Uma Abordagem para Gerenciamento Distribuído de Redes Utilizando Agentes Móveis

Christian Brackmann, André Fiorin, Cristiano Berni, Giordana Osmari, Gustavo Oliveira, Ramon Limberger, Rogério Turchetti

Instituto de Informática - Centro Universitário Franciscano (UNIFRA)
Rua dos Andradas, 1614 - 97010-032 - Santa Maria - RS – Brasil

{brackmann, razlen, cberni, giordana}@gmail.com,
menezesoliveira@brturbo.com, {ramon.limberger, turchetti}@gmail.com

RESUMO*: O agente SNMP é um padrão de gerenciamento de rede amplamente usado em redes TCP/IP. Tal agente é convencionalmente utilizado para gerenciamento de dispositivos de uma maneira centralizada. Através do gerenciamento centralizado o gerente pode tornar-se um gargalo e com excessiva carga de processamento, pois toda e qualquer informação deve ser retornada até o gerente. Uma alternativa é distribuir tal gerenciamento com a finalidade de obter maior escalabilidade. Neste trabalho é proposta uma abordagem de gerenciamento distribuída através de agentes móveis. O artigo apresenta o custo pago para distribuir o gerenciamento; e como resultado comprova-se que agentes móveis permitem reduzir a carga dos dispositivos gerentes.*

1. Introdução

Em virtude da relação custo-benefício, a utilização de redes de computadores vem crescendo de forma bastante significativa, atuando simultaneamente em conjunto das tecnologias de telecomunicações e informática. Entretanto, devido ao grande aumento de novos dispositivos que se comunicam através da rede, o monitoramento de redes de computadores está se tornando uma tarefa cada vez mais complexa, afetando o gerenciamento de redes em diversos aspectos como, por exemplo, a escalabilidade, sobrecarga dos dispositivos e canal de comunicação [STALLINGS - 1999] [KONA e XU].

O gerenciamento de redes essencialmente envolve o monitoramento e a coleta de informações para possível análise dos dispositivos monitorados. Convencionalmente tal gerenciamento é realizado através de agentes SNMP (*Simple Network Management Protocol*) [STALLINGS - 1999]. Entretanto, em ambientes de larga escala, a flexibilidade de gerenciamento pode ser afetada se for utilizada uma abordagem de gerenciamento centralizada [KONA e XU][CLEMENTS - 1997]. O problema ocorre pelo fato de que o gerente pode se tornar um gargalo, com uma excessiva carga de processamento, impedindo o monitoramento adequado no referido ambiente.

Uma alternativa é distribuir o gerenciamento entre diversos gerentes sobrepondo a estratégia centralizada. Neste sentido, o presente trabalho tem por objetivo desenvolver uma ferramenta distribuída para realizar o gerenciamento de dispositivos com o uso de agentes móveis (Java *Aglets*), a partir do protocolo de gerenciamento SNMP e analisar a latência de comunicação, bem como, consumo de recursos causados pelos agentes móveis, quando comparados ao uso convencional de ferramentas de gerenciamento

SNMP. Ressalta-se que o objetivo é manter em determinados pontos da rede (comunicação na LAN) o monitoramento convencional, entretanto integrado com agentes móveis possibilitando assim distribuir o gerenciamento (comunicação na WAN).

O artigo encontra-se estruturado da seguinte forma: na seção 2 serão destacados os agentes (SNMP e *Aglets*) responsáveis por executar o gerenciamento distribuído. Na seção 3 será apresentada a integração dos agentes SNMP e *Aglets*. A seção 4 apresenta os experimentos e resultados obtidos. Por fim, na seção 5 apresenta conclusões do artigo.

2. Agentes que Fazem Parte da Arquitetura para Gerenciamento Distribuído

Nesta seção serão apresentadas características conceituais sobre os agentes que fazem parte da ferramenta de gerenciamento distribuída, tais como: agentes SNMP utilizados para gerenciamento de redes; e agentes móveis, utilizados para estender/distribuir a ferramenta através de *aglets*.

2.1. Agente SNMP

O SNMP é um protocolo de gerenciamento típico de redes TCP/IP [4], da camada de aplicação que facilita o intercâmbio de informação entre os dispositivos de rede. O agente SNMP consulta as informações armazenadas por uma base de dados denominada MIB (*Management Information Base*), que procura abranger todas as informações necessárias para a gerência da rede. O SNMP também possibilita aos administradores de rede gerir o seu desempenho, como também encontrar e resolver problemas, e planejar o crescimento desta [RFC 1156].

Os comandos para as consultas são simples e baseados no mecanismo de busca e alteração. Neste mecanismo estão disponíveis as operações de manipulação de um objeto, de obtenção dos seus valores e suas variações. Sua utilização com um número limitado de operações, torna o protocolo de fácil implementação, simples, estável e flexível. O funcionamento do SNMP é baseado em dois dispositivos denominados agentes e gerentes. No contexto deste trabalho um gerente refere-se ao *host* que recebe todas as informações a respeito dos *hosts* gerenciados e um agente é o *host* que permite estender o gerente ou também ser gerenciada. Em outras palavras um *host* agente permite distribuir a aplicação de gerenciamento. Cada *host* gerenciado deve possuir um agente SNMP e uma base de informações MIB [RFC 3413].

Os agentes SNMP fornecem respostas, somente às solicitações do sistema de gerenciamento, com exceção de uma situação anormal, onde uma mensagem TRAP é enviada. Dentre as operações disponíveis (Figura 1), os seguintes comandos podem ser utilizados:

- *get*: um valor de um contador específico é solicitado ao agente SNMP;
- get-next: o valor do próximo contador é solicitado ao agente SNMP;
- *walk*: solicita os valores de todos os contadores de um objeto SNMP;
- *trap*: envia ao sistema de gerenciamento informações sobre situações anormais interceptadas pelo agente SNMP;
- *set*: instrui o agente para alterar um parâmetro configurável.

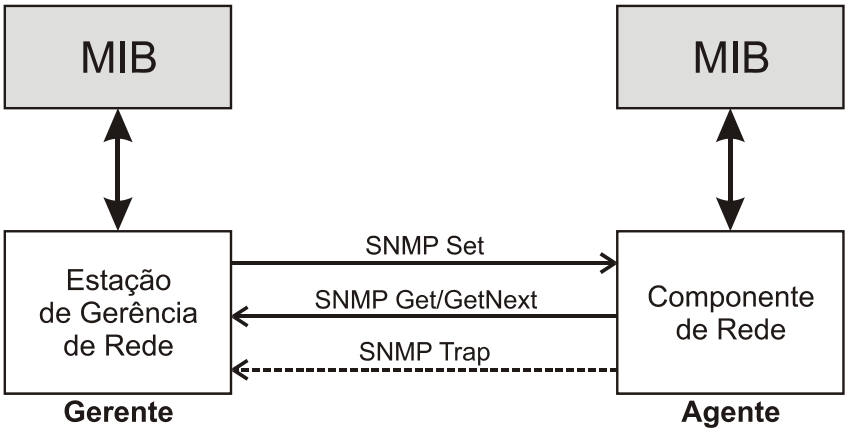

Figura 1. Comunidade e Funções SNMP

Ressalta-se que a instalação (opcional) de um agente SNMP é extremamente simples e são suportados pela maioria dos sistemas operacionais.

2.2. Agentes Móveis

Um AM (agente móvel) é uma entidade de *software* responsável pela execução de uma atividade, com a capacidade de migrar autonomamente através de uma rede em tempo real. Um agente migra em um ambiente distribuído de um *host* a outro. Quando um agente migra, sua execução é suspensa na agência original, o agente é transportado para uma outra agência no ambiente distribuído, onde a execução é retomada [SILVA].

O conceito de agentes móveis vem sendo proposto para suportar diferentes tipos de aplicações, dentre elas: comércio eletrônico, gerenciamento de *workflows*, gerenciamento de redes, serviços de telecomunicação, recuperação de informações distribuídas e redes ativas. Semelhante aos agentes estacionários[1], os AMs têm o diferencial de possuir a habilidade de aprender, se adaptar, argumentar e se comunicar com usuários e outros componentes de *software* apoiados em um ambiente de rede em tempo real, mesmo quando submetida a modificações. Dentre as diversas vantagens oferecidas pelos AMs sobre um ponto de vista cliente/servidor, estes também podem reduzir o tráfego de rede sensivelmente empacotando a conversação inteira em uma única transmissão de agente, e, ao lidar com grandes volumes de dados, os AMs podem aumentar a eficiência executando a maioria dos cálculos e seleção de informações em *host* onde os dados são armazenados [CAM - 2003].

2.2.1. Aglets

É um *middleware* que dá suporte ao desenvolvimento de sistemas baseados em AM sob a plataforma Java. Esta tecnologia foi originalmente desenvolvida pelo Laboratório de Pesquisa da IBM (*International Business Machines*) de Tóquio [CLEMENTS - 1997], e posteriormente o código fonte da plataforma foi aberto e está disponibilizado como projeto *open source*.

[1] Agentes estacionários limitam o seu ambiente de atuação no *host* que está contido

É importante salientar que o uso de *aglets* se aplica exclusivamente a empresas, pontos ou terminais geograficamente distantes. Também pode ser usado em aplicações que monitoram redes de grandes portes.

No caso de uma grande empresa com várias filiais distribuídas pelo país ou pelo mundo, cada uma contendo cerca de cem computadores, é mais viável o uso de agentes móveis, pelo fato de que mantém a alta escalabilidade da rede, permitindo ao gerente de uma LAN a execução de aplicativos em um ambiente externo.

Por ser completamente feito em Java, a alta portabilidade dos agentes e da plataforma é garantida. Sua execução é feita em um servidor de agentes chamado *Aglet Server*, que fornece um aplicativo gráfico (denominado *Tahiti*) que possibilita a criação, transferência e eliminação dos agentes, além da configuração de privilégios de acesso ao servidor. Quando um *aglet* é criado, este atua em um contexto, que é o ambiente de execução do agente. Este contexto permite a ele o acesso aos recursos do servidor, garantindo a segurança contra ações maliciosas de outros agentes.

Múltiplos *aglets* podem se comunicar entre si através de troca de mensagem, visto que uma mensagem é um objeto trocado entre eles. Esta comunicação pode ser feita de modo assíncrono ou síncrono. Os métodos básicos do ciclo de vida que um *aglet* pode passar são:

- Criação (*onCreate()*): invocado sobre o agente logo após sua criação, recebendo uma identificação.
- Clonagem (*onClone()*): invocado para clonar um *aglet* existente, porém o novo *aglet* recebe uma nova ID e recomeça a execução.
- Liberação (*onDisposing()*): invocado sobre o agente antes de sua destruição, pára sua execução e remove-o de seu contexto corrente.
- Despache (*onDispatching()*): invocado sobre o agente para enviá-lo a outro contexto.
- Retração (*onRetraction()*): invocado para puxar um *aglet* de um contexto para outro.
- Desativação (*onDeactivating()*): invocado sobre o agente para encerrar a execução e armazenar seu estado em uma memória secundária.
- Ativação (*onActivating()*):ativa um *aglet* restaurando-o no mesmo contexto.

No contexto deste trabalho os métodos *onCreate()*, *onDispatching()* e *onDisposing()* foram utilizados para criar um agente estendendo a aplicação gerente; despachá-lo ao *host* alvo para coleta de informações e; destruí-lo após cumprir seu itinerário.

3. Integração SNMP/Aglets

A proposta de integração baseia-se na abordagem apresentada em [CARDOSO - 2002], onde se cria uma interface SNMP nos AMs a partir de diferentes objetos Java, habilitando-os à execução de comandos do SNMP. A interface SNMP-Java consiste de uma biblioteca de classes modeladas a partir da estrutura de dados ASN.1 usada nos padrões originais do SNMP [ASSUNÇÃO].

Desta forma, não há necessidade de implementação de novas estruturas de dados, deixando os mecanismos de comunicação para o sistema de agentes.

Para interação com agentes SNMP, é necessário incluir capacidades SNMP aos AMs. Desta forma, eles se tornam habilitados a realizarem comandos GET e SET. Isso é possível através da importação, pelo AM *Aglets*, de classes Java.

Na próxima seção serão descritos os resultados e ambientes no qual foram executados os testes.

4. Ambiente de Teste e Resultados

Os experimentos foram realizados em computadores que avaliam o custo entre consultar informações diretamente aos agentes SNMP ou enviar um agente móvel até o outro dispositivo e colher informações para possível análise.

Nos testes foram utilizados dez tipos diferentes de MIB's para a realização das consultas. Para executar as consultas de uma forma eficiente, foram criados primeiramente dez objetos, e depois um vetor para armazenar as consultas MIB a serem feitas. Em seguida, o vetor é enviado no objeto e os MIB's executados localmente e retornados.

Os computadores estavam munidos pelos Sistemas Operacionais Windows XP e Linux Fedora Core, Java JDK 1.5 e com ASDK (*Aglet Software Development Kit*).

Para chegar aos resultados obtidos usando SNMP sem nova conexão, foi guardado o *timestamp* local, criada a conexão remota, feita a consulta remota, comparou-se o *timestamp* atual com o local. No final mostra o resultado na tela local.

Usando SNMP com nova conexão, foi guardado o *timestamp* local, criada a conexão remota, feita a consulta remota, e por fim mostra o resultado final na tela local.

Usando *Aglets* sem uma nova conexão, foi necessário criar um objeto local, guardar o *timestamp* local, e enviar o objeto para o computador remoto. Logo, faz-se a consulta local, o objeto se auto-reenvia ao criador, compara o *timestamp* atual com o já armazenado e mostra o resultado na tela.

Finalmente, usando *Aglets*, com uma nova conexão, o objeto é criado, o *timestamp* local é guardado, o objeto é enviado para a máquina remota, é feita a consulta. Logo após o objeto se auto-reenvia ao seu criador, os *timestamp* são comparados e o resultado final é mostrado na tela local.

A tabela 1 apresenta os resultados (dados médios) referentes ao tempo gasto para obter as informações na MIB e retorná-las ao computador gerente. Este experimento revela que o uso de *aglets* leva mais tempo para realizar as consultas elevando o tempo de resposta. Entretanto, este resultado já era esperado pelo fato de que em *aglets* um agente deve ser criado e executado em outro computador. Mas por outro lado este custo é recompensado pela redução da sobrecarga no computador gerente como pode ser visualizado na Tabela 2.

Tabela 1. Tempo de resposta

Serviço\Nº de Objetos	10	20
SNMP	48,5 ms	53,1 ms
Aglets	94 ms	106,3 ms

Durante os testes também foram verificados o uso do processador e da memória de cada computador. Observou-se que o uso da memória não foi afetado em nenhum computador durante a execução dos testes. O uso do processador varia de acordo com o tipo de serviço usado e da função de cada computador, conforme descrito na Tabela 2.

Tabela 2. Comparação do uso de processador

Uso do Processador	SNMP	Aglets
Computador que chama a consulta (gerente)	69%	7%
Computador que retorna o dado (agente)	32%	100%

Os resultados da Tabela 2 revelam que o maior consumo refere-se à execução de um agente em *aglet* no computador que irá retornar a consulta realizada pelo gerente, obtendo acréscimo de 68% se comparado ao gerenciamento sem o uso de *aglets*. Em contrapartida observa-se que o gerente reduz a sobrecarga de 69% para 7%, equivalente a uma redução de aproximadamente 89%.

Em síntese, os experimentos permitem concluir que o uso de *aglets* reduz o gargalo no *host* gerente gerado pelas ferramentas convencionais de gerenciamento centralizado.

5. Conclusão

Neste artigo foi proposto um gerenciamento distribuído de redes, com a finalidade de reduzir os encargos postos aos *hosts* gerentes pelas abordagens de gerenciamento centralizadas. Os experimentos revelaram que o uso de *aglets* para este fim, aumenta o tempo de resposta para as consultas aos *hosts* gerenciados e ainda consomem um elevado custo de processamento onde elas são executadas. Em contrapartida elas apresentaram ser uma ótima alternativa para distribuir o gerenciamento reduzindo a sobrecarga dos *hosts* gerentes. Esta redução foi de aproximadamente 89% no custo de processamento.

Com o uso de agentes móveis baseados em Java foi possível perceber que o ambiente de desenvolvimento de aplicações distribuídas torna-se mais adequado, pois a linguagem Java fornece total exploração dos conceitos de orientação a objetos e mantém perfeita escalabilidade de uma rede LAN/WAN, permitindo ao gerente de uma LAN executar aplicativos em um ambiente externo.

6. Referências

JALOTE, V. Falt-Tolerance in Distributed Systems. New Jersey: Prentice Hall, 1994.

ASSUNCAO, Marcos; WESTPHAL, Carlos Becker; KOCH, Fernando Luiz. Arquitetura de Grids de Agentes Aplicado ao Gerenciamento de Redes de Computadores e de Telecomunicação. Rio de Janeiro, 2003. In: Simpósio Brasileiro de Redes de Computadores, Proceedings, Pag. 789-804.

STALLINGS, WILLIAM. (1999) “SNMP, SNMPv2, SNMPv3 and, RMON1 and 2”, Addison-Wesley, Third Edition, September.

RFC 3411 (Standard 62) — An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks

RFC 1156 — Management Information Base for Network Management of TCP/IP-based internets.

RFC 3413 (Standard 62) — Simple Network Management Protocol (SNMP) Application.

ASSIS SILVA, Flávio M.. Universidade Federal da Bahia Laboratório de Sistemas Distribuídos LaSiD Prédio do CPD, Av. Adhemar de Barros, S/N, 40170110. Salvador/BA.

TOLMAN CAM, (2003) “A Fault-Tolerant Home-Based Naming Service for Mobile Agents”, Rensselaer Polytechnic Institute, Troy, New York, April.

KONA, Manoj K. and XU, Cheng-Zhong A Framework for Network Management using Mobile Agents - Department of Electrical and Computer Engineering Wayne State University, Detroit, MI 48202.

VERMA, DINESH C. (2000) “Policy-Based Networking: Architecture and Algorithms”, New Riders, First Edition: November.

CARDOSO, A. (2002) “Integrando Agentes Móveis com Sistemas Legados para Gerenciamento de Redes ATM Heterogêneas”, Dissertação Mestrado, Universidade Federal de Campina Grande. Paraíba, Agosto.

ASSUNÇÃO, MARCOS. Project HOPE 1.0 - Hot Topics em Gerência de Redes.

MELO, Inácio D. Júnior e CELESTINO JÚNIOR, Joaquim - Descoberta e Monitoramento de Recursos em Redes de Computadores usando Agentes Móveis- Departamento de Estatística e Computação – Universidade Estadual do Ceará (UECE).

CLEMENTS, P. E., PAPAIOANNOU, Todd and EDWARDS, John. *Aglets*: Enabling the Virtual Enterprise. Stakeholders, Engineering, Logistics and Achievement, Loughborough University, UK, 1997.