nomeadas M1, M2, M3, M4 e M5 [Alexander95].
- **ILMI 4.0** : Especificado pelo ATM Forum, é incluído como parte da UNI 4.0. O propósito deste padrão é auxiliar na gerência de configuração e "*status*" de uma interface específica. Ele também incorpora mecanismos para registro de endereços e serviços ATM através da UNI [ILMI4.0].
- **Funções OAM** : Com uma filosofia totalmente diferente das três abordagens anteriores, as funções OAM são voltadas para o gerenciamento a nível de camada e são utilizadas na distribuição automática de informações de gerência através de toda a rede, e da execução de testes (falha e desempenho) dentro delas [Minoli97].

2.3 Gerenciamento de Tráfego

As funções a seguir, apresentam uma abordagem para se gerenciar e controlar o tráfego e congestionamento numa rede ATM. Estas funções, específicas do ambiente ATM, podem ser combinadas dependendo da categoria de serviço a ser suportada [Traffic96].

- Controle de Admissão de Conexão (CAC);
- "*Usage Parameter Control*" (UPC);
- Descarte de "*Frames*";
- Controles de "*Feedback*";
- Controle de Prioridade de Perda de Células;
- "*Traffic Shaping*".

3. Descrição do ÁTILA

Este trabalho questiona a eficiência do modelo de gerência utilizado em sistemas tradicionais para um ambiente de redes ATM. Em conseqüência, duas abordagens podem ser adotadas na solução da gerência de redes ATM.

- Melhoria do modelo existente, adicionando novas funcionalidades e pequenas alterações na sua estrutura, sem contudo alterar as características básicas do paradigma adotado.
- Questionamento da adequabilidade dos padrões de gerência atuais, resultando, provavelmente, na redefinição total da estrutura de gerência.

As abordagens de gerência ATM citadas no item 2.2, referem-se a proposta relacionada a melhoria dos modelos existentes, descrevendo adaptações e extensões do paradigma Gerente/Agente. A arquitetura proposta, ÁTILA, é um esforço que segue na direção da segunda proposta, ou seja, na redefinição total da estrutura de gerência.

3.1 Arquitetura Funcional

3.1.1 Diagrama da Arquitetura Funcional

A "Arquitetura Funcional" do ÁTILA (Figura 01) é representada por um conjunto de blocos funcionais, onde cada bloco realiza suas funções específicas relativas ao tratamento e processamento de informações de gerência. Além disso, é apresentado um conjunto de interfaces que definem as operações disponíveis aos blocos funcionais. Estas interfaces tem

como objetivo prover a troca, acesso e manipulação de informações entre blocos adjacentes. Elas definem pontos conceituais que representam os limites entre os blocos funcionais, com o propósito de identificar o tipo de informação que é trocada entre estes blocos. Estas interfaces são baseadas nos conceitos de orientação a objeto. Portanto, as mensagens trocadas manipulam objetos. Além dos objetos também estão definidas as operações válidas nestas interfaces.

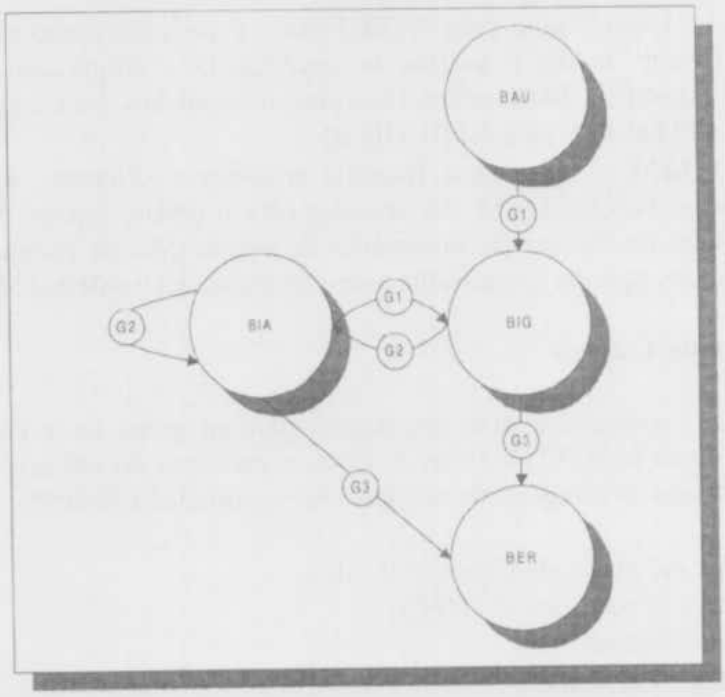

Figura 01 : Arquitetura Funcional do ÁTILA

3.1.2 Blocos Funcionais

- **BAU (Bloco Acesso ao Usuário)**

Este bloco tem como principal objetivo disponibilizar uma interface de acesso e manipulação às informações de gerência da rede ao usuário, através de um conjunto de operações pelas quais este pode interagir com o ambiente a ser gerenciado.

Funções de tratamento e apresentação de informações, devem ser disponibilizadas com o objetivo de fornecer uma interface "amigável" de visualização da rede. Com o objetivo de prover aos usuários a gerência de suas redes a partir de diferentes localidades, outra importante função é a facilidade de acesso ao sistema, obtida através de independência de plataforma. O BAU possui interface com o bloco Inteligência do Gerente (BIG).

- **BIG (Bloco Inteligência do Gerente)**

Este bloco processa e armazena informações globais relacionadas com o gerenciamento de redes para suportar, coordenar e controlar a realização das várias funções de gerenciamento.

As informações globais citadas acima, disponibilizam ao usuário uma visão ampla do ambiente gerenciado. Estas informações permitem a realização de avaliações e tomada de decisões, em função do processamento inteligente. Este é um dos fatores que caracteriza o comportamento pró-ativo da arquitetura proposta. Este bloco possui interfaces com os blocos Acesso ao Usuário (BAU), Inteligência do Agente (BIA) e Elemento de Rede (BER), descritos a seguir.

- **BIA (Bloco Inteligência do Agente)**

Este bloco realiza o monitoramento e o processamento local de informações relacionadas com o gerenciamento de um elemento de rede, com o objetivo de executar as funções de gerenciamento a este nível.

O BIA é baseado na distribuição das funções de gerência. A análise das informações e das mensagens a serem passadas ao BIG através da interface, é realizada de forma distribuída caracterizando os vários níveis de processamento e de inteligência.

Problemas podem ser previstos, analisados e solucionados localmente ao elemento de rede, em função da distribuição da inteligência junto a estes elementos e da garantia de acesso às informações atualizadas em virtude da proximidade das fontes de informação. Estas características fornecem subsídios para o comportamento pró-ativo da arquitetura também a nível de elemento de rede. Este bloco possui interfaces com os blocos Inteligência do Gerente (BIG) e Elemento de Rede (BER).

- **BER (Bloco Elemento de Rede)**

Este bloco representa as funções dos recursos de rede relacionadas ao gerenciamento. Ele disponibiliza as funções que permitem o monitoramento e o controle destes recursos, de forma a fornecer as informações necessárias ao gerenciamento e atuar nos elementos físicos da rede.

Este bloco possui interfaces com os blocos Inteligência do Gerente (BIG) e Inteligência do Agente (BIA).

3.1.3 Interfaces

- **Interface "G1"**

A interface "G1" encontra-se entre os blocos funcionais "BAU" e "BIG" e entre o "BIA" e o "BIG". Ela disponibiliza funções que possibilitam o acesso/manipulação das informações de gerência contidas no "BIG" aos blocos BAU e BIA. Estas funções, tem como objetivo, tanto fornecer ao usuário o suporte necessário ao desenvolvimento de aplicações de gerência, como possibilitar a troca de informações e mensagens entre os blocos "BIA" e "BIG".

Esta troca de informações e mensagens permite aos elementos do BIA interagirem com os elementos do BIG, com objetivo de obter informações gerais da rede, podendo assim, ter uma visão ampla da rede e desempenhar suas funções de maneira pró-ativa.

- **Interface "G2"**

Esta interface encontra-se entre os blocos funcionais "BIG" e "BIA" e entre os elementos que compõe o "BIA". Em virtude do caráter pró-ativo e distribuído da arquitetura, o monitoramento e controle junto ao elemento de rede, pode ser requisitado pelo BIG aos elementos do BIA. Estas requisições são realizadas através das operações disponibilizadas nesta interface. Isto visa a descentralização das atividades de gerência.

A interação entre os elementos do BIA, é realizada em função da necessidade de haver cooperação tanto de informações como de inteligência para análise, prevenção e solução de problemas localmente ao elemento de rede.

- **Interface "G3"**

A interface "G3" encontra-se entre os blocos funcionais "BIG" e "BER", e entre o "BIA" e o "BER". Ela permite o controle e monitoramento dos elementos de rede pelo "BIA" e "BIG". É através desta interface que se tem acesso aos elementos de rede propriamente dito, e às funcionalidades e informações de gerência que eles disponibilizam.

3.2 Arquitetura Física

3.2.1 Diagrama da Arquitetura Física

A "Arquitetura Física" (Figura 02) define os elementos físicos e as interfaces que os interligam. Estes elementos constituem os blocos que representam as implementações das funcionalidades definidas nos blocos contidos na "Arquitetura Funcional".

Esta arquitetura descreve a distribuição das funcionalidades, refletindo o caráter distribuído do ÁTILA. Ela agrupa os blocos funcionais em entidades físicas visando atingir requisitos de flexibilidade. Um elemento físico pode implementar mais de um bloco da arquitetura funcional e as interfaces desta arquitetura eqüivalem as interfaces da arquitetura funcional.

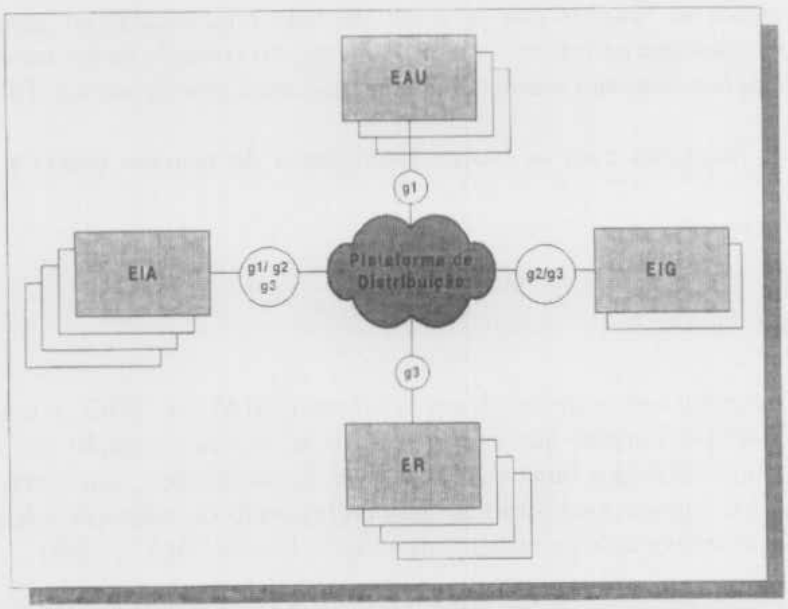

Figura 02 : Arquitetura Física do ÁTILA

3.2.2 Elemento Físicos

- **EAU (Elemento Acesso ao Usuário)**

Correspondendo ao "BAU", este elemento físico realiza o tratamento das informações disponibilizadas pelo "BIG" e apresenta-as de forma "amigável" aos usuários de informações de gerência. Existem diversas tecnologias que podem ser utilizadas na implementação destas funções. Dentre elas temos:

- Interfaces proprietárias que são acessadas a partir de um cliente específico e se utilizam de linguagens de programação gráficas como: Delphi, Visual Basic, PowerBuilder, etc.;
- Browsers WWW (Applets JAVA, HTML, CGI, etc.).

- **EIG (Elemento Inteligência do Gerente)**

Correspondendo ao "BIG", este elemento físico deve conter dentre outras funcionalidades, o armazenamento das informações de gerência, o tratamento inteligente destas informações e uma interface simples e funcional de acesso/manipulação deste elemento.

O EIG deve prover as alternativas de centralizar ou distribuir as funções globais, as quais incluem suporte às aplicações, funções de banco de dados, análise de informações, formatação de dados, relatórios, etc.

As tecnologias vigentes que podem implementar as referidas funções são :

- Sistemas Especialistas [Rocha96];
- Redes Neurais [Vieira96];
- Banco de Dados Ativo [Hasan96][Vasconcelos97], etc.

- **EIA (Elemento Inteligência do Agente)**

Este elemento físico que corresponde ao bloco funcional "BIA", deve ter autonomia para realizar monitoramento e controle dos recursos da rede, a partir de uma visão global disponibilizada por outros elementos do "EIA", e pelo "EIG".

Em função da complexidade destas funções, cada vez mais utiliza-se mecanismos inteligentes para tal tarefa. Para implementar este elemento pode-se utilizar:

- Sistemas Especialistas [Artola96];
- Redes Neurais [Vieira96];
- Agentes Inteligentes [Gaiti93].

- **ER (Elemento de Rede)**

O elemento ER corresponde ao bloco funcional "BER", e desempenha as funções de gerência nos elementos de rede. Dentre estas funções temos: armazenamento de informações locais de gerenciamento, controle destas informações e a interação com o recurso físico gerenciado.

Existem atualmente dois padrões que disponibilizam estas funções, que são :

- CMIS/CMIP (Common Management Information Service/Protocol) [Stallings93];
- SNMP (Simple Network Management Protocol) [Stallings93].

- **Plataforma de Distribuição**

Esse elemento físico representa o elo de ligação entre os demais elementos da arquitetura. Através dessa plataforma, ocorre troca de informações, usuários acessam/manipulam informações sobre o sistema, etc.

Várias tecnologias podem ser utilizadas para implementar esse elemento, dentre elas destacam-se as arquiteturas :

- ANSA [APM93] [Marshak91];
- CORBA [Siegel96].

3.3 Fluxo Dinâmico de Informações do ÁTILA

Uma das grandes contribuições do ÁTILA, evidenciado em sua arquitetura funcional, é a utilização de inteligência e distribuição no gerenciamento pró-ativo de redes ATM. Essa característica é atingida pela interoperação de seus blocos funcionais.

Esta distribuição permite uma flexibilidade não existente nos atuais modelos de gerência ATM vistos na seção 2.2. Assim, os blocos BIG e BIA engendram um processo de cooperação que agrega ao ÁTILA uma gama de possibilidades de interação no processo da atividade de gerência.

Esta gama de interações, ou fluxo dinâmico de informações, favorece, como será visto a seguir, a pró-atividade do ÁTILA.

A figura 03 apresenta o diagrama de fluxos de informação do ÁTILA. A associação destes fluxos formam um conjunto de possibilidades de interação entre os elementos gerentes e agentes do processo de gerência através dos blocos funcionais inteligentes (BIG e BIA), o que constitui uma importante característica do ÁTILA.

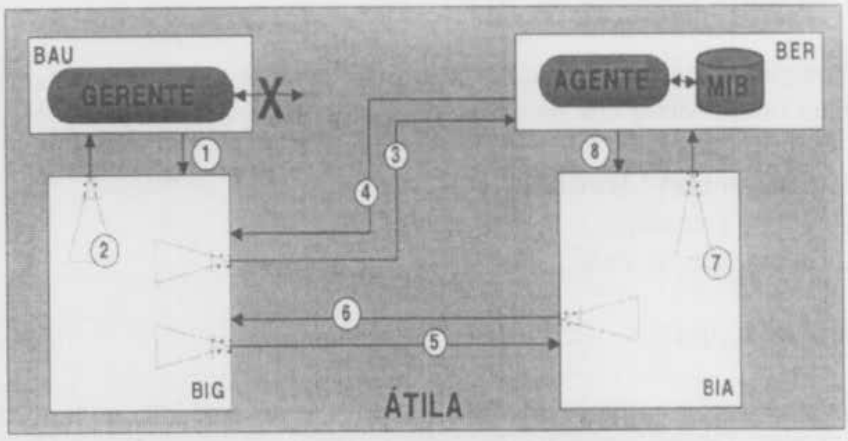

Figura 03 : Diagrama do Fluxo Dinâmico de Informações do ÁTILA.

Na tabela 01 são descritos os fluxos de informação apresentados na figura 03 e as restrições das interações entre os blocos da arquitetura funcional. Estas restrições são intrínsecas a concepção da arquitetura e dizem respeito às operações disponíveis entre os blocos funcionais, representadas na arquitetura funcional pelas interfaces entre esses blocos.

4. Implementação

Nesta seção é apresentada a implementação do ÁTILA, as justificativas das tecnologias utilizadas e o detalhamento das ferramentas que implementam as referidas tecnologias. Para finalizar, é descrito o protótipo, onde são apresentadas as peculiaridades de implementação de cada um dos elementos que o compõe.

4.1 Tecnologias Selecionadas

A figura 04 apresenta uma visão geral do ambiente de implementação do ÁTILA. Nesta figura se destacam, o WWW, o Banco de Dados Ativo, os Agentes Inteligentes (AIs), o SNMP e o CORBA, que são as tecnologias escolhidas para a implementação de cada elemento da

arquitetura física : EAU (Elemento Acesso ao Usuário), EIG (Elemento Inteligência do Gerente), EIA (Elemento Inteligência do Agente), ER (Elemento de Rede) e Plataforma de Distribuição, respectivamente. Em seguida é justificada a utilização de cada uma destas tecnologias para a implementação do protótipo.

De	Para	Restrições das Interações	Fluxos
BAU (Bloco Acesso ao Usuário)	BIG	Sem restrições	1
	BIA	Deve sempre passar pelo BIG	1 → 5
	BER	Sem passar pelo BIA	1 → 3
		Passando pelo BIA	1 → 5 → 7
BIG (Bloco Inteligência do Gerente)	BAU	Sem restrições	2
	BIA	Sem restrições	5
	BER	Sem passar pelo BIA	3
		Passando pelo BIA	5 → 7
BIA (Bloco Inteligência do Agente)	BAU	Deve sempre passar pelo BIG	6 → 2
	BIG	Sem restrições	6
	BER	Sem restrições	7
BER (Bloco Elemento de Rede)	BAU	Passando somente pelo BIG	4 → 2
		Passando pelo BIG e BIA	8 → 6 → 2
	BIG	Sem passar pelo BIA	4
		Passando pelo BIA	8 → 6
	BIA	Sem restrições	8

Tabela 01 : Fluxos e Restrições de Interação dos Blocos Funcionais

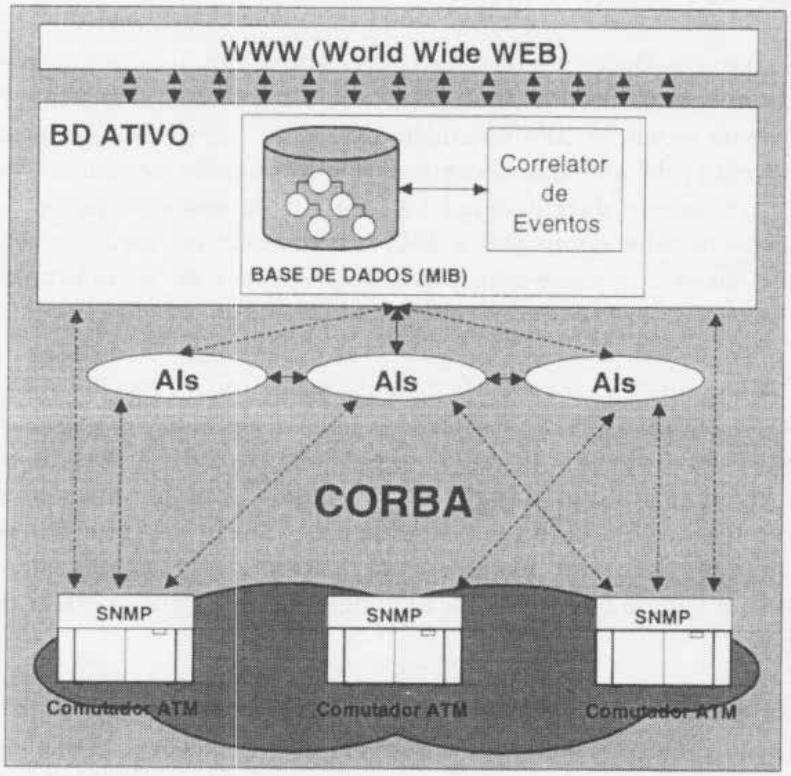

Figura 04 : Ambiente de Implementação

4.1.1 EAU (Elemento Acesso ao Usuário)

As possíveis formas de implementação deste elemento, citadas na arquitetura física, devem levar em conta não só a simplicidade/expressividade da apresentação das informações de gerência, como também a disponibilidade/independência deste elemento. Em função destas características, este elemento será implementado utilizando-se WWW (*World Wide Web*).

Outras características que nos levaram a utilização de WWW no protótipo foram :

- Protocolos abertos : Tanto o SNMP como o HTTP são protocolos da família TCP/IP. Eles são padronizados pelo IETF, e são largamente aceitos e empregados.
- Acesso remoto : Servidores WEB garantem acesso a partir de qualquer *host*, assumindo que um "*browser*" WEB esteja disponível, ao contrário de uma interface implementada em uma linguagem de programação gráfica, que depende da plataforma, ou seja, que requer um cliente específico instalado.

4.1.2 EIG (Elemento Inteligência do Gerente)

Dentre as opções citadas na arquitetura física sobre as tecnologias que implementam as funções do elemento físico EIG, optou-se por Banco de Dados Ativo (BDA). Uma das principais motivações foi a experiência obtida com o uso desta tecnologia em trabalhos anteriores [Vasconcelos97] e a certificação de que esta tecnologia apresenta bem fundamentadas as funcionalidades necessárias para a implementação do protótipo.

No ambiente proposto em [Vasconcelos97], o Sistema de Banco Dados Ativos é apresentado como uma "solução" para os problemas de gerência em um ambiente ATM. Esta abordagem surgiu em virtude dos sistemas de banco de dados utilizados atualmente na gerência de redes não possuírem as funcionalidades e a integração desejada com o sistema de gerência, e consequentemente não serem adequados à gerência pró-ativa. Em função dos fatos citados acima, não foram utilizadas outras tecnologias.

Os Sistemas de Banco de Dados Ativo [Dayal89], possuem o seu comportamento ativo especificado através de regras *Evento-Condição-Ação* (E-C-A). A semântica destas regras é muito simples. O sistema monitora continuamente os eventos especificados; uma vez detectada a ocorrência de um evento relevante a uma regra, avalia a condição associada a esta regra e, se a condição é verdadeira, o sistema executa a ação da regra. Este ambiente permite que as funções de gerência sejam especificadas como regras E-C-A, provendo *um mecanismo unificado para gerência de dados e eventos e automatização dos processos de monitoramento e controle* [Hasan96].

O Banco de Dados Ativo, através da especificação de regras, provê mecanismos inteligentes de filtragem/correlação de informações, mensagens de falha e alarmes e sugere a ação apropriada para a rápida resolução do problema. Como exemplos temos: o BDA detecta automaticamente gargalos na rede; sugere reconfiguração da rede virtual; pesa inteligentemente numerosas mensagens de falha e toma ações corretivas automáticas ou provê recomendações via console; utiliza dados do mundo real para planejamento eficiente de necessidade de recursos e justificativa de custos; e sugere alterações na topologia da rede para otimizar o desempenho e iniciar novos serviços.

Pode-se também se utilizar da funcionalidade de herança de regras. Esta funcionalidade disponibiliza uma visão abstrata e hierárquica da rede, possibilitando o tratamento (solução) de problemas em vários níveis, dependendo do grau de importância, dificuldade ou criticidade do

mesmo. Outra importante característica é a reusabilidade destas regras por diferentes aplicações.

Dentre as funções que podem ser desempenhadas pelo BDA no ÁTILA, destacam-se:

- A ativação/desativação de células OAM, que auxiliam no gerenciamento de uma rede ATM, pode ser realizada a partir da execução de regras no BD Ativo, dando ao sistema de gerência a capacidade de realizar testes e monitoramento de desempenho e falha na rede sem a interferência do gerente.
- Monitoramento da rede de forma mais dinâmica através de *poolings* assíncronos. A partir das regras do BDA, podem-se especificar critérios, baseados no comportamento da rede, que definam de forma dinâmica, a caracterização de limiares e a execução de *poolings*. Ou seja, a partir das regras e das informações da rede, pode-se saber quais são as áreas de maior criticidade na rede, com o objetivo de definir *poolings* mais constantes para estas.
- Através das regras do BDA pode ser especificado um *baseline* dinâmico, onde os valores são periodicamente analisados/tratados com o objetivo de fornecer ao sistema uma visão mais real da rede, propiciando assim a gerência pró-ativa.

O BDA no ÁTILA segue uma determinada seqüência de passos com o objetivo de identificar, analisar e possivelmente solucionar ou apresentar soluções a problemas que surgiram ou surgirão na rede, agindo assim de maneira pró-ativa. A seguir, é apresentado um exemplo do comportamento do BDA em relação a gerência de *FALHA*.

- Correlacionar múltiplos alarmes;
- Analisar alarmes e notificações;
- Cadastrar alarmes;
- Iniciar e executar testes;
- Analisar os resultados dos testes e reportar as informações.

4.1.3 EIA (Elemento Inteligência do Agente)

A técnica escolhida para a implementação do protótipo foi a de Agentes Inteligentes (AI), baseada em Sistemas Multi-Agentes (MAS). Esta técnica difere das técnicas "tradicionais" de resolução, em especial, a de Sistemas Especialistas, face as seguintes características adicionais: *modularidade dos agentes*, *distribuição do conhecimento* e *adaptação às mudanças exigidas pelo sistema*.

No ÁTILA, os Agentes Inteligentes podem ser no entanto, abstraídos como um SBC ("sistema baseado em conhecimento") com características adicionais (*percepção*, *comunicação*, *adaptação* e *distribuição)* (Figura 05). Características estas, que somadas define-se como o conceito de agente, como descrito em [Sichman92].

Dessa maneira, distribui-se o conhecimento por toda a rede e ataca-se o problemas em diferentes módulos, ou seja, "agentes racionais", que apoiam, analisam e executam tarefas de gerência usando mecanismos de *cooperação*, *competição*, *coabitação* ou *distribuição* [Demazeau90].

Em [Sichman92] é apresentado o conceito de agentes utilizado neste trabalho, os quais possuem as seguintes caraterísticas :

- Agentes são autônomos no sentido que eles podem definir suas próprias metas e planos internos;

- Eles são capazes de participar de complexas interações, utilizando-se de altos níveis de domínio em primitivas de comunicações independentes;
- Agentes são independentes de uma arquitetura particular. De fato, os agentes podem definir ou mudar a arquitetura tal qual a atividade de solução é executada;
- Eles são capazes de perceber no ambiente do qual estão inseridos, mudar seu comportamento e incorporar tais mudanças ao seu modelo interno;
- Eles são aptos a adicionar as habilidades e metas de outros agentes com a finalidade de resolver um problema cooperativamente;

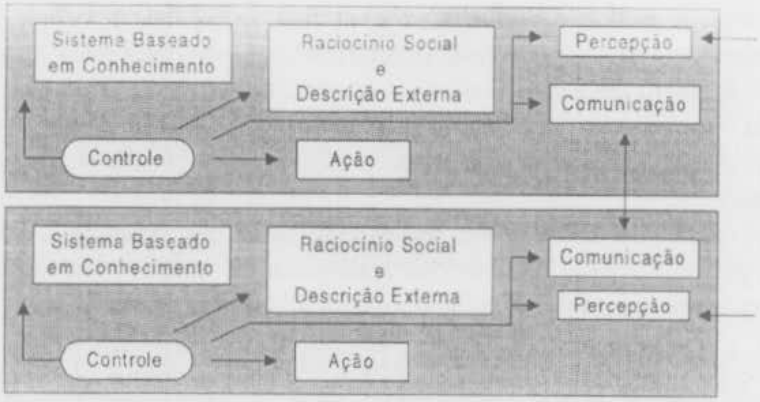

Figura 05: De Sistemas Baseados em Conhecimento à Agentes

4.1.4 ER (Elemento de Rede)

As funções de gerência desempenhadas pelo elemento de rede, restringem a escolha desta solução em dois ambientes : SNMP e CMIS/CMIP. Será utilizado o SNMP por ser um padrão de mercado, ou seja, a maioria dos equipamentos atualmente suporta este protocolo.

4.1.5 Plataforma de Distribuição

O CORBA apresentou-se como a solução mais adequada para a implementação dos mecanismos de distribuição do ÁTILA. Dentre os motivos merecem destaque: suporte à comunicação transparente entre objetos distribuídos e utilização de um modelo de interface único, descrito pela linguagem IDL. Para disponibilizar as operações desses objetos, o CORBA fornece mecanismos para a distribuição em sistemas heterogêneos indispensáveis ao ambiente do protótipo. Um outro aspecto considerado na opção pelo CORBA, diz respeito à disponibilidade de um ambiente para o desenvolvimento rápido e fácil de aplicações complexas, o ÁBACO [Souza96].

4.2 Ferramentas Utilizadas

Abaixo são descritas as ferramentas utilizadas para implementar as tecnologias utilizadas no protótipo. Para a implementação da interface WWW foi utilizada a linguagem JAVA, o banco de dados ativo escolhido foi o ODE (*Object Database Environment*) desenvolvido pela AT&T Bell-Labs. Para a implementação dos agentes inteligentes foi utilizado o ambiente JATLite. Para acesso aos agentes SNMP foi utilizada a API-SNMP/Java Advent, da AdventNet. A implementação da plataforma de distribuição CORBA utilizada foi o ÁBACO.

4.2.1 Linguagem Java - WWW (World Wide Web)

A escolha da tecnologia WWW para implementar o Elemento Acesso ao Usuário (EAU), levou a decisão entre HTML/CGI e "*Applets Java*". Embora HTML/CGI não seja uma ferramenta, ela

foi introduzida nesta seção devido ao costume desta tecnologia ser comparada com ambientes JAVA. Abaixo, são citadas as características de cada uma destas "ferramentas" e implicitamente é definida a escolha por "*Applets Java*".

CGI provê um método para gerar páginas WEB dinamicamente. Entretanto, a carga de processamento fica toda restrita ao lado do servidor WWW. Já que o cliente não pode controlar a execução dos binários CGI, seu escopo é usualmente limitado a transações simples como consultas a banco de dados.

JAVA é independente de arquitetura, ou seja, pode ser interpretado em qualquer máquina que possua uma máquina virtual JAVA (virtual machine). Isto permite a implementação de código móvel como parte de código HTML (Hyper Text Markup Language) (*Applets Java*). Web browsers que possuem uma máquina virtual podem fazer um "*download*" e processar "*Applets Java*".

Outro fator importante para a escolha de JAVA, deve-se ao fato de outras ferramentas utilizadas no protótipo, no caso API SNMP da Advent e JATLite também se utilizarem de JAVA.

4.2.2 ODE (Object Database Environment) - Banco de Dados Ativo

ODE [Gehani91] é um sistema de banco de dados baseado no paradigma de objeto. Ele utiliza a linguagem O++, que é uma extensão da linguagem C++, para definir, consultar e manipular seus objetos. O++ prove facilidades para se criar objetos persistentes, visualizando assim a memória em duas partes: volátil e persistente. O ODE possui facilidade para associarmos Restrições e "Triggers" com objetos. Restrições e "Triggers" são associados as definições das classes, o que faz com que elas sejam fáceis de ler, implementar e de se utilizar da funcionalidade de herança disponibilizada na linguagem O++. As funcionalidades "Triggers" e Restrições (*Constraints*) caracterizam ODE como um banco de dados ativo.

ODE utiliza um modelo de regras do tipo E-A, que é mais simples do que o modelo E-C-A. Este modelo além de eliminar a necessidade de uma parte C (condição) separada – a condição é parte de um evento neste modelo – elimina também a necessidade de se ter tipos especiais de acoplamento como os propostos no modelo E-C-A.

Neste Banco de Dados podemos especificar eventos compostos como expressões de eventos, usando uma linguagem de operadores de eventos. Esta linguagem é equivalente, em termos de expressividade, a expressões regulares sobre "strings" ou eventos lógicos. Esta ferramenta compila especificações de eventos compostos arbitrários em autômatos finitos, tornando a detecção destes eventos eficiente.

4.2.3 JATLite - Agentes Inteligentes

O JATLite [JATLite] é um ambiente baseado na família de protocolos TCP/IP que fornece facilidades na construção de sistemas multi-agentes usando um conjunto de APIs Java. Ele está estruturado em quatro camadas: *Abstract Layer, Base Layer, KQML Layer, Router Layer*. A partir destas camadas é possível implementar agentes com vários níveis de abstração.

A partir da camada *Router*, é permitida a comunicação entre os agentes, e são disponibilizadas aos agentes o conhecimento das características dos demais. A partir da camada KQML, o JATLite provê toda a funcionalidade para que agentes possam se comunicar em alto nível.

Dentre os fatores que nos levaram a utilizar a ferramenta JATLite, temos : a sua disponibilidade ("*freeware*"), experiência com sua utilização em trabalhos anteriores e devido ao fato dela se

basear na linguagem JAVA.

4.2.4 Advent API SNMP/JAVA - SNMP

Advent [Advent] é um pacote que contém um conjunto de classes JAVA. Ele suporta a versão 1 do SNMP e inclui um analisador que permite carregar múltiplos módulos da MIB em qualquer tempo, tanto para aplicações quanto para *applets*. O Advent foi planejado, principalmente, para o desenvolvimento de gerentes SNMP e aplicações de gerenciamento. O pacote está designado a habilitar o desenvolvimento de *applets* JAVA orientado a objetos, e aplicações JAVA que usam SNMP para acessar os nós gerenciados. No caso de *applets*, suporte especial é provido para garantir as restrições de segurança.

Advent provê um programa JAVA para o servidor WWW chamado "SNMP APPLET SERVER" (SAS), que permite o *applet* enviar e receber pacotes SNMP para e de qualquer dispositivo gerenciado acessível a partir de um *applet host*.

O pacote é composto por 4 categorias de classes: variáveis SNMP, comunicação SNMP, MIB SNMP relacionadas e classes mistas. Além disso, existem três classes que não estão incluídas nas categorias acima, são elas: SnmpClient; MibException; SnmpException.

4.2.5 ÁBACO - CORBA

O ÁBACO é um ambiente com suporte gráfico que serve de apoio ao desenvolvimento de aplicações complexas no CORBA. Esse ambiente fornece uma interface amigável na qual aplicações distribuídas podem ser construídas rapidamente e sem a necessidade de nenhum conhecimento do CORBA por parte do desenvolvedor. Toda e qualquer funcionalidade relativa a comunicação é deixada totalmente de lado, cabendo ao desenvolvedor a tarefa de implementar somente a lógica da aplicação. Afora os motivos citados acima, a experiência adquirida com a utilização deste ambiente em [Souza96], foi a principal razão da sua escolha para a implementação do protótipo.

Duas linguagens de programação foram desenvolvidas para a implementação do ÁBACO: a CDL (*Component Description Language*), que tem como objetivo principal criar objetos distribuídos configuráveis, denominados componentes no ÁBACO, a partir de objetos distribuídos ordinários; e a linguagem CCL (*Component Configuration Language*), utilizada na descrição da configuração de aplicações complexas a partir de objetos configuráveis programados em CDL.

Modelo de Componentes

No ÁBACO, os objetos tem uma estrutura na qual serão contempladas tanto as características dos objetos distribuídos, quanto as características dos objetos do paradigma de configuração. A utilização desta estrutura híbrida, provém da necessidade desses objetos serem reconhecidos pelas plataformas de origem. Desta forma estes objetos continuam utilizando todos os mecanismos dessas plataformas, além de passarem a utilizar também as vantagens do paradigma de configuração.

A figura 06 ilustra um objeto do ambiente ÁBACO, denominado **componente.**

Um componente do ambiente ÁBACO tem a seguinte estrutura:

Estrutura interna do componente :

- uma *interface* com a descrição das operações implementadas pelo objeto.

Estrutura externa do componente :

- um conjunto de portas representando as operações *requeridas* pelo componente (portas de saída);
- um conjunto de portas representando as operações *oferecidas* pelo componente (portas de entrada) e
- um conjunto de ligações entre as portas de entrada do componente e as operações descritas na *interface* do objeto.

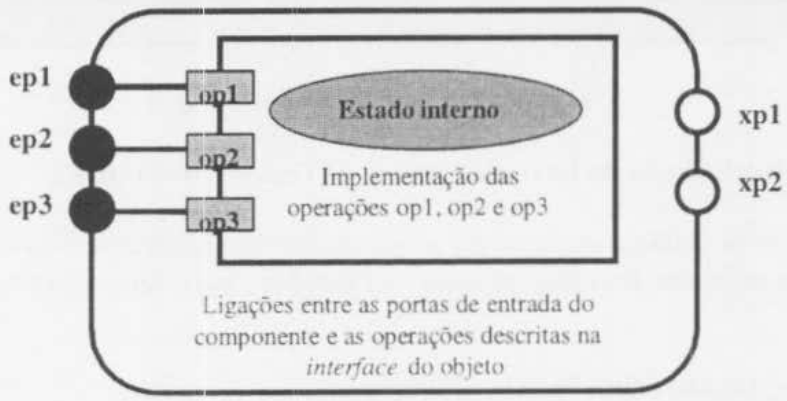

Figura 06 : Estrutura do Componente do Ambiente ÁBACO

4.3 Descrição do Protótipo

Nesta seção são apresentadas as especificações do ambiente do protótipo. Primeiramente, é detalhada a interface gráfica, onde são descritas as funções disponibilizadas por esta. Também são descritos os objetos relativos ao banco de dados ativo (BDA) ODE e as respectivas interfaces IDL responsáveis pelo acesso e manipulação das informações de gerência que ele disponibiliza. A seguir, é descrito a especificação em JATLite de um agente genérico, cuja estrutura é herdada por todos os agentes implementados no protótipo. Para finalizar, é definida a estrutura dos agentes SNMP e a suas interfaces IDL; e um exemplo da especificação de um objeto do protótipo como um componente do ÁBACO.

4.3.1 Elemento Acesso ao Usuário – Interface WWW (JAVA)

A interface do protótipo (Figura 07) foi desenvolvida em Java. Neste ambiente os elementos que compõe a rede a ser gerenciada, são tratados como objetos dos tipos : SubMp, Snmp e Pool. Dentre as funções desempenhadas através de "*menus*" ativados a partir destes objetos, podem-se destacar :

- Ativação/desativação/monitoramento do estado e configuração dos Agentes Inteligentes associados a cada objeto (elemento de rede);
- Ativação/desativação/monitoramento das regras associadas ao ODE e Agentes Inteligentes;
- Monitoramento e controle da rede a partir das funções disponibilizadas pelo ODE e pelos Agentes Inteligentes em cada objeto.

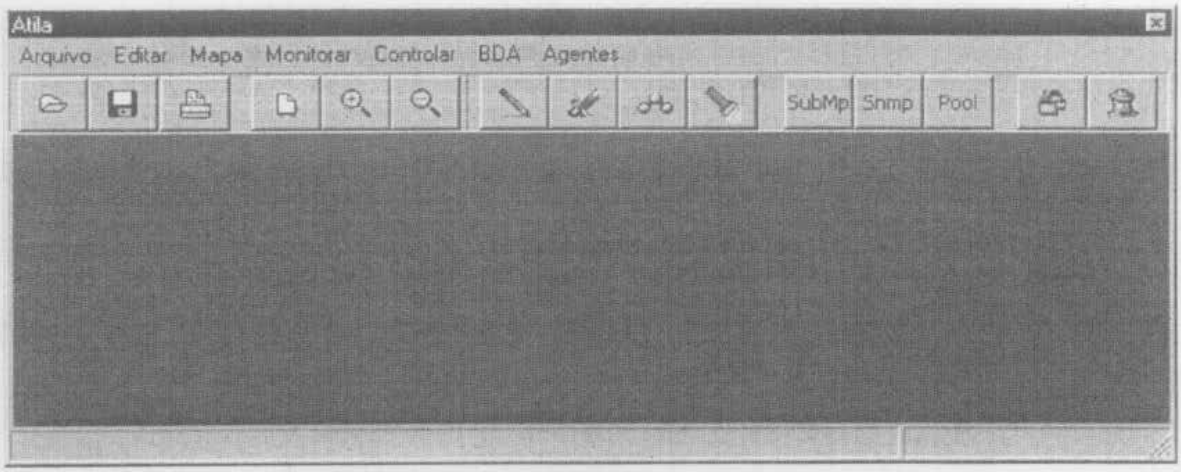

Figura 07: Interface do Protótipo

4.3.2 Elemento Inteligência do Gerente - Banco de Dados Ativo (ODE)

Com a intenção de possibilitar o acesso e a manipulação das informações do ODE, um conjunto de classes foram definidas para implementar as operações necessárias para a realização destas tarefas.

Classe *Banco_de_Dados*: Abrir, Fechar e Renomear;
Classe *Regra* : Criar, Apagar, Habilitar, Desabilitar e Consultar;
Classe *Dado* : Consultar (Simples (Objeto), Cruzar(Join)), Incluir_Objeto, Atualizar_Objeto e Apagar_Objeto.

Para tornar disponíveis essas operações de forma distribuída, interfaces em IDL foram definidas. Através dessas interfaces, qualquer elemento do protótipo pode interagir com o ODE para manipular as suas informações. Para facilitar as especificações, um módulo chamado BDA foi criado, e dentro desse módulo, todas as interfaces de manipulação do banco de dados foram especificadas. Abaixo são definidas as interfaces IDL relativas às classes *Banco_de_Dados* e *Regra*. As IDLs relativas à classe *Dado*, são definidas levando-se em conta a estrutura do banco de dados, ou seja que informações ele armazena. Em virtude disto, as interfaces para esta classe só serão definidas quando da definição desta estrutura.

```
module BDA {

// Definição da interface da classe Banco de Dados
interface Banco_de_Dados {
        boolean Abrir (in string Nome_BD, in long Pemissao);
        boolean Fechar (in string Nome_BD);
        boolean Renomear (in string Old_Nome_BD, in string New_NomeBD);
};
// Definição da interface da classe Regra
interface Regra{
        typedef string ListaRegras[100];
        boolean Criar (in string Classe, in string Nome_Regra, in string Regra);
        boolean Ativar (in string Classe, in string Nome_Regra, in long Duracao);
        boolean Desativar (in string Classe, in string Nome_Regra, in long Duracao);
        string Consultar (in string Classe, in string Nome_Regra);
       ListaRegras Listar (in string Classe);
      };
}
```

4.3.3 Elemento Inteligência do Agente – Agentes Inteligentes (JATLite)

Nossa plataforma utiliza um modelo genérico de agentes inteligentes, ou seja, todos os agentes serão subclasses de um agente denominado **Átila**. O modelo utilizado nesta implementação foi proposto em [Franca97] sendo baseado no modelo apresentado em [Sichman92]. Este modelo está organizado em um hierarquia de agregação de classes. Estas classes estão estruturadas da seguinte forma (Figura 08):

- **ClasseAtilaControle** : Esta classe abstrata apresenta as regras do comportamento básico de relacionamento com outros agentes (*métodos de políticas de comportamento*); o processamento de mensagens (*métodos de política de mensagens*); o controle e a funcionalidade dos serviços disponíveis apresentados em cada agente (*métodos de política de serviço e política de decisão*).
- **ClasseAtilaInformacoes** : Esta classe abstrata define as informações locais para cada agente (modelo interno) e faz o tratamento das informações do ambiente (modelo externo). Esta classe é responsável pela base de conhecimento dos agentes, relacionada ao conhecimento local e ao conhecimento global.
- **ClasseAtilaMensagens** : Esta classe é responsável pela comunicação entre os agentes na plataforma utilizada em nosso ambiente. Utiliza-se do mecanismo de troca de mensagens entre objetos (*caixa de mensagem*) e definições das interfaces (*portas comunicação*). Esta classe também suporta a comunicação entre os agentes e outros elementos do protótipo.

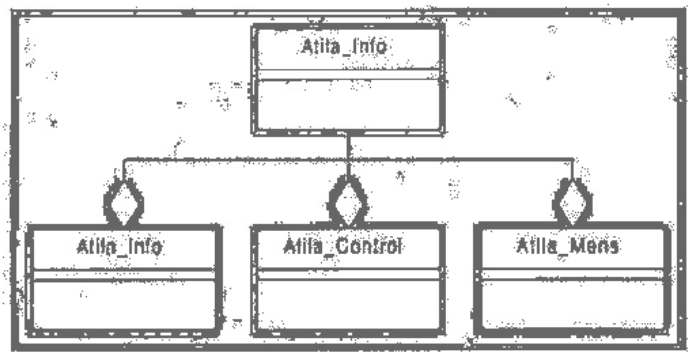

Figura 08: Hierarquia do Agente Átila

Especificação em JATLite:

```
import router.*;
class ag-atila extends router {
    private atila_info object_info;
    private atila_control object_ control;
    private atila_mens object_mens;
    public AtivaAgente() : Boolean ;
    public DesativaAgente() : Boolean;
    public AdicionaRegra (Regra: AtRegra);
    public RemoveRegra;
    public RealizaConsulta;
    public SubmeteReq (TpR: AtRequisicao; DescricaoR : AtC AtRequisicao) : AtRequisicao;
    public EnviaMens (TipoM : AtMensagem; DescricaoM : AtCMensagem) : AtMensagem;
    public RecebeMens (TipoM : AtMensagem; DescricaoM : AtCMensagem);
}
```

4.3.4 Elemento de Rede – SNMP (API Advent Java/Snmp)

As operações necessárias para o monitoramento e controle dos elementos de rede, são realizadas nesse protótipo do Átila por agentes SNMP. Para que essas operações pudessem estar disponíveis tanto para o acesso dos agentes inteligentes como para o ODE, elas foram descritas em IDL.

```
// Definição em IDL das interfaces dos agentes SNMP
interface SNMP{
    string get(in string host, in string OID);
    boolean set(in string community, in string host, in string OID, in string Val);
};
```

4.3.5 Plataforma de Distribuição – CORBA (ÁBACO)

A comunicação entre os objetos do protótipo é realizada através de interfaces IDL. Na implementação desses objetos, foi utilizado o ÁBACO. Esse ambiente introduz um estilo de programação orientada a componentes no CORBA, tornando transparente a utilização dessa arquitetura no desenvolvimento de aplicações distribuídas. Ou seja, os objetos que implementam os agentes inteligentes, o ODE e os agentes SNMP, são criados como componentes do ÁBACO e suas relações de comunicação são mapeadas como conexões automaticamente geradas nesse ambiente. Na figura 09 abaixo é apresentado um exemplo de um componente referente a um agente SNMP.

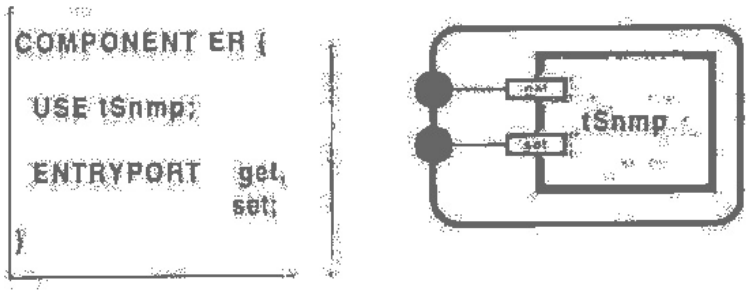

Figura 09: Componente tSnmp

4.3.6 Funcionamento Geral do Protótipo

O Banco de Dados Ativo utilizado no protótipo, o ODE, contém uma base de dados estruturada, com informações de gerenciamento de uma rede ATM. Além destas informações, o ODE contém regras que implementam a sua inteligência e disponibiliza ao protótipo regras "padrões". Estas podem ser referenciadas pelas aplicações de gerenciamento ou serem reutilizadas com o objetivo de se definir regras mais sofisticadas.

O ODE também disponibiliza um conjunto de operações de manipulação de banco de dados, que podem ser utilizadas tanto pelos desenvolvedores de aplicação quanto pelos agentes inteligentes (AIs). Estas operações são especificadas em IDL e estão acessíveis via ÁBACO.

No caso do desenvolvimento das aplicações, os usuários podem especificar as funções de gerência como regras E-C-A, provendo um mecanismo unificado para gerência de dados e eventos e automatização dos processos de monitoramento e controle. Em relação aos AIs, as operações permitem a interação com o ODE, objetivando a atualização/consulta das informações globais de gerenciamento.

O gerenciamento a nível de elemento de rede é realizado pelos AIs. Estes cooperam entre si e com o ODE trocando informações de gerenciamento com o objetivo de diagnosticar e solucionar problemas que possivelmente ocorram na rede.

A interação entre os agentes inteligentes e o ODE tem como objetivo realizar monitoramento e controle inteligente de forma distribuída. Através de regras especificadas no ODE, agentes inteligentes podem ser acionados para realizar as funções de gerenciamento junto aos elementos de rede. Estas funções são desempenhadas de forma inteligente através do conhecimento individual e autônomo presente em cada agente.

Os agentes inteligentes do protótipo são implementados como objetos do JATLite, ou seja a interacão entre eles se dá através desta plataforma. Esta interação é realizada em função da necessidade de haver cooperação tanto de informações como de inteligência. Além disso, um conjunto de interfaces IDL é disponibilizado para permitir a comunicação, através da camada *Router*, entre os agentes e os outros elementos do protótipo.

Sendo a única forma de atuar junto aos recursos gerenciados, o agente SNMP continua tendo uma função passiva nesta arquitetura. O comportamento ativo a ser desempenhado localmente aos elementos de rede é realizado pelos AIs, os quais são auxiliados pelo ODE.

Os agentes SNMP disponibilizam interfaces IDL de acesso, as quais permitem o ODE e os agentes inteligentes interagirem diretamente com os elementos de rede com o intuito de realizar monitoramento e controle.

5. Conclusão

A principal contribuição do ÁTILA, a Arquitetura Inteligente e Distribuída para Gerência ATM proposta neste trabalho, é a tentativa de quebrar o tradicional modelo Gerente/Agente. Assim, o ÁTILA apresenta-se como uma nova arquitetura de gerência, a medida que introduz novos blocos funcionais inteligentes com suporte à distribuição, permitindo um fluxo de informação que bem caracteriza a originalidade da proposta.

O ÁTILA inscreve-se dentro do projeto PROTEM/CNPq denominado GERENTE (Gerenciamento de Redes de Telecomunicações). Trata-se de um ambiente de experimentação para soluções não convencionais de gerência de redes ATM. É resultado de uma dissertação de mestrado defendida no Departamento de Computação da UFC, em Março 1998.

6. Bibliografia

[Advent] "Advent SNMP Package Version 1.2".

[Alexander95] Alexander, Peter e Carpenter, Kacey – "ATM Net Management: A Status Report" - Data Communications - September 1995.

[Artola96] Artola - "Um Sistema Especialista para Gerência Pró-ativa Remota", 14º. SBRC, Fortaleza-Ce, Brasil - Maio 1996.

[Dayal89] McCarthy, D. R. e Dayal, U. - "The Architeture of An Active Database Management System", ACM-SIGMOD, Portland, Oregon - Maio 1989 - Pág. 215-224.

[Demazeau90] Demazeau – "Descentralized Artificial Intelligence" – Elsevier Science Publishers – Holanda – 1990.

[Franca97] França, Mardônio – "Agentes SIM: Uma Abordagem de Gerenciamento Pró-Ativo de Redes de Computadores utilizando Sistemas Multi-Agentes" - IV Encontro de Iniciação Científica, Universidade Estadual do Ceará - Novembro 1997.

[Gaiti93] Gaiti - "Distributed Artificial Intelligence as an AIP technique for Network Management", IFIP Transactions C: Communication Systems. Advanced Information Processing Techiniques for LAN and MAN Management. Versailles, France - Abril 1993.

[Gehani91] Gehani, N. H. e Jagadish, H. V. - "Ode as an Active Database: Constraints and Triggers". 17th Int'l Conf. Very Large Data Bases, Barcelona, Espanha – 1991 - Paginas 327-336.

[Hasan96] Hasan, Masum Z. - "The Management of Data, Events and Information Presentation for Network Management". Phd Thesis, Department of Computer Science - 1996.

[ILMI4.0] ATM-FORUM <af-ilmi-0065.000>, "Integrated Local Management Interface (ILMI) Specification Version 4.0" - Setembro 1996.

[JATLite] "JATLite" - URL: http://java.sanford.edu/JATLiteDoc.html.

[Marshak91] Marshak, David S. - "ANSA - A Model for Distributed Computing". Network Monitor - Guide to Distributed Computing, Novembro 1991.

[Minoli97] Minoli, Daniel e Golway, Thomas – "Planning & Managing ATM Networks", Prentice Hall - 1997.

[Rocha96] Rocha, Marco; Fernandez, Luis e Westphall, Carlos - "Gerência Pró-ativa de Redes de Computadores usando Agentes e Técnicas de Inteligência Artificial"- 14°. SBRC, 1996.

[Sichman 92] Sichman – "When Can Knowledge-based Systems Be Caleed Agents ?" – Simpósio Brasileiro de Inteligência Artificial – Rio de Janeiro, Brasil – 1992.

[Siegel96] Siegel, J. - "CORBA Fundamentals and Programming", Editora Wiley Computer.

[Souza96] Souza, Cidcley T. de – "Um Ambiente para o Desenvolvimento de Aplicações Orientadas à Configuração Utilizando Objetos Distribuídos", Dissertação de Mestrado – Depto. de Computação, Universidade Federal do Ceará, Fortaleza – Ceará –Brasil, 1996.

[Stallings93] Stallings, W. – "SNMP, SNMPv2 and CMIP, The Practical Guide to Network Management Standards", Addison-Wesley Publishing Company,Inc. - 1993.

[Traffic96] ATM-FORUM <af-tm-0056.000>, "ATM Forum Traffic Management Specification Version 4.0" - Abril 1996.

[Vasconcelos97] Vasconcelos, Marcelo V. - "Utilizando Banco de Dados Ativos no suporte ao Gerenciamento Pró-ativo de Redes ATM", II SFBSID'97 - Novembro 1997 - Pág. 347-358.

[Vieira96] Vieira, M. e Sari, Solange - "Prototipação de um Sub-agente Adaptativo Baseado em Redes Neurais", 14°. SBRC, Fortaleza-Ce, Brasil - Maio 1996.