

Utilização de agentes móveis e a plataforma JavaEE para gerência de redes de computadores

Allysson Steve Mota Lacerda

Programa de Pós-Graduação em Engenharia Elétrica
Universidade Federal de Minas Gerais (UFMG)
Belo Horizonte – MG – Brazil

stevelacerda@cpdee.ufmg.br

1 Gerenciamento de redes

O termo gerenciamento de redes significa coisas diferentes para pessoas diferentes. Em alguns casos, envolve um monitoramento solitário da rede, utilizando alguma aplicação desatualizada. Em outros casos, envolve uma base de dados distribuída, coleta automática de informações e uma estação de trabalho gerando gráficos em tempo real do tráfego e das modificações na topologia da rede. Em geral, o gerenciamento de redes é um serviço que abrange uma variedade de ferramentas, aplicações e dispositivos para auxiliar os administradores de rede na monitoração e manutenção das redes.

1.1 Histórico

O início da década de 1980 apresentou uma imensa expansão das redes de comunicação. Assim que as empresas passaram a perceber os ganhos de produtividade e custo, começaram a instalar novas redes ou expandir as existentes, à medida que novas tecnologias e produtos foram desenvolvidos. Já na metade da década de 1980, algumas empresas presenciaram vários problemas devidos à utilização de tecnologias de rede diferentes e, em alguns casos, incompatíveis.

No final da década de 1980, os recursos necessários para gerenciar as grandes e heterogêneas redes resultaram em uma crise para diversas organizações. Surgiu então a necessidade urgente de um gerenciamento automatizado (incluindo o que é normalmente conhecido como planejamento de capacidade) que pudesse ser integrado a diversos ambientes.

2 Protocolos de gerenciamento

Vários padrões para o gerenciamento de redes foram propostos nos últimos anos. Entretanto, a grande maioria não foi aceita pelo mercado, por motivos que vão desde a limitação de funcionalidades à complexidade de implantação. A seguir serão apresentados alguns padrões e suas principais características.

2.1 SNMP

Desde sua criação, o Simple Network Management Protocol (SNMP) é o modelo de gerenciamento de redes TCP/IP mais popular. O IETF (Internet Engineering Task Force) criou o SNMP para permitir o gerenciamento remoto de dispositivos de redes utilizando um conjunto padronizado de operações. Atualmente, o SNMP é amplamente implementado por servidores, impressoras, switches, modems, roteadores etc.

2.1.1 Arquitetura básica

O SNMP é composto por dois tipos principais de entidades:

- **Gerente** – servidor que executa o software de gerenciamento de uma determinada rede. Esses servidores são normalmente conhecidos como Network Management Stations (NMS).
- **Agente** – software que reside em um dispositivo remoto a ser gerenciado. Atualmente, quase todos os dispositivos baseados em IP provêm algum tipo de agente SNMP embutido. O agente possui duas funções principais: aguardar requisições SNMP vindas do NMS e monitorar eventos internos para alertar o NMS quando ocorrerem mudanças importantes.

O NMS é configurado para coletar informações de todos os principais dispositivos da rede periodicamente. Pelo fato de poder existir centenas ou milhares de dispositivos gerenciados, não é comum a coleta de informações em um determinado dispositivo mais que algumas vezes por hora. Entretanto, um grande intervalo entre as coletas de informações pode fazer com que um problema não seja detectado: se uma máquina reiniciar, por exemplo, entre uma coleta e outra, o NMS jamais saberá que isso aconteceu.

Para resolver esse problema, um agente SNMP pode enviar informação para o NMS utilizando traps SNMP. Um trap é uma notificação de evento não-solicitada, que normalmente indica a ocorrência de algum problema. Por não ser orientado a conexão e normalmente indicar problemas na rede, é comum que um trap não chegue ao NMS e, com isso, informações importantes podem ser perdidas.

Devido a essa falha, SNMPv2c e SNMPv3 incluem um outro tipo de pacote, denominado inform SNMP. Ele é praticamente idêntico ao trap padrão, exceto pelo fato de o agente aguardar uma confirmação de recebimento do NMS. Se essa confirmação não for recebida em um determinado período, o agente transmite novamente o inform.

3 Agentes Móveis

Tradicionalmente, aplicações distribuídas são desenvolvidas utilizando a arquitetura cliente-servidor, na qual os processos se comunicam através de troca de mensagens ou chamadas de procedimento remoto (RPC). Esse modelo de comunicação é normalmente síncrono, ou seja, o cliente envia uma requisição e fica aguardando uma resposta.

Uma arquitetura alternativa é a Remote Evaluation (REV), na qual o cliente envia o seu próprio procedimento (código binário) para o servidor executar e retornar os resultados. Alguns sistemas mais novos criaram o conceito de mensagens ativas, que podem migrar entre os nós da rede carregando códigos a serem executados. Um conceito mais genérico é o de objetos remotos, que encapsulam os dados juntamente com os métodos para manipulá-los e podem ser transportados de um dispositivo para outro.

O paradigma dos agentes móveis evoluiu dessas arquiteturas. A Figura 1 mostra como ele se difere do RPC e do REV. No RPC, há uma comunicação bidirecional para o envio de dados (mensagens), enquanto que no REV o código é enviado pelo cliente para o servidor e apenas dados (resultados) são retornados. Já um agente móvel é um programa (código, dados e contexto de execução) enviado pelo cliente para o servidor. Diferentemente de uma chamada de procedimento, após a execução, o agente móvel

pode migrar para outros servidores, retornar os resultados ou migrar de volta para o cliente, dependendo do que for mais apropriado.

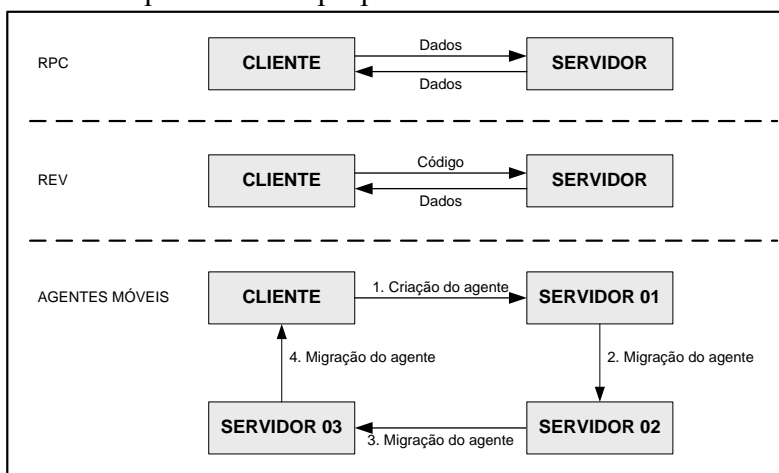


Figura 1 Diferenças entre RPC, REV e agentes móveis

3.1 Benefícios

A utilização dos sistemas baseados em trocas de mensagem ou chamadas de procedimento remoto estão sendo aos poucos substituídos por aqueles baseados em agentes móveis. Essa mudança deve-se, principalmente, às vantagens dos agentes móveis no que diz respeito aos custos de implantação e manutenção dos sistemas. A seguir são apresentadas algumas dessas vantagens e aplicações que delas se beneficiam.

- **Menor utilização da rede** - aplicações de busca e filtragem de informação frequentemente coletam grandes quantidades de informações localizadas em algum servidor na rede, filtram-nas e geram uma quantidade relativamente pequena de informações como resultado. Com um sistema baseado em agentes móveis, as informações são processadas em seu local de origem e apenas os resultados irão trafegar pela rede. Sistemas de mineração de dados são exemplos de aplicações que se beneficiam dessa característica.
- **Assincronismo entre clientes e servidores** - algumas aplicações envolvem várias interações cliente-servidor, que requerem uma conexão estabelecida por um período bastante longo ou várias conexões isoladas. Com a utilização de agentes móveis, o cliente não precisa manter a conexão estabelecida enquanto aguarda a resposta do servidor. Essa característica é bastante vantajosa para dispositivos móveis (celulares, smartphones, handhelds, laptops etc.), que normalmente possuem conexões lentas ou que são desabilitadas para economia de energia.
- **Paralelismo e tolerância a falhas** - agentes móveis também podem ser utilizados como forma de possibilitar o paralelismo nas aplicações, uma vez que podem ser executados concorrentemente em um sistema distribuído. Um cliente pode dividir suas tarefas entre diversos agentes, a fim de prover paralelismo ou tolerância a falhas, nos casos em que a mesma tarefa é dada a vários agentes.
- **Flexibilidade** - Os servidores normalmente possuem um conjunto fixo de serviços providos (interface pública). Entretanto, as necessidades dos clientes

podem mudar ao longo do tempo, exigindo modificações nessas interfaces. A utilização de agentes móveis torna-se vantajosa nessas situações, uma vez que cada cliente pode manter sua própria interface nos servidores e, sempre que for exigida alguma modificação, os agentes podem ser substituídos por novas versões, sem que haja uma parada nos serviços.

3.2 Desvantagens

Como foi visto, a utilização de agentes móveis pode facilitar a implementação de diversos tipos de aplicações distribuídas. No entanto, traz consigo vários problemas, principalmente nas áreas de segurança e robustez.

O uso de agentes móveis requer que cada dispositivo do sistema possibilite a sua execução. Isso pode permitir a entrada de falsos agentes, como vírus ou cavalos de tróia. A menos que alguma medida seja tomada, esses agentes podem consumir recursos (processador, memória, disco etc.), destruir dados importantes ou interromper o funcionamento do dispositivo hospedeiro. Da mesma forma, os agentes também precisam se proteger dos dispositivos visitados. Informações sobre o usuário (por exemplo números de cartões de crédito ou senhas) ou sobre o estado do agente móvel não podem ser acessadas ou modificadas por processos sem autorização. Devido a tais problemas, a segurança é normalmente a maior preocupação durante o desenvolvimento de um sistema baseado em agentes móveis.

Além disso, com o desenvolvimento de sistemas cada vez maiores e mais complexos, os problemas enfrentados com o gerenciamento dos agentes móveis têm se agravado bastante. Aplicações de origem devem ser aptas a realizar intervenções em algum agente e, para isso, é necessário ter o controle sobre sua localização e identificação.

4 Desenvolvimento de sistemas de agentes móveis

4.1 Modelo de referência

Primeiramente será apresentado um modelo genérico de um sistema de agentes móveis, que servirá como base para as próximas seções. A Figura 2 mostra um modelo simples de um sistema e todas as interações envolvidas. Na imagem, CA e CB são duas máquinas conectadas por uma rede. A fim de fazer parte do sistema, cada uma delas precisa executar um processo denominado servidor de agentes. Esses processos são responsáveis pela hospedagem e execução dos agentes e provêm algumas primitivas básicas para os programadores, como migração, comunicação e acesso aos recursos do equipamento. Servidores de agentes se comunicam através de um protocolo servidor-servidor.

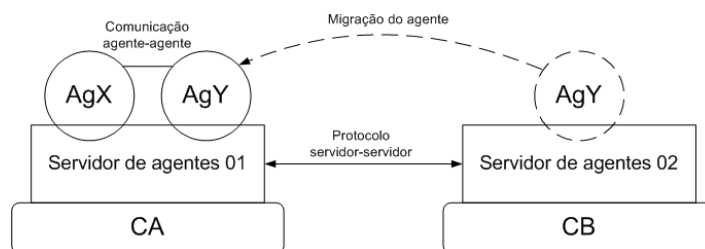


Figura 2 Modelo genérico de um sistema de agentes móveis

Na figura, o servidor de agentes em CA está hospedando um agente, denominado AgX. AgY, um agente hospedado em CB, decide migrar para CA, possivelmente para comunicar-se com AgX ou para acessar algum recurso do dispositivo. Em resposta a essa requisição, o servidor de agentes em CB contacta o servidor em CA e envia o agente. O servidor em CA o recebe e assinala uma thread para executá-lo a partir do ponto em que foi interrompido em CB.

Em geral, um mesmo dispositivo pode ter vários servidores de agentes, cada um hospedando vários agentes móveis simultaneamente. Isso é bastante útil nos casos em que se deseja prover serviços especializados para cada cliente.

5 Java 2 Platform, Enterprise Edition (JavaEE)

O desenvolvimento de software para utilização por corporações empresariais enfrenta inúmeros desafios, dentre os quais podemos destacar:

- **Produtividade** – Agilidade no desenvolvimento e manutenção das aplicações é um fator cada vez mais decisivo. No campo das aplicações distribuídas, no entanto, a diversidade de tecnologias e modelos de programação reduz consideravelmente a produtividade.
- **Escalabilidade** – As aplicações devem ser capazes de se ajustar facilmente ao crescimento, esperado ou não, de demanda.
- **Integração com softwares existentes** – As empresas acumulam durante anos um conjunto de dados e aplicações que devem ser acessados pelos novos sistemas. Uma forma eficiente e organizada de acessar este legado torna-se vital.
- **Liberdade de escolha de fornecedores** – Idealmente, componentes fornecidos por empresas produtoras de software diferentes devem ser capazes de se comunicar de maneira transparente. Assim, a empresa cliente se beneficia de um mercado competitivo de fornecedores de software, conseguindo menores preços e maior qualidade.
- **Segurança** – Em um mundo conectado, tão importante quanto disponibilizar a informação, é impedir o acesso não autorizado a informações e serviços específicos. Assim, mecanismos de segurança são essenciais.

Foi com o intuito de enfrentar estes desafios que a arquitetura JavaEE foi criada. A arquitetura JavaEE é um conjunto de especificações e práticas que possibilitam soluções para o desenvolvimento, instalação e gerenciamento de aplicações multicamadas baseadas em servidores.

5.1 Arquitetura

A plataforma JavaEE define quatro tipos de componentes de aplicação que um produto JavaEE deve suportar:

- **Clientes** – aplicativos Java executados em computadores clientes que geralmente possuem interface gráfica (GUI), embora isso não seja obrigatório. Podem se comunicar tanto os componentes web quanto os de servidor, para acessar os recursos necessários.

6 Integração entre JavaEE e agentes móveis

Sistemas baseados em agentes móveis possuem características que não são comuns à grande maioria dos softwares. O fato de o programa poder migrar de maneira autônoma entre os diversos dispositivos traz uma série de complicações ao seu desenvolvimento. Além disso, o despreparo dos desenvolvedores é outro fator agravante. Como resultado disso tudo, o que se percebe é uma enorme quantidade de produtos extremamente complexos ou com recursos limitados.

Como alternativa, o presente trabalho propõe a utilização de recursos e ferramentas disponíveis na plataforma JavaEE para o desenvolvimento de sistemas de agentes móveis.

6.1 Como realizar a integração

Boas ferramentas de desenvolvimento de sistemas baseados em agentes móveis devem oferecer, no mínimo:

- **Tolerância a falhas** – mecanismos que garantam o correto funcionamento do sistema, mesmo em situações adversas.
- **Escalabilidade** – possibilidade de crescimento do sistema, de acordo com as novas necessidades.
- **Facilidade de implementação** – recursos que tornem transparentes para o usuário questões que não estejam diretamente ligadas às funcionalidades do sistema.
- **Desempenho** – realização de todas as operações de forma satisfatória e em tempo compatível com as necessidades do projeto.
- **Segurança** – garantias de autenticidade, controle de acesso, integridade e confidencialidade dos agentes.

Entretanto, os produtos encontrados no mercado são, geralmente, muito complexos ou possuem limitações quanto aos recursos oferecidos. Outro fator importante é a dependência de fornecedores, já que a maioria das implementações utiliza soluções proprietárias. A seguir serão mostradas algumas alternativas JavaEE para a solução de problemas comuns.

- **Independência de fornecedores** – JavaEE é um conjunto de especificações, não um produto. Atualmente existem diversos fornecedores de soluções JavaEE disponíveis. Além disso, é possível utilizar simultaneamente produtos de diversos fornecedores, sem a necessidade de modificações na aplicação.
- **Independência de plataforma** – JavaEE pode ser executado em qualquer sistema operacional que possua uma Máquina Virtual Java implementada. Permite ainda a utilização de sistemas operacionais distintos, simultaneamente.
- **Escalabilidade** – com suporte nativo à criação de clusters, JavaEE possibilita a adição de novas máquinas ao sistema, de forma transparente para o usuário.
- **Tolerância a falhas** – quando uma máquina entra em estado de falha, todas as requisições a ela feitas podem ser redirecionadas automaticamente para outras, também sem intervenção do usuário.

- **Segurança** – segurança sempre foi um fator de extrema importância em JavaEE, que implementa vários padrões de segurança acordados pela indústria, além de possibilitar a adição de novos. Possui mecanismos de autenticação, controle de acesso, criptografia, entre outros.
- **Desempenho** – JavaEE possui um desempenho inferior, quando comparado a aplicações escritas em C, por exemplo. Isso ocorre em função de JavaEE ser compilado para um estado intermediário (bytecodes), sendo interpretado para a plataforma desejada durante a sua execução. Essa lentidão vem sendo melhorada ao longo do tempo e é justificável pela independência de plataforma alcançada com sua utilização.
- **Identificação e localização** – JavaEE faz uso da Java Naming and Directory Interface (JNDI) para criar uma interface padrão para acesso a vários sistemas de nomeação e diretório. Isso possibilita a troca de tais sistemas sem qualquer mudança para o usuário.
- **Mobilidade** – fator primordial a qualquer sistema de agentes móveis, a mobilidade é tratada em JavaEE através da utilização da serialização de objetos, utilizada pela Remote Method Invocation over the Internet Inter-ORB Protocol (RMI-IIOP).
- **Integração** – através dos conectores e adaptadores de recursos, JavaEE permite a integração com sistemas existentes. É possível, ainda, a utilização de documentos XML que sirvam de entrada para outros sistemas. Isso torna JavaEE uma ótima escolha para migração ou expansão gradativa de soluções já implantadas.
- **Facilidade de implementação** – A sintaxe da linguagem Java, semelhante a C++, é de fácil aprendizado. Além disso, JavaEE possui uma série de bibliotecas que tratam de questões não relacionadas à lógica da aplicação em si.

Esses são alguns dos fatores que transformaram JavaEE no padrão mais utilizado mundialmente para a criação de aplicações distribuídas de grande porte, apesar de ser um padrão relativamente novo. Prova de seu sucesso é a grande utilização por empresas de grande porte, como IBM, Oracle e BEA, que possuem soluções JavaEE.

7 Conclusões e trabalhos futuros

O grande problema enfrentado pelos desenvolvedores de sistemas baseados em agentes móveis é a falta de recursos que auxiliem em sua criação, principalmente nas áreas de segurança e mobilidade dos agentes. As ferramentas existentes ainda são muito complexas ou possuem funcionalidades limitadas.

A alternativa proposta é a utilização de aplicações construídas sobre a plataforma Java 2, Enterprise Edition (JavaEE), que se mostrou bastante adequada ao desenvolvimento de sistemas de agentes móveis. Ferramentas como RMI-IIOP, JNDI, JCA e EJB oferecem recursos que visam a minimizar o esforço dos desenvolvedores com questões que não estejam diretamente ligadas à funcionalidade da aplicação. Isso diminui a complexidade do sistema e, conseqüentemente, a ocorrência de falhas.

Apesar de todas as vantagens observadas com a utilização da plataforma JavaEE, durante as fases de projeto de implementação podem surgir novas dificuldades, que não fazem parte do escopo dessa etapa do trabalho.

Para as próximas etapas, será realizada também uma comparação com outras abordagens possíveis, notadamente Java Management eXtensions (JMX), Java Dynamic Management Kit (JDMK), Jini Network Technology e JavaEE Management Extensions. Outra abordagem interessante é a utilização de agentes móveis em dispositivos embarcados, como celulares e smartphones, que normalmente possuem suporte a Java.

8 Referências bibliográficas

CISCO SYSTEMS, Documentation Center. **Internetworking Technology Handbook**. New York: Cisco Press, 2005.

PRAS, Aiko. **Network Management Architectures**. Twente: Ph.D. thesis, University of Twente, 1995.

DOOLEY, Kevin. **Cisco Cookbook**. O'Reilly, 2003.

SCHMIDT, Kevin. **Essential SNMP**. 2nd edition. Sebastopol: O'Reilly, 2005.

MORRIS, Stephen. **Network Management, MIBs and MPLS**. Upper Saddle River: Prentice Hall, 2003.

NILSEN, Jaran. **Managing Software Agents in JavaEE Application Servers**. Grimstad: Agder University College, 2005.

KARNIK, Neeran. **Security in Mobile Agent Systems**. Twin Cities: Ph.D. Thesis. Univeristy of Minnesota, 2001.

MADEIRA, Edmundo Roberto Mauro. **A Mobile Agent-based Model for Service Management in Virtual Active Networks**. Campinas: Universidade Estadual de Campinas, 2001.

ROMAN, Ed. **Mastering Enterprise JavaBeans**. 3rd edition. Indianapolis: Wiley Publishing, 2005.