UNIPAC - Universidade Presidente Antônio Carlos

Faculdade de Ciência da Computação

Trabalho de Conclusão de Curso

Agentes Inteligentes de Software Móvel em Gerenciamento de Redes Cliente-Servidor

Por:
Luciano Gonçalves Moreira
Graduando do 8º período do Curso de Ciência da Computação

Orientador:
Prof. Luís Augusto Mattos Mendes

Banca:
Élio Lovisi Filho;
Gustavo Campos Meneses.

Professores da Faculdade de Ciência da Computação da UNIPAC – Universidade Presidente Antônio Carlos

Resumo. AISM podem ser usados para, entre outras funções, dar mais dinâmica e autonomia, tanto ao *Software* de Administração/Gerência de redes, quanto ao *Software* do usuário; podem ajudar a executar ações automáticas em nome do Administrador da rede ou do usuário comum diminuindo assim, o tráfego de dados pela rede e consequentemente a latência da mesma; descobrir e reagir a padrões de comportamento da rede; tudo isso devido principalmente à sua característica de mobilidade, o que torna sua utilização promissora como paradigma de computação em rede. Será proposto ao fim deste artigo, um AISM para gerenciamento de redes,que terá a finalidade de trocar informações com agentes e Sistemas de gerenciamentos atuais, ambos baseados no protocolo SNMP. Além de técnicas, ferramentas, aplicativos e linguagens de programação para futura modelagem, desenvolvimento e implementação do AISM.

1. Introdução

Com a expansão da Internet, sistemas de hipermídia e a redução dos preços de componentes de *hardware* e *software*, contribuíram para a criação de um modelo com Aplicações Distribuídas, ou seja, de computação(processamento de informações) centrada em rede. Neste modelo, a própria rede é a inteligência do sistema, ao contrário, por exemplo, do modelo Cliente/Servidor. Os aplicativos de *software* e sistemas de informação são armazenados em rede e o cliente o consulta só quando for necessário. Uma das opções de desenvolvimento desse modelo está baseado em Agentes Inteligentes de *Software* Móvel(AISM), que são os encarregados de estabelecer interfaces e gerenciar a informação, a fim de satisfazer as necessidades do cliente. Além disso os AISM também podem ser utilizados no gerenciamento dinâmico de grandes configurações, monitoramento, detecção e correção de falhas em dispositivos de rede, apresentando-se assim como uma próspera e eficiente solução para o gerenciamento de redes modernas.

Sendo que para ambos os casos, a mobilidade, além de outras características dos AISM, permitem a implementação de Sistemas de Gerenciamento e Aplicações de *software* de usuários distribuídos, com tráfego de dados reduzido e capacidade de respostas rápidas a alterações ocorridas no domínio da rede. As várias tendências favoráveis para a adoção deste modelo são, entre outras, o aumento da complexidade do relacionamento Cliente/Servidor, a necessidade de gerência de grandes volumes de informação, desenvolvimento nos sistemas de comunicação e a entrada no mercado de pessoas com pouco treinamento em computação.

2. Agentes Inteligentes de Software

Existem diversas classificações da área de Inteligência Artificial(IA), que se diferenciam pela manipulação do conhecimento, ou seja, como adquiri-lo, armazená-lo e empregá-lo. Classificando a IA quanto ao método de solução de problemas tem se a IA Simbólica(IAS), a IA Conexionista (IAC), a IA Evolucionária (IAE) e a IA Híbrida (IAH).Quanto a localização espacial tem-se ainda a IA Monolítica (IAM) e a IA Distribuída (IAD), sendo esta última a que será tratada neste trabalho. O comportamento da IAD depende de um determinado conjunto de partes (ou módulos), que funcionam de modo relativamente independente, para resolver de modo cooperativo um determinado problema. Um modelo mais cooperativo é o de Agentes Inteligentes Software(AIS)[FRANCESCHI 2002].

Não existe um consenso entre os pesquisadores sobre a exata definição de Agente Inteligentes de Software. Um AIS pode ser entendido como um software capaz de executar uma tarefa em nome de um usuário, automaticamente, ou seja sem a intervenção humana. Uma definição mais abrangente seria em [RUSSELL 1995]:um agente é algo ...um software,... que pode ser entendido como, situado em um ambiente, percebendo este ambiente através de sensores, age sobre o mesmo através de seus atuadores, ... de acordo com sua base de conhecimento.

A diversidade e complexidade de papéis que os AIS podem assumir, tornam uma definição exata muito difícil. Para auxiliar ou até mesmo substituir uma definição formal muitos cientistas criaram e sugerem, características gerais que podem ser associadas a um AIS. Uma lista delas e bem abrangente é apresentada abaixo e divididas em duas categorias conforme [CARDIERE 1998]. A primeira está associada a uma noção fraca sobre AIS que define a abordagem inicial das pesquisas na área :

“Autonomia: Agentes devem operar sem a intervenção humana e devem ter uma espécie de controle sobre suas ações e estado interno.

Cooperação: Agentes interagem com outros agentes e possivelmente seres humanos através de algum tipo de linguagem específica.

Reatividade: Um agente "percebe" seu ambiente e responde prontamente a alterações que ocorram neste. Isto pode sugerir o fato de que um agente gasta grande parte do seu tempo em uma espécie de "sleep state" da qual ele acorda se algo muda em seu ambiente (p. ex. a chegada de um e-mail)

Proatividade: Um agente não só responde a mudanças em seu ambiente como também está apto a exibir um comportamento direcionado a objetivos tomando iniciativas.

Continuidade temporal: Agentes são processos que estão continuamente rodando podendo estar ativos ou passivos("sleeping").

Orientação a eventos(goal): Um agente é capaz de suportar tarefas complexas, decidindo se deve dividi-las em tarefas menores e em qual ordem deve executá-las”

E a segunda esta associada a uma noção mais forte, mais sólida de agentes é utilizada principalmente por pesquisadores do campo de Inteligência Artificial :

“ Mobilidade: A habilidade de um agente em se mover sobre uma rede eletrônica.

Benevolência: Assume-se que os agentes não possuem conflitos de objetivos e sempre tentarão fazer o que for solicitado a eles buscando cooperação.

Racionalidade: Um agente sempre vai agir de forma a alcançar seus objetivos e nunca de maneira que prejudique obtê-los.

Aprendizado/Adaptabilidade: Um agente deve estar apto a aprender e a se ajustar aos hábitos, método de trabalho e preferências do usuário.

Colaboração: Um agente não deve impensadamente executar instruções, mas deve considerar que os seres humanos cometem erros, seja através de instruções mal formuladas ou omissão de informações. Neste caso os agentes devem certificar-se através de perguntas formuladas aos usuários para resolver problemas como este. Um agente pode também se recusar a executar determinadas tarefas se estas causarem danos à outros usuários”.

Porém não há ainda nenhum AIS com todas estas características implementadas, embora existam sim diversos projetos que apresentem algumas destas. Também não se tem ainda um consenso entre os cientistas sobre a importância de cada uma destas características , mas a maioria concorda que são estas as responsáveis pela diferença entre um AIS e um programa convencional.

2.1. Agentes Inteligentes de Software Móvel

Agentes Inteligentes de Software Móvel (AISM), cuja característica principal é a mobilidade, ou seja, capacidade de se transportarem entre diferentes nós de uma rede, ou entre diferentes redes, como no caso da Internet. Suas principais áreas de aplicação são sistemas de computação móvel, Aplicações Distribuídas e gerenciamento de redes . Para estas áreas onde são aplicados os AISM, é necessário utilizar uma plataforma de *software* que defina e implemente os mecanismos de mobilidade e de comunicação dos agentes . Tal plataforma, denominada Sistema de Agentes Móveis (SAM), oferece suporte às operações fundamentais de criação, descarte, persistência e transferência dos AISM's, como exemplificado no Anexo A no fim deste artigo[LOBATO 2003].

Um AISM se movimenta em um ambiente distribuído, entre os *hosts* de uma rede aos quais possui acesso. O acesso é configurado através de um servidor de agentes (oferecido pelo SAM, instalado nos *hosts*), que, dentre outras funções, possui a tarefa de fornecer aos AISM um ambiente adequado à sua execução. Como demonstra a figura 1, quando um AISM é transferido na rede, sua execução é interrompida no *host* original, ele é transportado para outro no ambiente distribuído e sua execução é reiniciada a partir do ponto de interrupção anterior. A migração dos AISM pode ser classificada em fraca ou forte. Na migração fraca, o AISM reinicia a execução em um novo *host* a partir da primeira linha de seu código. Na migração forte a execução reinicia do ponto onde foi interrompida. Ou seja, quando um AISM é transferido de um *host* a outro, este leva consigo o código e os dados de sua execução, fato que lhe permite ser transportado em toda a rede, sem que suas variáveis internas devam ser novamente inicializadas. Isto garante a continuidade temporal(ou persistência*)* de seu estado interno, característica essencial na definição de AISM[LOBATO 2003].

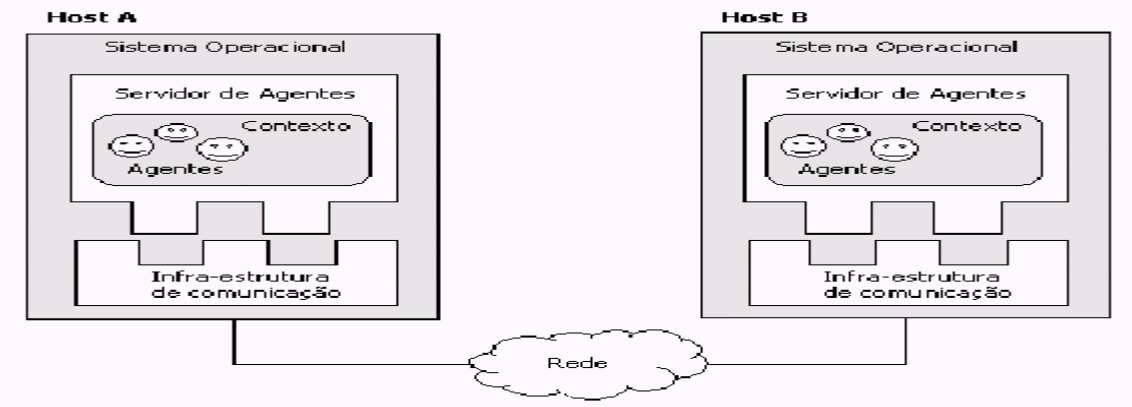

Figura 1.Estrutura Geral de um Sistema de Agentes Móveis Fonte: [LOBATO 2003]

Segundo [LOBATO 2003] ,as principais funções de um Servidor de Agentes são:

- proteção de recursos dos *hosts* : diz respeito à determinação dos limites de atuação dos AISM;

- transferir um AISM : um servidor local negocia com outros servidores, interrompe a execução do AISM e transfere-o para um servidor remoto, onde reinicia sua execução e

- não-interferência entre múltiplos AISM's : é possível porque um servidor fornece aos AISM's uma interface de comunicação, que inibe a interferência direta entre eles, nos casos em que um AISM necessita do serviço de outros, ele deve emitir uma mensagem com o respectivo pedido.

Ainda segundo [LOBATO 2003],a existência de um AISM é representada segundo as etapas de um modelo denominado "ciclo de vida" . Estas etapas correspondem à execução das operações:

- criação : é executada somente uma vez, o AISM recebe um identificador, tem seu estado interno iniciado e é preparado para instruções adicionais;

- inicialização : é efetuada a cada vez que um AISM chega a um novo *host*. Isto só é possível porque um AISM possui sua própria *thread* de execução e é projetado para executar assincronamente;

- desativação : um AISM pára o seu processamento e armazena seu estado interno e resultados intermediários em um disco sendo utilizado para isto, o mecanismo de serialização de objetos;

- destruição : implica na finalização das atividades de um AISM, liberando todos os recursos que está utilizando e

- clonagem : através da qual são criadas múltiplas cópias de um AISM móvel.

3. Linguagem Java para desenvolvimento de AISM's e SAM's

*Aglets Software Development Kit(*ASDK), é uma plataforma(ou SAM) para desenvolvimento de AISM's, denominados *Aglets*. Este SAM foi desenvolvido totalmente na linguagem Java, caracterizada por ser portável, distribuída, segura e multitarefa, atributos essenciais para o desenvolvimento de agentes móveis, motivos pelos quais a tornaram apropriada para implementação de AISM . Os *aglets* são objetos Java projetados para se mover através dos *hosts* de uma rede, que se caracterizam por apresentar a própria *thread* de controle, serem dirigidos a eventos e se comunicarem através de passagem de mensagens. Trata-se da combinação de AIS com o modelo de código móvel dos *applets Java* [LOBATO 2003].

Segundo[LOBATO 2003], dois mecanismos podem ser utilizados para a implementação da mobilidade de AISM: *Remote Method Invocation* (RMI) e *sockets*. RMI é um mecanismo que possibilita invocar métodos entre processos(por exemplo AISM) Java localizados em diferentes *hosts* de uma rede. Por exemplo, isto permite que um AISM desenvolvido em Java chame métodos e acesse variáveis dentro de um aplicativo Java, o qual pode estar executando em diferentes ambientes(servidores de agentes) Java, e troque objetos através de uma conexão em rede; sem a necessidade de saber previamente qual protocolo é utilizado na comunicação. O RMI transmite de fato objetos inteiros através de uma rede, sendo, então, o mais conveniente para um modelo distribuído totalmente orientado a objetos.

Por este e outros motivos o RMI também pode ser utilizado como mecanismo de comunicação entre AISM e sistemas de agentes. Além deste, há outros que podem ser usados, por exemplo: RPC ou chamada a procedimento remoto, rechamada assíncrona, caixa postal. No RMI, o processo comunicação ocorre como via RPC, enviam chamadas de procedimento com argumentos sendo passados ou descritos de tal modo que possam ser reconstruídos do lado remoto. Porém, apesar das características descritas acima torná-lo o mecanismo ideal na migração e por oter bons resultados na comunicação entre AISM 's, o RMI somente pode ser utilizado nos SAM's desenvolvidos em linguagem Java, o que não o impediu de ser o mais utilizado atualmente, principalmente porque o Java é a melhor linguagem para desenvolvimento de AISM's. Para os SAM's desenvolvidos em outras linguagens é utilizado o *sockets.*

Definir um *aglet* como um objeto com sua própria *thread* é uma forma prática de implementar em Java a característica de autonomia, pois tem sua própria linha de processamento e pode executar concorrentemente a outros *aglets* no mesmo *host* , para isso deve-se indicar que as classes do programa implementem a interface *Runnable* do pacote *java.lang*. Para ter reatividade, em sua API, os *aglets* dispõem de vários métodos que executam ações em resposta a estímulos recebidos. O método *getAgletInfo,* funciona como sensores, por exemplo, sendo responsável por retornar informações sobre um *aglet* específico no contexto de execução. O método *setProperty* é um exemplo de atuador que permite a um *aglet* além de ler, altere propriedades do servidor em que está inserido. O método *sendMessage* afeta o comportamento de *aglets* através do envio de mensagens[LOBATO 2003].

A persistência(ou continuidade temporal) em agentes é manter a consistência de seus atributos internos ao longo de seu ciclo de vida, sendo que *"...é a* serialização de objetos, *através da qual é implementada a característica de persistência nas etapas de clonagem, ativação, desativação e movimentação dos aglets"* , daí temos que a criação se dá através do método *createAglet*; clonagem é realizada através do método *clone*; desativação o método *deactivate* é utilizado; inicialização através do método *activate;* destruição realizada através do método *dispose.* A plataforma ASDK permite que *aglets* sejam movidos entre nós de uma rede através de chamadas aos métodos *dispatch* e *retract*. O método *dispatch* é utilizado para enviar *aglets* de um *host* local a outro *host* na rede, enquanto o *retract* permite que um *host* faça o retorno de *aglets* despachados anteriormente que se localizam em contextos remotos [LOBATO 2003].

A forma de utilização da linguagem Java para implementação esta demonstrado no item 8.1 e no Anexo B. E a utilização do ASDK esta demonstrado no item 8.2 e Anexo A.

4. Uso de AISM's em Aplicações Distriduídas

Os avanços na tecnologia aplicada a redes de computadores têm enfatizado a investigação de AISM como um paradigma promissor no desenvolvimento de *softwares* complexos e distribuídos para redes de computadores. Para compreender as vantagens de utilização dos AISM em Aplicações Distribuídas, é necessário comparar suas características às dos atuais paradigmas da computação em rede , de modo a efetuar um paralelo das principais diferenças existentes entre as abordagens tradicionais e aquelas baseadas em código móvel.

No paradigma cliente-servidor, os componentes computacionais e o código das aplicações não podem ser transferidos após sua criação. Já os paradigmas de execução remota e de código sob demanda(que não serão tratados neste trabalho) superam esta limitação pois fornecem a característica de mobilidade ao código processado pelos componentes; isto é, estes paradigmas permitem a execução de código em uma localização remota. No entanto, no extremo da facilidade de transferência, encontra-se o paradigma de agentes móveis que fornece mobilidade não somente ao código que deve ser processado, mas aos próprios componentes que efetuam a execução dos serviços[LOBATO 2003]

Conforme [LOBATO 2003], para melhor entendimento vamos comparar de forma mais aprofundada o paradigma cliente-servidor que é o mais utilizado, com o paradigma de agentes móveis. Ambos demonstrados também através das figuras 2 e 3 respectivamente:

"O paradigma cliente-servidor é largamente utilizado. Neste paradigma, um componente computacional B, denominado servidor e localizado em um site Sb, oferece um conjunto de serviços na rede. O código e os recursos necessários à execução destes serviços estão disponíveis no site Sb. Um componente A, denominado cliente e situado em um site Sa, solicita ao servidor a execução de uma tarefa. Como resposta, o componente B processa o código da tarefa solicitada por A, utilizando, para isso, os recursos que estão disponíveis em no site Sb. A execução da tarefa produz um resultado que é enviado pelo servidor em reposta à solicitação do cliente. Exemplos de aplicação deste paradigma: páginas dinâmicas com script interpretado no servidor, Common Object Request Broker Architecture (CORBA), RMI, etc" .

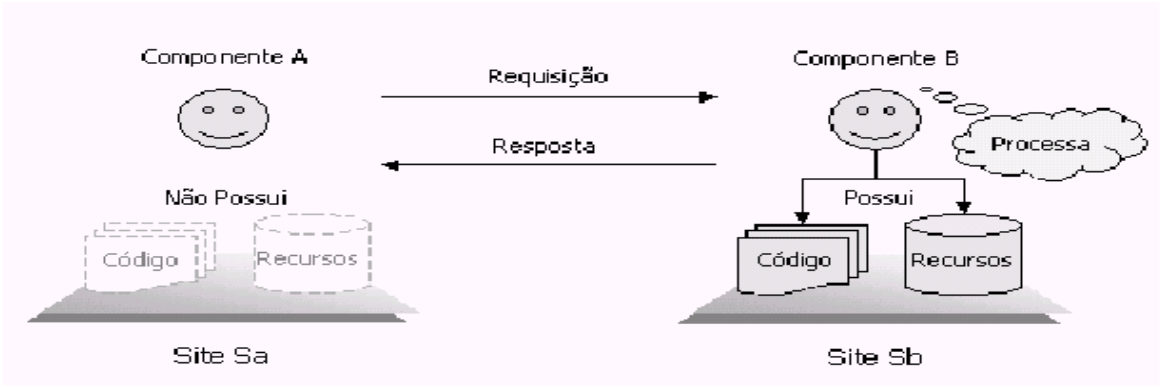

Figura 2. Paradigma Cliente-Servidor. Fonte : [LOBATO 2003].

"No paradigma de agentes móveis, um componente computacional A, hospedado inicialmente no site Sa, possui o código de um serviço que precisa ser processado; porém, alguns dos recursos necessários ao processamento estão disponíveis apenas em outros sites, como no site Sb. A então migra para Sb, carregando o código e resultados intermediários de sua tarefa. O componente A termina o processamento utilizando os recursos disponíveis em Sb. Exemplos: agentes móveis em aplicações de busca de informações, comércio eletrônico, gerência de redes, computação móvel, etc" .

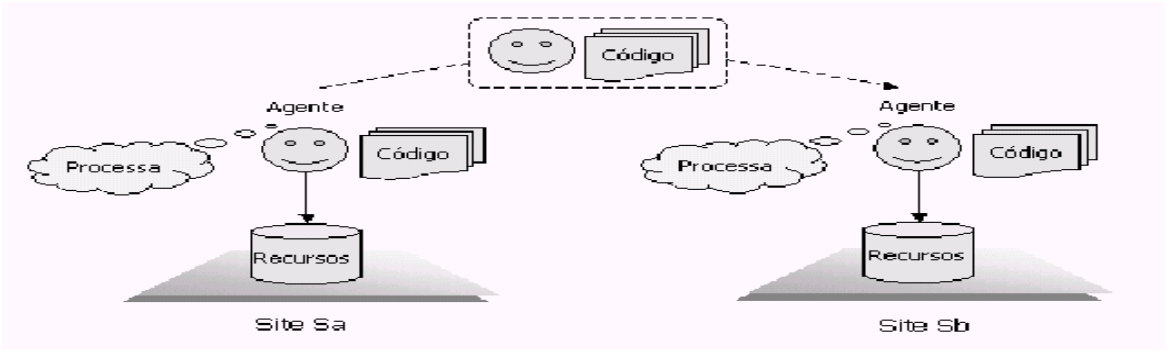

Figura 3. Paradigma de Agentes Móveis. Fonte : [LOBATO 2003].

No paradigma cliente-servidor o Cliente necessita de serviços fornecidos pelo servidor; que também possui o conhecimento e os recursos para processar os serviços, ou seja, o servidor controla todo o processamento. Já no paradigma de agentes móveis há alto grau de flexibilidade nas estações, o conhecimento está distribuído na rede, os AISM migram para onde estão os recursos, o *host* original ou hospedeiro do AISM controla o processo.

Uma restrição concreta do paradigma cliente-servidor esta no fato do comportamento limitado de seus componentes, ou seja, o cliente sempre e somente toma a iniciativa da comunicação, enquanto o servidor somente atua, após as solicitações do cliente, o que foi determinada por elas. O estado de uma transação encontra-se dividido entre os componentes cliente e servidor. Se ocorre uma falha(perda de conexão ,por exemplo) de algum dispositivo da rede durante o processamento de uma solicitação, torna-se difícil recuperar o estado da transação interrompida e reiniciar-se o processo de sincronização entre os componentes[LOBATO 2003].

Em oposição, os AISM, apresentam características de cliente e de servidor em diferentes instantes de seu ciclo de vida; por manterem o encapsulamento de seu estado não necessitam de conexões de rede permanentes. Conclui-se portanto que, enquanto nos outros paradigmas, a mobilidade que se tem, diz respeito a uma possível transferência de código entre componentes(cliente e servidor, por exemplo) computacionais; o paradigma de agentes móveis se diferencia dos demais principalmente porque envolve a mobilidade de um ou mais componentes(por exemplo, do cliente ou servidor) computacionais completo é movido para um local remoto, juntamente com seu estado, código, e até mesmo alguns recursos necessários para o processamento dos serviços[LOBATO 2003].

5. Uso de AISM's na Gerência de Redes

Varias técnicas têm sido implementadas pelos pesquisadores na procura por soluções para os problemas do gerenciamento de redes, tendo em vista a diversidade de dispositivos de *hardware* e plataformas de *software* utilizados, o que requer cada vez mais dedicação e atenção dos administradores de rede para um bom gerenciamento.

5.1 Gerenciamento de Redes Tradicional

No sentido de minimizar a complexidade da gerencia de redes, a ISO/OSI subdividiu-a em cinco áreas funcionais segundo [SOARES 1995]

- Gerência de falhas. Trata a localização de problemas, isolando-os e solucionando-os quando possível. Tendo como causas mais prováveis para que ocorram problemas, *"os erros de projeto e implementação da rede, erros de sobrecarga, distúrbios externos, tempo de vida útil de equipamentos expirado e má implementação de softwares (famosos bugs)"*

- Gerência de configuração. Responsável pelas configurações dos dispositivos que compõem a rede.

- Gerência de desempenho. Responsável pelo monitoramento do desempenho dos equipamentos e *softwares* pertencentes a rede através dos registros de algumas taxas de medida como, erros, utilização e tempo de respostas. Visando otimizar a qualidade dos serviços da rede.

- Gerência de contabilização. Controlam, quantos e quais recursos e de que forma são utilizados pelos usuários , como também garante ou removem permissões de acesso à rede.

- Gerencia de segurança. Responsável pelo controle do acesso às informações disponíveis na rede, administrando também as tentativas de entrada na rede.

Uma arquitetura clássica de gerenciamento de redes utiliza-se de interações entre os dispositivos de uma rede e uma estação de gerenciamento. Um dispositivo interage com a estação através de um processo de *software* da aplicação distribuída de gerenciamento, denominado "agente"(*software* normal sem autonomia, nem mobilidade). Este tem a finalidade de fornecer uma visão externa a respeito dos parâmetros do dispositivo que são configuráveis. Na estação de gerenciamento, todas as informações coletadas pelos "agentes" são processadas, através de um processo da aplicação distribuída de gerenciamento, denominado gerente**,** que interage com a estação de gerência, que ao analisar o estado da rede e conforme o resultado encontrado, o Administrador, determina que ações serão executadas[LOBATO 2003].

A idéia básica destas arquiteturas de gerenciamento é que os processos de *software* destes dispositivos são configurados para atuar de duas formas: ou para serem "agentes" gerenciados ou os próprios gerentes. Seguindo esta arquitetura clássica, um exemplo a ser funcionalmente analisado é, o protocolo SNMP que foi desenvolvido para ser uma ferramenta de gerenciamento de redes estruturado e sistemático. Este protocolo foi e é amplamente implementado em produtos comerciais e por isso foi aceito como padrão para gerenciamento de redes. O modelo de gerenciamento do SNMP, segundo [BRANDÃO] consiste em:

- *"Diversos nós gerenciados, cada um com uma entidade SNMP que provê acesso remoto às*

  informações gerenciadas, geralmente chamados de agentes.

- *Pelo menos uma entidade que rode aplicações de gerenciamento, chamada de gerente.*

- *Um protocolo de gerenciamento usado para trocar as informações entre gerente e agentes (o protocolo SNMP).*

- *As informações gerenciadas"*.

De acordo com o modelo SNMP ,os nós gerenciados podem ser *hosts*, ou seja, roteadores, pontes, impressoras, micros computadores, servidores ou qualquer outro dispositivo capaz de trocar informações com o gerente, que esta presente, em pelo menos um destes nós, geralmente nos servidores ou estações próprias de gerência . Os "agentes", processos de *software*, presentes nestes dispositivos(nós gerenciados) tem a função de coletar e armazenar as informações sobre o estado da rede local e de sua periferia, onde esta instalado, fornecendo as informações necessárias ao gerente quando solicitadas ou quando algum evento importante ocorrer e aceitar os comandos enviados pelo gerente para que a configuração local seja alterada. Estes "agentes" são desenvolvidos de forma a serem o mais simples possível, para apenas coletar dados, onde todo o processamento é executado nas estações de gerenciamento[BRANDÃO].

Estas estruturas clássicas das aplicações de gerenciamento de redes não são mais tão eficazes tendo em vista que as redes atuais, tendem a ser cada vez mais complexas, velozes, heterogêneas e com alto trafego de dados. No entanto, neste tipo de estrutura, alguns problemas relacionados ao tráfego de dados podem ocorrer com freqüência : fluxo alto de dados próximo as estações de gerência, dificuldade de manter a consistência dos dados, protocolos e interações não são totalmente rápidos para acompanhar alterações e atualizações quase que constantes nas redes de alta velocidade[LOBATO 2003].

Por isso, existe uma grande preocupação de cientistas, pesquisadores e dos próprios administradores de rede em agilizar, melhorar e automatizar grande parte dos serviços de gerenciamento de redes.

5.2. AISM's Gerenciando uma Rede

Tendo em vista todas as vantagens dos AISM's, é possível a construção de modelos dinâmicos de gerenciamento distribuído, personalizados para melhorar a qualidade dos serviços no monitoramento de redes, com tráfego de dados reduzido e capacidade de respostas rápidas a alterações ocorridas no domínio da rede, prevendo eventos ou aumentando o tempo de resposta aos eventos ocorridos, através do monitoramento automático dos dispositivos[BIANCHINI].Bem como o gerenciamento de grandes configurações, detecção e correção de falhas em dispositivos de rede[LOBATO 2003].

O uso de *softwares* implementados com características de autonomia, aprendizagem, cooperação e mobilidade, ou seja AISM, flexibilizam a aplicação distribuída de gerenciamento descrita acima. Estas característica de aprendizagem, colaboração, autonomia e racionalidade, podem ser feita através de uma Base de Conhecimento(em um *host*) onde cada AISM ao interagir com a mesma pode atualizar com novos conhecimentos adquiridos, para que estes possam ser utilizados por outros para auxilia-los em futuras decisões. O que evita uma implementação de mecanismos de colaboração, aprendizagem e conhecimento em todos os AISM's.

6. Benefícios da Utilização de AISM em Aplicações e Gerenciamento Distribuídos

Com o uso de AISM's em Aplicações e Gerenciamento de redes Distribuídas constatou-se ,

que AISM's tem **adaptação dinâmica** às mudanças ocorridas em seu ambiente de execução. Em Gerenciamento de rede, mais um exemplo, um sistema de gerência pode distribuir AISM em uma rede, para que estes possam manter o melhor esquema de configuração dos dispositivos da rede em dado momento, a fim de resolver o problemas que ocorram. Ou seja, o grupo de agentes aplica uma configuração, através da qual é obtida a eficiência máxima no funcionamento da rede, naquele momento em que ocorre o problema, para corrigi-lo ou para que seja corrigido pelo administrador da rede, que pode vir a ser automática sendo utilizada neste caso uma Base de Conhecimento com as informações necessárias.

Esta habilidade que os AISM tem de reagir dinamicamente aos mais diversos tipos de situações, inclusive as mais adversas, contribui para o desenvolvimento de sistemas distribuídos mais robustos e **tolerantes a falhas**. No caso de problemas com um *host*, todos os AISM ativos neste *host* são invocados em tempo hábil, a fim de que possam se mover e assim continuar com a execução de suas tarefas em outros *hosts* da rede, quando projetados para utilizar uma alternativa de migração entre *hosts*. Além da adaptação dinâmica e da tolerância a falhas, a implementação de AISM's confere às aplicações de sistemas distribuídos outros cinco benefícios principais, conforme[LOBATO 2003]:

Redução do tráfego de dados na rede. Nos sistemas com Aplicações Distribuídas, os AISM são úteis pois reduzem o fluxo de dados em uma rede, porque permitem que as solicitações de um aplicativo sejam despachadas junto com o agente para um *host* onde a interação entre dados e processamento possa ocorrer localmente. A utilização de AISM se fazem necessário pelo fato de que, em termos de tráfego de dados na rede, a solução mais adequada é transferir as requisições de uma aplicação de usuário, ao local onde os dados remotos estão armazenados para que sejam processados, do que transferir estes dados ao *host* do usuário onde possam ser processados.

Redução da latência de rede. Considerando sistemas críticos, como o controle de tráfego aéreo por exemplo, que necessitam responder em tempo real às mudanças ocorridas no ambiente. Sistemas como podem ser suscetíveis a latências de rede mais ou menos significativas; isto é, quando os programas desses sistemas de controle são processados de forma centralizada, o tráfego de dados na rede pode retardar de maneira inaceitável as respostas do sistema, ainda que corretas. Uma possível solução para este problema envolve a transferência de AISM para controlar localmente os processos da aplicação.

Encapsulamento de protocolos. Em sistemas distribuídos, cada *host* possui os protocolos necessários, para a interpretação dos dados que chegam a ele e também para a codificação dos dados que devem ser enviados por ele. Tais protocolos sofrem freqüentemente atualizações com novas exigências de eficiência ou segurança. Isto acaba por se tornar uma tarefa incômoda, às vezes impossível de ser realizada satisfatoriamente. Para corrigir este problema, os AISM podem se mover para os *hosts* da rede com o objetivo de estabelecer canais de comunicação baseados em protocolos proprietários. Diminuindo a necessidade de atualização freqüente dos protocolos.

Execução autônoma e assíncrona. Com o crescimento da Computação Móvel, a conexão e desconexão dos dispositivos portáteis a uma rede tornou-se um tópico de pesquisa importante. Pois tais dispositivos constantemente se utilizam de conexões de rede fixas frágeis e caras, isto significa, que aplicações que necessitam de conexão continua, entre um dispositivo móvel e este tipo de rede, muito provavelmente sofreram com quedas e até perda total da conexão, por isso não são nem econômica nem tecnicamente viáveis. Uma solução para este tipo de situação seria encapsular solicitações dessas aplicações juntamente com AISM's e despachá-los para o *host* desejado na rede fixa, onde possa ser realizado o processamento da tarefa desejada; pois mesmo que se perca a conexão ou necessite desconectar o dispositivo, ao ser reconectado mais tarde este faça retornar o AISM já com os resultados encontrados, sem que haja necessidade de qualquer tipo de sincronização entre a conexão e o processamento das tarefas. Isto é possível porque, após se moverem para o *host* desejado, os AISM's tornam-se independentes do *host* onde foi originado e podem operar autônoma e assincronamente, quando possuírem dados e códigos para.

Integração de sistemas heterogêneos.A computação em rede é e sempre será heterogênea. Uma vez que os AISM, desenvolvidos em Java principalmente, são independentes das características de *hardware* ou *software* dos sistemas onde são executados, como por exemplo sistema operacional e camada de transporte dos protocolos de rede, dependendo somente de sua plataforma(SAM) de execução para atuar autonomamente. Isto faz com que os AISM sejam excelentes mecanismos para a integração de sistemas heterogêneos.

7. Conclusão

De acordo com o trabalho elaborado, nota se que as aplicações de AISM em serviços de computação em rede é um dos tópicos mais estudados no mundo. Isto, tanto para aplicações de *Softwares* para usuários finais que se utilização da rede para executar sua tarefas, quanto

para Aplicações para Administração/Gerência de redes.Mas, para fazer a migração de todo um sistema de Aplicações distribuídas, que normalmente envolve diversos tipos de Aplicações de Software para diferentes fins, inclusive Administração/Gerência da rede, estando este sistema operando geralmente sob o Paradigma Cliente/Servidor, migrar para, o Paradigma Agentes Móveis pode ser um tanto quanto oneroso para qualquer empresa, mas é claro que a viabilização desta migração vai depender do interesse, do porte e da quantidade de informação dessa empresa, que trafega pela rede.Porém no caso de Administração/Gerência de redes o investimento pode não ter um custo muito elevado, pois se trata somente de Aplicações voltadas para Administração/Gerência de redes, tornando esta migração mais viável, tendo em vista diminuição do fluxo de dados próximos as estações de gerência, realização automática de algumas tarefas do Administrador e a não necessidade de se ter diversas estações de gerência, modernas e com *softwares* de Administração/gerência caros e sofisticados, espalhadas pela rede. Em ambos os casos, o fator que com certeza viabiliza a utilização deste novo Paradigma da Computação em rede, é a considerável diminuição do tráfego de dados pela rede e consequentemente a latência (tempo de resposta), segurança, entre outros benefícios descritos no trabalho. Contribuindo portanto para um melhor desempenho da rede.

8. Trabalhos Futuros

De fato, estes benefícios e as boas perspectivas do uso de AISM na computação , constituem verdadeiras razões pelas quais os agentes móveis devem ser utilizados em sistemas distribuídos. Sendo assim pode se Ter o seguinte modelo:

Um AISM hospedado em um *host* tem a função de coletar e analisar os dados deste, através mensagens via protocolo SNMP, aos agentes SNMP instalados neste mesmo *host*. O AISM ao analisar o comportamento do seu *host* na rede, conforme seus parâmetros este então decide também conforme sua implementação, se modifica qualquer parâmetro do funcionamento do *host* na rede através do agente SNMP, ou se é transferido para uma estação ou servidor mais próxima que contenha uma base de conhecimento implementada, onde realizará inferências ate descobrir qual será sua atuação. Neste caso o após a consulta o AISM executa o que foi determinado pela base de conhecimento, retornar ao *host* de origem e alterar qualquer parâmetro no funcionamento deste, ou ser despachado para uma estação principal onde seus dados coletados possam ser analisados por um Administrador e este então toma a decisão que achar conveniente. E essa, pode ser despachar o AISM para seu *host* original(ou outro), com alteração em seu corpo e executar modificações neste *host*; e ou alterar, acrescentar informações na base de conhecimento para reconhecer problemas futuros.

8.1 Implementação do AISM proposto

Para implementação do AISM utilizaremos a linguagem Java, através do kit JDK, cuja versão sugerida é JDK 1.4 ou superior, onde poderá ser utilizada uma plataforma para desenvolvimento de *Applets* que pode, por exemplo, ser o Eclipse a partir da sua versão 2.1; ou qualquer ferramenta Java mais atual para o desenvolvimento de aplicações *Applets*, encontradas no endereço http://www.sun.com .Com tudo, ainda é necessário fazer o download do *Aglets.class*, classe que contém os principais métodos que cobrem o ciclo de vida do AISM e outros necessários a algumas de suas funções essenciais como, por exemplo, enviar e

receber mensagens entre outros .A versão 1.0.3 desta classe pode ser encontrada no endereço http://din.uem.br/~ia/vida/agentes/download.htm .Bem como importar o pacote *Aglet2_0_2.jar* contido no ASDK para o pacote onde esta sendo implementado o projeto. Como, por exemplo, pode ser visto no Anexo B a tela principal do Eclipse com as classes e os *imports* necessários.

8.2 Plataforma de Execução dos AISM's

Para utilização dos AISM como visto antes precisaremos de uma plataforma ou ambiente de execução dos mesmos. Neste caso vamos fazer uso do Aglets 2.0.2, cujos aplicativos e a documentação podem ser encontrados no endereço http://sourceforge.net/projects/aglets. Para instalar a plataforma *Aglets*, pode ser necessário utilizar a ferramenta *ant*. Existe uma cópia do *ant* no Aglets 2.0.2, porém, por razões de incompatibilidade com os sistemas, pode ser necessário efetuar o *download* da última versão da ferramenta. Nesta situação, o *download* do *ant* pode ser realizado através do endereço http://ant.apache.org/ .como é mostrado em [LOBATO 2003]. O Anexo A apresenta alguns exemplos de utilização da ferramenta.

8.3. Construção da Base de conhecimento(opcional)

Para coleta de informações e representação da mesma, podemos utilizar as técnicas para desenvolvimento de aplicações. Para implementação da mesma, utilizaremos a ferramenta JPE *(Java Prolog Environment)*, cuja documentação e aplicativo podem ser encontrados no endereço http://jpe.dcc.ufrj.br/pt/versões.php para download. O Anexo C apresenta a tela principal da ferramenta.

9. Referências Bibliográficas

[FRANCESCHI 2002] FRANCESCHI, A. S. M. ; ROISENBERG, M.;BARRETO J. M. Desenvolvendo Agentes de Software para Gerência de redes Utilisando Técnicas de Inteligencia Artificial.

[CARDIERE 1998] CARDIERE, M. A. C. A. "Agentes Inteligentes: Noçoes Gerais." Monografia, Julho/1998 - Faculdade de Engenharia Elétrica e Computação, Unicamp.

[BRANDÃO]BRANDÃO, A. J. S.; MOREIRA, E. S. "Agentes Móveis e Sistemas de Gerenciamento SNMP" – *Mathematics and Computing Institute*, Universidade de são Paulo. Disponível em http://www.ppgia.pucpr.br/~maziero/pesquisa/ceseg/wseg02/07.pdf .

[LOBATO 2003] LOBATO C. A.; DAMASCENO K. N. F. "Agentes Móveis Aglets na Busca de Informações". Trabalho de Conclusão de Curso(Bacharel) – Centro de ciências Exatas e Naturais, Universidade federal do Pará. Disponível em http://www.cultura.ufpa.br/informatica/tcc/karla-cidiane.pdf .

[BIANCHINI] BIANCHINI, C. P.; ALMEIDA, E. S.; FONTES, D. S.; ANDRADE, R. M. C. "Um Padrão para Gerenciamento de Redes". Disponível em http://www.cin.ufpe.br/~sugarloafplop/final_articles/15_GerenciamentoInteligentesRedes.pdf

.

[SOARES 1995] SOARES, L. F.G.; LEMOS G.; Colcher S.; “ Redes de computadores das LAN’s, Man’s e Wan’s. 2ª Edição. Rio de Janeiro, 1995. Ed. Campus.

[RUSSELL 95**]** RUSSELL, S.; NORVIG P.; *“Artificial Inteligence: A Modern Approach”*. Prentice-Hall, 1995.