Uma Proposta para Gerência Pró-Ativa em Redes Utilizando Agentes Inteligentes Hierarquicamente Distribuídos

Luis Cesar Dias Morais[1], Celso de Renna e Souza (*in memoriam*), [2]Sérgio Roberto Matiello Pellegrino

[1]Coordenação do Curso de Sistemas de Informação - Universidade Ibirapuera (UNIb)
Avenida Iraí, 297 - Moema
CEP 04082-000 - São Paulo - SP - Brasil

luiscdm@ibirapuera.br

[1, 2]Divisão de Ciência da Computação - Instituto Tecnológico de Aeronáutica (ITA)
Praça Marechal Eduardo Gomes, 50 - Vila das Acácias
CEP 12228-900 - São José dos Campos - SP - Brasil

{luiscdm, pell}@ita.br

Resumo. *Este artigo pretende descrever a criação de uma sociedade de agentes inteligentes hierarquicamente distribuídos, incumbidos de prover gerência pró-ativa a redes de computadores baseadas no padrão Ethernet, quando utilizando protocolos da camada de aplicação. Para que isto fosse possível são descritas as formas de como informações fornecidas pelas MIBs (Management Information Base) RMON2 (Remote Monitoring 2), SNMPv2 (Simple Network Management Protocol Version 2) entre outras foram concatenadas à técnicas de IA (Inteligência Artificial), IAD (Inteligência Artificial Distribuída) e SE (Sistemas Especialistas), que juntamente com a Linguagem Java e o Shell JESS (Java Expert System Shell) solucionaram de maneira eficaz diversos problemas comuns a redes.*

1. Introdução

Já há bastante tempo, a Internet e as redes em geral vêm evoluindo e crescendo em proporções cada vez maiores, em importância e quantidade. Essa situação traz ligada a si a necessidade de melhoria da segurança e da qualidade dos serviços oferecidos, haja vista que, cada vez mais, pessoas e empresas se utilizam desses recursos em seu cotidiano.

A administração de redes é importante e difícil [COMER 2001] e manter o nível de satisfação dos usuários num patamar aceitável exige dos administradores grande conhecimento do funcionamento interno das mesmas, aliado a muita experiência para que se possa evitar ou solucionar rapidamente eventos indesejados.

Com o objetivo de minimizar os incômodos causados pela lentidão, perda de informações ou até mesmo interrupção de serviços, este artigo apresenta uma proposta para prover gerência pró-ativa "inteligente" a redes de computador baseadas no padrão *Ethernet*.

2. Gerência Pró-Ativa

Comumente a abordagem realizada no gerenciamento de redes é a gerência dita reativa. Nela, primeiramente, o problema ocorre para então ser corrigido. Nos dias atuais, essa abordagem tem se tornado quase proibitiva, visto que, além dos prejuízos financeiros que a falha de um serviço pode trazer, há que se levar em conta ainda a perda de tempo e a insatisfação dos usuários.

Várias pesquisas, até então, procuraram medir e associar valores a ocorrências de degradação e interrupção. Pode-se citar o estudo realizado na Universidade de Austin, EUA, o qual demonstrou que o impacto de uma falha da rede produz um decréscimo na receita e aumenta o custo de uma empresa [BRISA 1993]. Conforme esse estudo, o custo de uma falha na rede variava de 2% da receita anual no primeiro dia da paralisação e até cerca de 30% no 30º dia. Já em [INFONETICS 2000] há um outro estudo, realizado também nos EUA, demonstrando que o custo do tempo de inatividade de redes por paradas e degradações de serviço custou às empresas algo em torno de 32,7 milhões de dólares em termos de perda de produtividade de funcionários conectados em rede e em termos de perda de receitas.

O que se pretende, ao implementar gerência pró-ativa, é o não uso de uma abordagem ao problema de forma reativa, passando a uma abordagem onde agentes inteligentes pesquisam parâmetros incomuns a rede em vigilância, prevendo futuras falhas e evitando antecipadamente sua ocorrência. Logo, a gerência pró-ativa é aquela que tem por objetivo a identificação de situações que caracterizam estados que possam vir a degradar os serviços oferecidos [NETO 1997].

Uma definição mais completa também é encontrada em [NOTARE 1999], que a caracteriza pelas reações automáticas do sistema quando uma tendência de degradação é percebida dentro da rede, onde uma mera solução reativa para um dado problema não é suficiente para uma boa performance.

2.2 Agentes SNMP, RMON e RMON2

SNMP é o protocolo de gerenciamento mais utilizado em redes baseadas no modelo TCP/IP, ele utiliza o UDP (*User Datagram Protocol*) como serviço de transporte, sendo o IP utilizado para rotear a troca de informações entre gerentes e agentes. Funciona basicamente, utilizando cinco serviços que são chamados primitivas SNMP, quais sejam, `Get-Request`, `Get-Response`, `Get-Next-Request`, `Set` e `Trap`.

O RMON é uma MIB definida pela IETF (*Internet Engineering Task Force*) nas RFC (*Request for Comments*) 1757, atualizada da RFC 1271 (*Ethernet*) e 1513 (*Token Ring*) que fornece estatísticas da camada MAC (*Medium Access Control*), bem como filtros e captura de pacotes por análise de protocolos. RMON2 é uma extensão da MIB definida pela IETF nas RFCs 2021 e 2074 que adiciona estatísticas de rede e aplicações na pilha de protocolos TCP/IP, permitindo realizar medidas de tráfego até a camada de aplicação [WALDBUSSER 1993, 1997 ; IDDON 1997] aproximando-se, assim, dos conceitos das arquiteturas distribuídas atuais.

Vale ressaltar que tanto o RMON2 quanto o SNMP possuem um grande número de objetos gerenciáveis e, em conseqüência disso, podem fornecer um grande número de parâmetros, o que na prática, é excelente, pois tal monitoração reflete não só a capacidade diversificada dos mesmos, como também sua alta especialização. Porém,

tais fatos implicam que o gerente de redes possua um grau de conhecimento muito elevado sobre todo o seu parque de máquinas.

A escolha correta dos parâmetros a serem monitorados é uma das mais difíceis e importantes atividades de um gerente de redes, visto que uma escolha infeliz pode, além de significar a obtenção de uma informação inútil ao gerenciamento, ocasionar ainda, uma sobrecarga nos canais de comunicação. Isso é reforçado por [CRUZ 1997] que afirma que o gerenciamento deve assegurar a qualidade da rede sem degradá-la.

3. Sistemas de Produção

Sistema de produção é um nome genérico para todos os sistemas baseados em regras de produção, isto é, pares de expressões consistindo em uma condição e uma ação. A idéia inicial dos sistemas de produção foi introduzida por Axel Thue (1863 - 1922) e ficou famosa em 1936, graças a Emil Leon Post [WEIBEL 2006 ; MT 2006].

3.1 Sistemas Especialistas

Sistemas Especialistas (SEs) são as aplicações mais conhecidas dentre as técnicas de IA. São sistemas que armazenam e manipulam o conhecimento especializado de modo a simular o comportamento de um especialista em um domínio específico.

Os SEs são um subconjunto dos Sistemas Baseados em Conhecimento (SBC). Sistemas convencionais são baseados em algoritmos e emitem um resultado final correto, processam um volume de dados de maneira repetitiva, enquanto que um SE é baseado em uma busca heurística e trabalha com problemas para os quais não existe uma solução convencional.

Os SEs são programas que capacitam um computador a auxiliar em processos de tomada de decisão. O *know-how* de um ou mais peritos humanos é utilizado para instruir o computador em como resolver um problema ou tomar uma decisão. A chave para o desempenho de um SE está no conhecimento armazenado em suas regras e em sua memória de trabalho [BITTENCOURT 1998]. Entre as aplicações mais recentes dos SEs está o raciocínio de agentes inteligentes [HILL 2003].

3.1.1 Sistemas Especialistas Hierárquicos

Os Sistemas Especialistas Hierárquicos (SEHs) são uma evolução dos SEs tradicionais. Neles são incorporadas habilidades de cooperação e interação com compartilhamento de conhecimento e dados. Isso permite o tratamento e controle de problemas de forma distribuída, possibilitando alcançar uma meta final pela interação de diferentes entidades. Para que o modelo seja viável, no entanto, é necessário também que suas entidades obedeçam a um controle hierárquico previamente estabelecido. Esse controle foi obtido com o auxílio da ferramenta MOSES (Modelo para Sistemas Especialistas Hierárquicos) [SILVA 2000]. O ambiente de testes, descrito a seguir foi populado com Agentes Especialitas Hirárquicos (AEHs), que têm suas formas de operação detalhadas na subseção 4.1.

Para que se possa pensar em hierarquia é necessário entender que uma estrutura hierárquica prevê comunicação e interação entre elementos superiores e subordinados pertencentes à hierarquia, tanto para a atribuição como para a execução de tarefas necessárias para resolução de um problema [SILVA 2000].

4. O Ambiente de testes

Para a análise e testes dos protótipos, objeto deste estudo, foi criado um ambiente isolado da rede institucional, com quatro segmentos de rede contendo faixas de IPs inválidas, a fim de se evitar perturbar a rede pública da instituição. A representação do ambiente de testes pode ser vista na Figura 1.

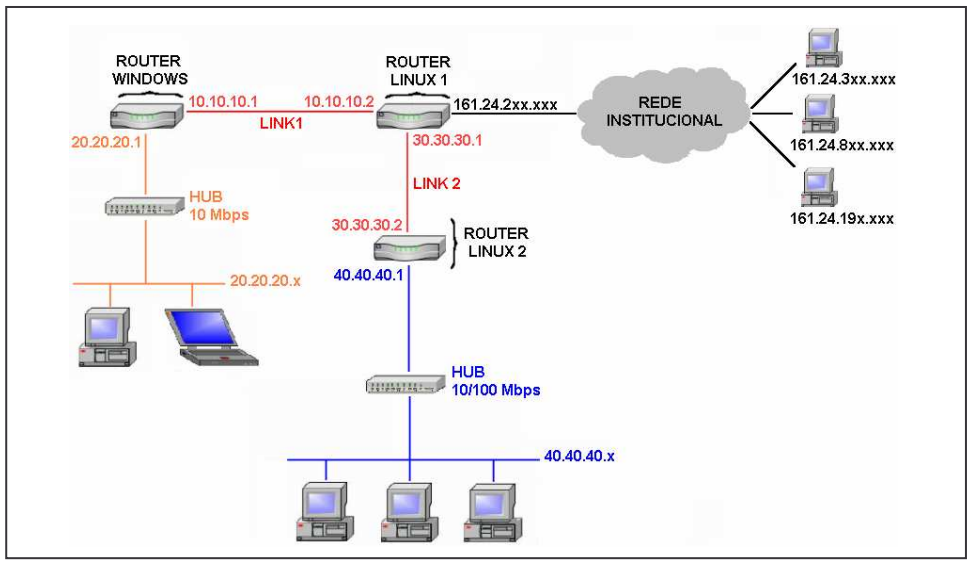

Figura 1. Representação do Ambiente de Testes

4.1 Estrutura Físico-Lógica do Ambiente de Testes

O ambiente padrão *Ethernet* foi configurado com equipamentos do tipo PC como roteadores. O primeiro "Router *Multihomed* Linux 1" utilizando o sistema operacional Linux, sendo responsável pelo repasse de pacotes entre as sub-redes confinadas e a rede pública da instituição. Sua configuração obedeceu a seguinte ordem: a interface `eth0` foi apontada para (sub-rede 20), a qual sai para rede institucional e confina os segmentos de testes; a interface `eth1` foi conectada via cabo *crossover* à interface de um segundo roteador "Router *Multihomed* Windows" utilizando sistema operacional MS-Windows.

O roteador "Router *Multihomed* Linux 2" utilizou também um sistema operacional Linux, foi conectado via cabo *crossover* tendo sua interface `eth0` apontada para a (sub-rede 30) e uma outra `eth1` apontada para (sub-rede 40).

No "Router *Multihomed* Windows" foram configurados um servidor DNS e um servidor de e-mail. O mesmo ainda comportou o agente mestre da sociedade idealizada. Suas interfaces `LAC` e `LAC2` foram apontadas respectivamente para a (sub-rede 10) e (sub-rede 20). Todos os demais *hosts* receberam agentes do tipo escravo.

4.2 Organização dos AEHs no Ambiente de Testes

A organização da sociedade hierárquica constituída pode ser vista na Figura 2. Nela, é possível notar que no nível zero está situado o AEH raiz (agente mestre) e que os níveis 1 e 2 comportam os AEHs intermediários ou folhas (agentes escravos), estando constituídos no nível 3 apenas AEHs folhas (agentes escravos).

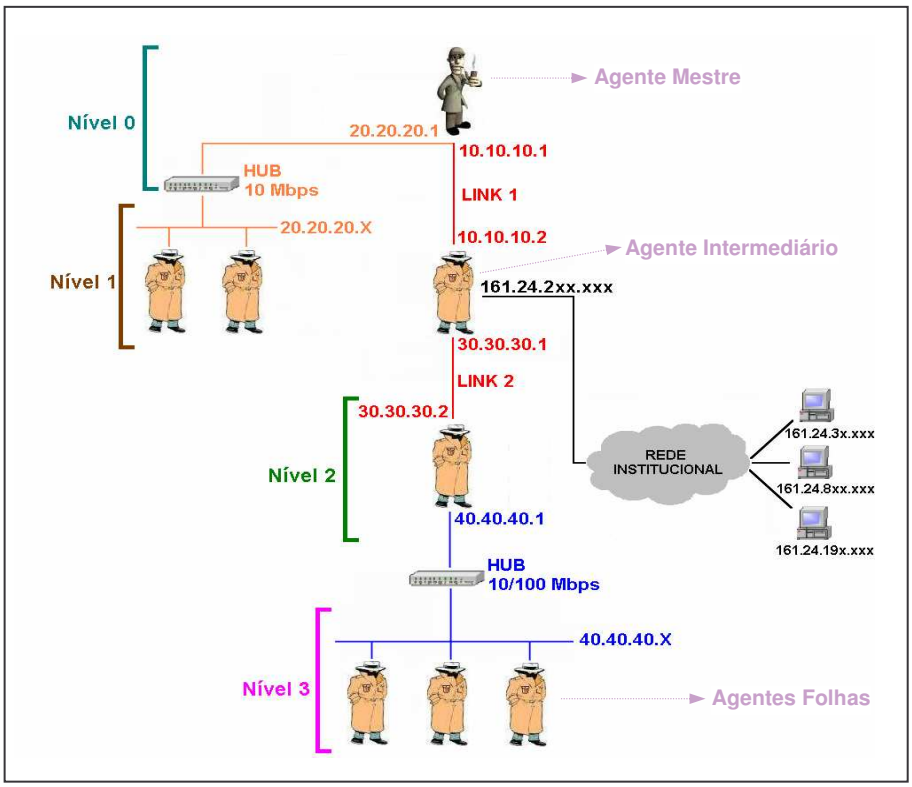

Figura 2. Representação da Distribuição dos AEH no Ambiente de Testes

4.2.1 Funcionamento dos AEHs no Ambiente de Testes

A busca para solução de um problema percebido por algum dos agentes integrantes da sociedade é realizada da seguinte forma: quando um agente é acionado, ele pesquisa em sua base de conhecimentos acionando o motor de inferência do JESS localmente, buscando no seu conjunto de regras os fatos necessários para a solução do problema alvo. Se esse agente for do tipo escravo e, caso a solução não seja encontrada em sua base de conhecimentos, o mesmo solicita auxílio ao agente imediatamente superior a si, podendo esse ser ainda um agente escravo-intermediário ou não. Esse por sua vez, pesquisa em sua base de conhecimentos a qual aciona também localmente o motor de inferência do JESS.

Se uma solução for encontrada, esse agente envia a resposta ao agente escravo, que lhe fez a solicitação. Caso a solução não faça parte da base de conhecimentos desse agente, o mesmo irá solicitar auxílio ao agente mestre da sociedade, que irá pesquisar em sua base de conhecimentos para buscar uma solução ou, ainda, indicará se algum dos agentes da sociedade possui tal conhecimento. Se houver resposta positiva de algum deles, esta será repassada ao agente que a solicitou, para que o mesmo possa solucionar o problema. Caso contrário, será enviado um e-mail ou uma mensagem on-line ao gerente da rede, informando-o que um determinado problema ocorreu e que o mesmo não foi solucionado pelos agentes da sociedade.

Vale ressaltar que tanto para os casos em que houver ou não sucesso na solução de quaisquer problemas, será criado um arquivo de *log* que registrará todos os eventos ocorridos na tentativa da solução, podendo este arquivo ser consultado a qualquer momento pelo gerente da rede. Cabe ainda esclarecer que a forma de raciocínio

escolhida para percorrer as bases de conhecimento de todos os agentes foi o encadeamento para frente "*default*" no JESS, o qual se mostrou mais adequado para a monitoração e solução dos problemas estudados.

No caso de uma anomalia ser reconhecida por um agente mestre, esse acionará sua base de conhecimentos local. Se a solução requerida estiver presente nessa base, as regras correspondentes são então, executadas acionando os devidos *scripts* de verificação ou correção e arquivos de histórico são gerados. Caso contrário, uma mensagem é enviada no sentido *top-down* para os agentes imediatamente inferiores. Estes pesquisam em suas bases de conhecimento a fim de encontrar regras que casem com o problema alvo solicitado pelo agente mestre. O procedimento é executado até que a solução requerida seja encontrada. Esta é então enviada ao agente mestre, que é o detentor da solicitação, e o processo é finalizado.

Quando não existir em nenhum dos agentes da sociedade, regras que contenham uma solução para o evento ocorrido, é enviada uma mensagem ao agente mestre informado que não houve sucesso. Em ambos os casos, é gerado um arquivo de *log* dos eventos ocorridos, além de ser enviado ao gerente da rede um e-mail contendo toda a traçabilidade dos eventos ocorridos.

Já para situações onde agentes escravos reconhecem uma anomalia, a base de conhecimentos local do agente é acionada pelo motor de inferência do JESS. Se a solução requerida estiver presente na base local, as regras correspondentes são então executadas e os devidos arquivos de histórico são gerados. Caso contrário, uma mensagem poderá ser enviada nos sentidos *top-down* ou *button-up*, ou ainda para agentes adjacentes "colegas" ao detentor do processo ativo. O procedimento é executado até que a solução requerida seja encontrada. Ao ser encontrada uma solução esta é enviada ao agente escravo solicitante, o qual é o dono do processo e então o procedimento é finalizado.

Igualmente ao que ocorre com solicitações realizadas por agentes mestres, se não existirem em nenhum dos agentes da sociedade regras que contenham uma solução para o evento ocorrido, é enviada uma mensagem ao agente escravo solicitante de que não houve sucesso.

Para se construir as *baselines*, foi desenvolvido em linguagem Java um protótipo denominado COHPA (Coletor Hierárquico Pró-Ativo) com o qual é possível setar-se os IPs dos *hosts* a serem investigados, o período de captura dos parâmetros fornecidos pelas MIBs e ainda os serviços a serem monitorados. Obviamente, todos os agentes estão preparados para responder a estímulos que indiquem degradações ou falhas de um ou mais serviços, ou seja, nas bases de conhecimento de todos os agentes da sociedade existem critérios de monitoração e correção desses serviços.

5 A Geração de Tráfego

Foram desenvolvidos vários *scripts* de carga em CShell para a criação de tráfegos especializados para os protocolos: Telnet, FTP, POP3, SMTP, DNS e Ethernet. Isso permitiu não só a validação do protótipo, como também das bases de conhecimento, pois *baselines* artificiais bem controladas puderam ser investigadas de modo sistemático pelos agentes.

Definiu-se para cada protocolo em monitoração, valores limítrofes para o acionamento das respectivas bases de conhecimento. Esses valores foram definidos por meio da realização de experimentos, nos quais chegou-se aos extremos para que as anamolias surgissem. Na Figura 3 são ilustrados os momentos de acionamento da base de conhecimentos do protocolo Telnet, durante um período de análise de uma hora.

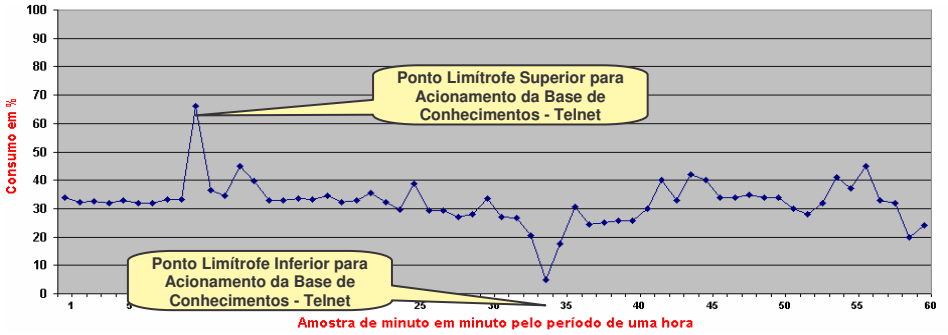

Figura 3. Plotagem de Dados Coletados Sobre o Protocolo Telnet

6 Considerações Finais

Este estudo procurou demonstrar como a utilização de técnicas de IA, IAD e regras de produção mapeadas em JESS associadas às informações contidas em MIBs, contribuem fortemente para a prevenção e solução de problemas de forma pró-ativa em redes de computadores baseadas no padrão *Ethernet*.

Apesar de não citado anteriormente, os protótipos desenvolvidos não se limitaram a monitorar apenas parâmetros pertencentes as MIBs RMON2 e SNMPv2, tendo se estendido também à monitoração de parâmetros ligados aos sistemas operacionais Linux e MS-Windows, bem como a coletas de informações das MIBs system e interface.

Para todos os protocolos monitorados foram obtidos resultados satisfatórios, para os problemas estudados. Basicamente isto ocorreu por três motivos: (i) o algoritmo para o cálculo da taxa de utilização de um protocolo utilizado pelo COHPA é extremamente sensível, o que acarreta no disparo das regras pelo motor de inferência praticamente em tempo real quando um limiar é atingido; (ii) a velocidade oferecida pelo JESS na busca e encontro de uma solução é ótima, mesmo quando executado em equipamentos com poder de processamento relativamente baixo; (iii) a comunicação via RMI (*Remote Method Invocation*) implementada pela linguagem Java e utilizada pela ferramenta MOSES para a comunicação entre os agentes da sociedade, além de ter um impacto muito pequeno, algo em torno de 0,03% sobre canal de comunicação, não representa empecilho nenhum ao tempo de resposta na comunicação entre os agentes.

Um ponto forte a ser destacado, é a realização do gerenciamento pró-ativo em equipamentos contendo sistemas operacionais diferentes e em diferentes versões, no caso o Windows e o Linux. Outro ponto positivo se dá pelo fato de que problemas locais podem ser resolvidos por agentes presentes em equipamentos remotos, desde que constituintes da sociedade e alcançáveis por endereçamento IP.

Há de se levar em conta, ainda, a portabilidade oferecida pela linguagem Java, na qual foram desenvolvidos o MOSES, o COHPA e também o Shell JESS. Assim, podem ser acrescentados não só novos módulos aos aplicativos, bem como, enriquecer as bases de conhecimento com novas regras, propiciando o desenvolvimento de uma solução relativamente barata e facilmente extensível.

Este trabalho não pretende por fim à discussão sobre gerência pró-ativa de redes. Por esse motivo os autores sugerem algumas direções para trabalhos futuros, entre outros: (i) estabelecer o tratamento de limiares de forma automática, tendo por base que as características de funcionamento normal da maioria das redes mudam com certa freqüência. Isso traria ao gerente de redes não só um ganho significativo de tempo de análise, como também poderia evitar possíveis equívocos por parte do mesmo; (ii) as soluções para o gerenciamento de redes baseadas em RMON e RMON2 ainda são bastante caras, assim uma boa alternativa seria o desenvolvimento de probes open-source, o que certamente reduziria os custos de implantação deste modelo de gerenciamento; (iii) se aplicado a uma "grande" estrutura de rede, essa poderia ser constituída de *hosts* com poder de processamento considerável, habitados por agentes do tipo mestre. Esses poderiam então, se comunicar com agentes "*experts*" propiciando que grandes bases de conhecimento fossem velozmente consultadas.

Referências

COMER, Douglas E. (2001) Redes de Computadores e Internet - 2ª ed.. Bookman, Porto Alegre.

BRISA. (1993). Gerenciamento de Redes: Uma Abordagem de Sistemas Abertos. Makron Books/Telebrás, São Paulo.

INFONETICS Research, Inc. (2000). The Cost of Network Downtime in the US 2000. September.

NETO, Francisco Watzko. (1997). Gerenciamento Pró-Ativo: Consideração do Tráfego de Aplicações e Utilização de Séries Temporais. Dissertação de Mestrado. Campinas, SP, UNICAMP.

NOTARE, Mirela Sechi Moretti Annoni, *et al*. (1999). Gerência Proativa de Redes com Técnicas de Inteligência Artificial. CLEI99 - XXV Conferencia Latinoamericana de Informática. Asuncion, Paraguay.

WALDBUSSER, S. (1993). Token Ring Extensions to the Remote Network Monitoring MIB. Request for Comments 1513, September.

WALDBUSSER, S. (1997). Remote Network Monitoring Management Information Base Version 2 using SMIv2. Request for Comments 2021, January.

IDDON, R. (1997). Remote Network Monitoring MIB Protocol Identifiers. Request for Comments 2074, January.

CRUZ, Fernando Augusto da Silva, *et al*. (1997). Uso de Inteligência Artificial na Implementação de um Sistema de Gerência Proativo para Redes ATM. In: XXIII Conferência Latino-Americana de Informática XXIII CLEI'97. Valparaiso-Chile.

WEIBEL, Peter. (2006). MECAD - Media Centre d'Art i Disseny. La imagen inteligente. disponível em 28/07/2006 em http://www.pucsp.br/~gb/texts/

MT. (2006). The MacTutor History of Mathematics archive. disponível em 30/10/2006 em http://www-history.mcs.st-andrews.ac.uk/history/Mathematicians/Thue.html

BITTENCOURT, Guilherme. (1998). Inteligência Artificial: Ferramentas e Teorias. Editora da UFSC, Florianópolis.

HILL, Ernest Friedman. (2003). Jess in Action: Java Rule-based Systems. Manning Publications Co.

SILVA, Vera Lúcia da. (2000). Um Modelo Hierárquico para Sistemas Especialistas. Dissertação de Mestrado. São José dos Campos, SP, ITA.