Monitoramento de Qualidade de Serviço em Sistemas Multimídia Distribuídos: um Estudo de Caso para Sistemas Baseados em Agentes Móveis

Paulo César de Oliveira
Centro Nacional de Pesquisa Tecnológica em Informática para a Agricultura (CNPTIA/EMBRAPA)
Av. Dr. André Tosello, s/n.º – Cidade Universitária "Zeferino Vaz"
CP 6041 – CEP 13083-970 – Campinas – SP
e-mail: paulo@cnptia.embrapa.br

Luiz Affonso Guedes
Universidade Federal do Pará
Belém – PA
e-mail: affonso@dca.fee.unicamp.br

Eleri Cardozo
Faculdade de Engenharia Elétrica e de Computação (FEEC)
Universidade Estadual de Campinas (UNICAMP)
CP 6101 – CEP 13083-970 – Campinas – SP
e-mail: eleri@dca.fee.unicamp.br

Resumo

Sistemas baseados em agentes móveis (SBAM) surgiram recentemente como uma abordagem promissora para o desenvolvimento de sistemas distribuídos. Atualmente, poucas experiências de utilização desta abordagem estão relatadas na literatura. Este artigo trata do emprego de SBAM no domínio de monitoramento de qualidade de serviço em sistemas multimídia distribuídos. Uma descrição detalhada da arquitetura proposta para desenvolvimento, aspectos de implementação, e resultados obtidos a partir da utilização do SBAM desenvolvido são descritos neste artigo.

Abstract

Mobile agent-based systems (MABS) have recently emerged as a promising approach for distributed systems development. Currently, few experiences regarding the deployment of such approach are reported in the literature. This paper addresses the deployment of MABS in the domain of quality of service monitoring in distributed multimedia systems. A detailed description of the proposed development architecture, some implementation aspects, and the results obtained from the utilization of the developed MABS are described in this paper.

1. Introdução

A abordagem baseada em agentes de software (no escopo deste trabalho serão chamados apenas de agentes) constitui-se numa alternativa para o desenvolvimento de sistemas distribuídos. Embora os termos *agente* e *sistema baseado em agentes* (SBA) sejam amplamente utilizados em diversas áreas da computação, não há uma definição de consenso para o seu significado.

Agentes podem ser definidos como programas autônomos que recebem autoridade de seus donos (usuários ou outros programas) para agir em favor destes na execução de tarefas. Visando completar as tarefas que lhes foram atribuídas, agentes podem se comunicar com outros programas, com o ambiente que os abriga ou com seres humanos [Oliv97]. Esta definição se concentra nas seguintes características fundamentais de agentes:

- autonomia: um agente pode tomar decisões próprias, baseando-se nas metas, preferências e políticas definidas por seus donos ou a eles associadas;
- delegação: usuários ou outros programas podem delegar tarefas a agentes e revesti-los com autoridade para agir em seu favor;
- comunicação: habilidade que agentes têm de interagir com outros programas (agentes ou não), com ambientes que os abrigam e com seres humanos;
- flexibilidade: agentes possuem interfaces e comportamento bem definidos, e não assumem papéis fixos; eles podem agir como clientes, servidores, observadores, etc., dependendo de suas necessidades correntes;
- eqüidade: agentes comportam-se como pares; não há relações de hierarquia entre eles.

As características apontadas como fundamentais refletem a visão de que um sistema de software baseado em agentes é composto por entidades (agentes) autônomas que executam tarefas a elas atribuídas (delegadas) por seus donos. Cada entidade é vista como par pelas outras entidades que compõem o sistema e exerce papéis flexíveis com interfaces e comportamento bem definidos. Modularidade e encapsulamento são enfatizados através de agentes. Além disso, paralelismo pode ser explorado, visto que agentes são entidades autônomas.

Embora não consideradas como fundamentais, as características abaixo estão extremamente relacionadas a agentes:

- cooperação: agentes podem agir de forma colaborativa, visando atingir metas comuns;
- inteligência: habilidade apresentada por agentes de raciocínio e aprendizado a partir de interações com seres humanos, outros programas e ambientes que os abrigam;
- mobilidade: agentes podem se mover numa rede de computadores heterogênea, visando progressivamente completar tarefas que lhes foram atribuídas.

A característica de mobilidade, aliada às características fundamentais, torna *sistemas baseados em agentes móveis* (SBAM) uma abordagem singular para o desenvolvimento de sistemas distribuídos [Oliv97].

Diferentemente de outras abordagens amplamente utilizadas, tal como o modelo Cliente/Servidor baseado em RPC, em SBAM agentes (compostos por dados e código) se movem através da rede, conforme ilustrado na figura 1.

Figura 1: Migração de um agente móvel para a utilização de recursos remotos

Visto que agentes se movem através de redes potencialmente heterogêneas, seu código deve ser executado de maneira idêntica em cada nó para onde eles podem ser transportados [Ling95]. Desta forma, o seu código deve ser independente de plataforma.

Sistemas baseados em agentes móveis apresentam as seguintes vantagens:

- estruturação intuitiva de sistemas distribuídos: componentes de sistemas são entidades autônomas, com vida própria [ANSA95];
- arquitetura flexível para computação distribuída: agentes têm interfaces e comportamento bem definidos, e não assumem papéis fixos [Nwan96];
- computação assíncrona: agentes podem ser despachados para outros nós enquanto outras tarefas são executadas por seus donos [Nwan96];
- redução de custos de comunicação: ao contrário de paradigmas clássicos, agentes se movem até os locais onde recursos (por exemplo, uma base de dados) estão disponíveis para realizar tarefas; portanto não é necessário manter conexões síncronas (estáticas) entre requisitantes e provedores de serviços [ANSA95] [Nwan96];
- suporte à operação desconectada (*"fire and forget"*): computadores móveis e dispositivos leves (tais como Assistentes Digitais Pessoais) normalmente se conectam de forma intermitente a uma rede. Aplicações clientes em execução naqueles equipamentos podem iniciar um agente contendo uma requisição de um serviço e despachar o agente durante uma rápida sessão de conexão. A resposta pode vir numa conexão realizada num momento posterior [ANSA95] [Harr95] [Nwan96];
- suporte a aplicações clientes baseadas em máquinas com recursos computacionais restritos: agentes executam tarefas em nós para onde migram, diminuindo a carga de processamento nos nós em que foram criados [ANSA95] [Harr95] [Nwan96].

O artigo: "*Mobile Agents: Are they a good idea?*" [Harr95] reforça o argumento de que SBAM se constituem numa abordagem singular para o desenvolvimento de sistemas distribuídos. Os autores afirmam que correntemente não há outra alternativa que apresente o mesmo conjunto de funcionalidades e vantagens que a abordagem baseada em agentes móveis.

Apesar de ser uma área de recente concentração de esforços de pesquisa e desenvolvimento, sistemas baseados em agentes móveis têm atraído o interesse e o investimento de um conjunto expressivo e crescente de universidades, centros de pesquisa e empresas. Entretanto há um número muito reduzido de experiências de utilização desta abordagem apresentados na literatura.

Este artigo relata a experiência de desenvolvimento e de utilização de um SBAM para monitoramento de qualidade de serviço (QoS) em sistemas multimídia distribuídos (SMD). A sua organização é descrita na seqüência. A próxima seção apresenta algumas das infra-estruturas existentes para suporte a sistemas baseados em agentes móveis. O aspecto de QoS em sistemas multimídia distribuídos é abordado na seção 3, concentrando-se na etapa de monitoramento de QoS como estudo de caso para SBAM. A seção 4 descreve o projeto e a implementação de um SBAM para monitoramento de QoS em SMD. Os resultados obtidos a partir do emprego do SBAM desenvolvido em casos de teste são apresentados na seção 5. Considerações e conclusões finais encerram este artigo.

2. Infra-estruturas para Suporte a Sistemas Baseados em Agentes Móveis

Uma infra-estrutura para suporte a SBAM é um sistema de software distribuído numa rede de computadores (possivelmente heterogênea), que permite a criação, a execução, a migração e o término de agentes, bem como a comunicação entre agentes, agentes e outros programas e agentes e seres humanos.

Esforços de desenvolvimento de infra-estruturas para suporte a SBAM emergiram a partir de 1994. A tecnologia Telescript [Gene96], baseada na linguagem de mesmo nome, foi concebida e desenvolvida do princípio, enquanto outras infra-estruturas foram construídas baseando-se em linguagens de programação existentes e seus respectivos ambientes de execução (tais como: Perl [Perl97] e Tcl/Tk [SunS97]).

A partir de meados de 1996, a evolução na tecnologia Java [Java97] causou um grande impacto na área de sistemas distribuídos e consequentemente em SBAM. A linguagem Java é orientada a objetos, interpretada, portável e contém capacidades de *"multithreading"*. Seu ambiente de execução é baseado numa arquitetura de máquina virtual, característica fundamental para a execução de agentes móveis pois implica em código independente de plataforma.

A tecnologia Java engloba facilidades amplas para suporte a segurança e distribuição. Com respeito a este último aspecto, a versão 1.1 introduziu capacidades de extrema relevância: a invocação de método remoto (*"Remote Method Invocation"* – RMI) e a seriação de objetos (*"Object Serialization"* – OS). A primeira consiste em um mecanismo de interação entre objetos, onde um objeto pode invocar métodos de um outro objeto remoto de forma síncrona. A segunda diz respeito à codificação e descodificação de classes para transporte através da rede. Estas capacidades podem ser empregadas na construção de infra-estruturas para suporte a SBAM, facilitando a implementação de serviços como migração de agentes e comunicação remota.

As outras linguagens citadas anteriormente (Telescript, Perl e Tcl/Tk) e seus respectivos ambientes de execução não reúnem um conjunto expressivo de características e vantagens tal qual a tecnologia Java. Por isso deixaram de ser foco de concentração de esforços de desenvolvimento de infra-estruturas para suporte a SBAM.

A tabela 1 apresenta um conjunto de implementações existentes de infra-estruturas para suporte a sistemas baseados em agentes móveis, sua origem e linguagens suportadas.

Infra-estrutura	Origem	Linguagem
Tabriz	Indústria	Telescript
Odyssey	Indústria	Java
Agent Tcl	Academia	Tcl/Tk
Aglets Workbench	Indústria	Java
Voyager	Indústria	Java
TACOMA	Academia	C e Tcl/Tk
Mole	Academia	Java
Ara	Academia	Tcl/Tk e C/C++

Tabela 1: Implementações existentes de infra-estruturas para suporte a SBAM

Dentre as infra-estruturas listadas, TACOMA [Joha95], Mole [Mole97] e Ara [Ara97] são esforços de pesquisa em contexto acadêmico que necessitam de maior amadurecimento e exposição externa. Tendo em vista o quadro apresentado até aqui, as implementações correntemente existentes de infra-estruturas para suporte a SBAM de maior relevância são: Tabriz [Gene96] e Odyssey [Gene96], Agent Tcl [Gray95], Aglets Workbench [IBM96] e Voyager [Obje97].

A infra-estrutura Tabriz para suporte a sistemas baseados em agentes móveis foi apresentada pela General Magic no ano de 1994 e se tornou a primeira do gênero disponível comercialmente. Sua arquitetura é baseada numa linguagem de programação chamada Telescript. Esta infra-estrutura é robusta e abrangente. Entretanto é uma tecnologia proprietária, cara e que requer recursos computacionais significativos. Assim sendo, a General Magic recentemente descontinuou o produto, partindo para o desenvolvimento de uma alternativa baseada na linguagem Java, que foi chamada de Odyssey.

Odyssey implementa parcialmente a funcionalidade e os conceitos da infra-estrutura Tabriz. Sua versão corrente é 1.0 beta 2. Trata-se de um esforço em fase inicial, cujas capacidades não podem ser completamente exploradas pela falta de documentação.

Agent Tcl é uma infra-estrutura em desenvolvimento na Faculdade Dartmouth, construída como extensão sobre o sistema de *"scripts"* Tcl/Tk. Sua implementação correntemente disponível, versão 1.1, ainda é funcionalmente limitada, isto é, apenas pequena parte da arquitetura proposta está operacional. Em breve a versão 1.2 deverá ser lançada contendo avanços nesse sentido.

A infra-estrutura Aglets Workbench (AWB) permite a manipulação de agentes móveis escritos em Java. Ela foi desenvolvida pela IBM no Laboratório de Pesquisa de Tóquio e disponibilizada para uso externo a partir de meados de 1996. Atualmente, encontra-se na versão alfa teste 5. Aglets Workbench é uma das infra-estruturas para suporte a SBAM mais utilizadas correntemente. Tal fato é conseqüência de uma série de fatores, tais como: um modelo abrangente e bem projetado, facilidade de operação, boa documentação, grande exposição a contextos externos ao seu ambiente de desenvolvimento e utilização da tecnologia Java.

Voyager é uma plataforma baseada na tecnologia Java cujo principal componente é um intermediário de requisições de objetos (*"Object Request Broker"* – ORB) que provê suporte a objetos e agentes móveis. Além do ORB, serviços de persistência, comunicação em grupo e diretório de nomes fazem parte desta plataforma. A versão corrente é a 1.0, lançada como produto em setembro de 1997 pela empresa ObjectSpace.

3. Qualidade de Serviço em Sistemas Multimídia Distribuídos: um Estudo de Caso para Sistemas Baseados em Agentes Móveis

Sistemas multimídia são sistemas computacionais que manipulam de forma integrada vários tipos de meios de representação de informação. Tais meios podem ser estáticos (por exemplo: texto e gráficos) ou dinâmicos (por exemplo: áudio, vídeo e animação). Freqüentemente, sistemas multimídia são distribuídos, isto é, seus componentes estão localizados em diferentes nós de processamento numa rede local ou de longa distância. Sistemas multimídia distribuídos que manipulam meios dinâmicos (também chamados de meios contínuos) alcançaram grande popularidade recentemente. Exemplos de SMD incluem sistemas de conferência e de ensino à distância.

Meios contínuos são tratados por SMD como uma seqüência de segmentos, onde cada segmento consiste de dados obtidos de um dispositivo de captura amostrados em uma taxa constante. Uma seqüência de segmentos é denominada *fluxo*. Segmentos normalmente demandam algum processamento antes de serem armazenados ou transmitidos, por exemplo: codificação, filtragem e compressão.

Qualidade de serviço neste contexto pode ser definida intuitivamente como uma medida de quão satisfeito está o usuário com respeito a um serviço prestado por um SMD. Embora a noção de qualidade de serviço seja intuitiva, uma série de parâmetros mensuráveis pode ser estabelecida para se definir tal conceito objetivamente. Estes parâmetros se dividem em dois níveis: usuário e sistema.

Em geral, do ponto de vista do usuário de um SMD, QoS pode ser definida em termos dos seguintes parâmetros (parâmetros de QoS em nível de usuário):

- resolução: estabelece a precisão do processo de digitalização de um segmento de um meio. É descrita como função de bits por segmento e taxa de amostragem;
- distorção: mede a perda de informação por segmento de um meio;
- nível de sincronização: mede a estabilidade de exibição de segmentos de um meio e de segmentos de meios diferentes porém relacionados.

Do ponto de vista de sistema computacional, Vogel [Voge95] define os seguintes parâmetros de QoS (parâmetros de QoS em nível de sistema):

- atraso fim a fim: o tempo transcorrido entre a captura (ou acesso a base de dados para a obtenção) de um segmento de um meio e a sua exibição;
- jitter: variação do atraso fim a fim entre duas exibições sucessivas de segmentos de um meio;
- taxa de erro de pacotes (*"packet error rate"* – PER): percentual de pacotes descartados devido a falhas de transmissão;
- taxa de erro de bits (*"bit error rate"* – BER): percentual de bits afetados por falhas de transmissão.

Quanto menor forem os valores dos parâmetros no nível de sistema apresentados anteriormente, maior será a qualidade de serviço. Apesar de haver uma divisão em dois níveis, é importante ressaltar que parâmetros em nível de sistema estão relacionados aos em nível de usuário, e vice-versa. Em geral, o jitter tem impacto no nível de sincronização e PER e BER têm impacto sobre a distorção.

A qualidade de serviço em SMD é função dos recursos alocados para a manipulação de um ou mais meios de representação de informação associado(s) ao serviço. Tais recursos englobam: dispositivos de captura e exibição, unidade central de processamento, memória, infra-estrutura de rede e software (sistemas operacionais, sistemas para processamento de meios, infra-estrutura para computação distribuída e sistemas de informação).

Sistemas multimídia distribuídos devem empregar políticas efetivas de gerência de recursos para prover suporte à qualidade de serviço. As políticas dirigem a maneira como recursos são alocados, monitorados, otimizados e liberados. Visto que componentes de um SMD são normalmente executados em diferentes domínios administrativos, a gerência de recursos deve se basear em mecanismos de negociação. O processo de negociação não é trivial pois pode envolver conflito, isto é, disputa pelos recursos desejados.

O suporte à qualidade de serviço deve ser provido durante todo o ciclo de vida de um sistema multimídia distribuído. Segundo Hafid e Bochmann [Hafi95], tal ciclo pode ser dividido em três fases:

- estabelecimento: onde recursos são alocados para um serviço;
- gerência: onde recursos são utilizados e gerenciados;
- encerramento: onde recursos são liberados.

A fase de estabelecimento é composta por quatro etapas: especificação de QoS no nível de usuário, mapeamento de parâmetros de QoS no nível de usuário para parâmetros no nível de sistema, negociação de QoS requerida e reserva de recursos. Na primeira etapa, normalmente são especificados os valores desejados juntamente com margens de tolerância.

A fase de gerência contém três etapas:

- monitoramento de QoS: a qualidade de serviço corrente é medida através da obtenção de valores de parâmetros no nível de sistema. A partir dos valores medidos, verifica-se se o SMD está honrando a QoS estabelecida na fase anterior;

- adaptação de QoS ocorre quando a qualidade de serviço alcançada não é a desejada, porém está dentro de limites de tolerância;

- renegociação de QoS ocorre quando há a violação de valores mínimos para parâmetros de qualidade de serviço ou quando o usuário solicita a renegociação. Ela é similar à negociação exceto pelo fato de que o SMD continua sua execução enquanto a renegociação é conduzida.

Propostas iniciais de suporte a QoS em sistemas multimídia distribuídos se caracterizaram por seguirem arquiteturas centralizadas [Camp93]. Mais recentemente, abordagens baseadas no modelo Cliente/Servidor [Kerh94] e em intermediação (*"brokerage"*) [Nahr95] foram desenvolvidas. Na primeira, aplicações cliente interagem com servidores (gerentes de recursos) para negociação e monitoramento de recursos. Na abordagem de intermediação, um protocolo de negociação permite a interação entre gerentes de recursos (intermediários) localizados em diferentes nós de uma rede. Um intermediário age como consumidor, quando obtendo recursos, ou como fornecedor, quando oferecendo recursos disponíveis.

Aurrecoechea et al. [Aurr95] apresentam uma revisão ampla sobre as arquiteturas existentes de suporte a QoS em SMD mais promissoras. Guedes et al. [Gued97] afirmam que nenhuma delas é suficientemente abrangente e propõem em seu trabalho um modelo baseado em agentes para a negociação e gerência de qualidade de serviço em sistemas multimídia distribuídos. Este modelo segue o ciclo de vida de Hafid e Bochmann [Hafi95] e explora as potencialidades de SBAM como paradigma de desenvolvimento de sistemas distribuídos.

O modelo proposto por Guedes et al. [Gued97] é bastante amplo. O trabalho apresentado neste artigo se restringe à porção do modelo que trata da etapa de monitoramento de QoS. Esta etapa faz parte da fase de gerência do ciclo de vida de SMD e foi selecionada como estudo de caso para o emprego de sistemas baseados em agentes móveis.

A arquitetura do modelo proposto por Guedes et al. [Gued97] contém uma agência de qualidade de serviço em cada nó de uma rede onde um dado SMD é executado. Cada agência de QoS é composta por contratos, um servidor de contratos, uma fábrica de agentes, e agentes que podem ser estáticos ou móveis (figura 2).

Um contrato descreve os fluxos que compõem um SMD, bem como os parâmetros de QoS associados aos fluxos. Fluxos são descritos através do tipo do meio associado, o seu nó de origem e os seus nós de destino. Parâmetros são especificados como um conjunto de atributos e valores associados. Um atributo descreve um parâmetro. Os valores descrevem a qualidade de serviço desejada bem como faixas de tolerância. O servidor de contratos gerencia contratos armazenados em uma base.

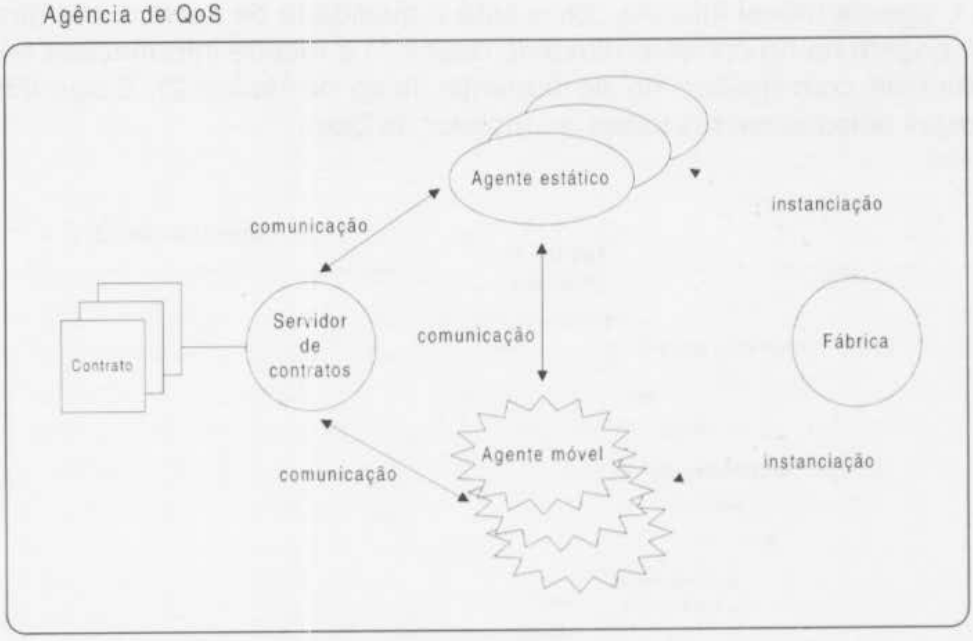

Figura 2: Componentes de uma agência de QoS [Gued97]

Uma fábrica cria e termina agentes estáticos e móveis. O modelo define sete tipos de agentes estáticos e três tipos de agentes móveis, associados à negociação e gerência de QoS. Na seqüência serão apenas listados os agentes participantes da etapa de monitoramento. Os agentes estáticos são:

- agente de interface: interage com o usuário para informar os valores de QoS obtidos durante a fase de gerência;
- mapeador de QoS: mapea valores de parâmetros de QoS no nível de sistema para o nível de usuário;
- monitor de QoS: tem como função medir a qualidade dos fluxos com destino no nó em que reside. Além disso, a partir de valores de parâmetros de QoS referentes a fluxos com origem no nó em que reside, informa outros agentes para que ações cabíveis (por exemplo, adaptação e renegociação) possam ser tomadas;

O agente móvel monitor de contrato tem como responsabilidade realizar a etapa de monitoramento. Visando cumprir este objetivo tal agente migra para as agências envolvidas em um contrato, inicialmente obtendo valores de parâmetros de QoS para fluxos com destino no nó corrente. Também tem como atribuição relatar para a origem estes valores. A tarefa do monitor de contrato tem escopo global, isto é, a sua realização depende de ações tomadas em várias agências de QoS.

Um cenário de monitoramento de QoS, adaptado de [Gued97] é ilustrado pela figura 3. Esta figura mostra uma agência de QoS, onde agentes fixos são representados por elipses e o agente móvel é denotado por uma estrela. A entidade externa que agrega sensores locais é representada por um retângulo com bordas arredondadas. Fluxos de dados são denotados por setas.

Neste cenário, um agente monitor de contrato migra para as agências de QoS localizadas nos nós associados aos fluxos do contrato sob monitoramento. Ao chegar numa agência, o monitor de contrato interage com o monitor de QoS local. Nesta

interação, o agente móvel informa como está a qualidade de serviço nos destinos dos fluxos com origem no nó corrente (fluxo de dados 1) e recebe informações de QoS dos fluxos multimídia com destino no nó corrente (fluxo de dados 2). Estas informações são fornecidas pelos sensores locais ao monitor de QoS.

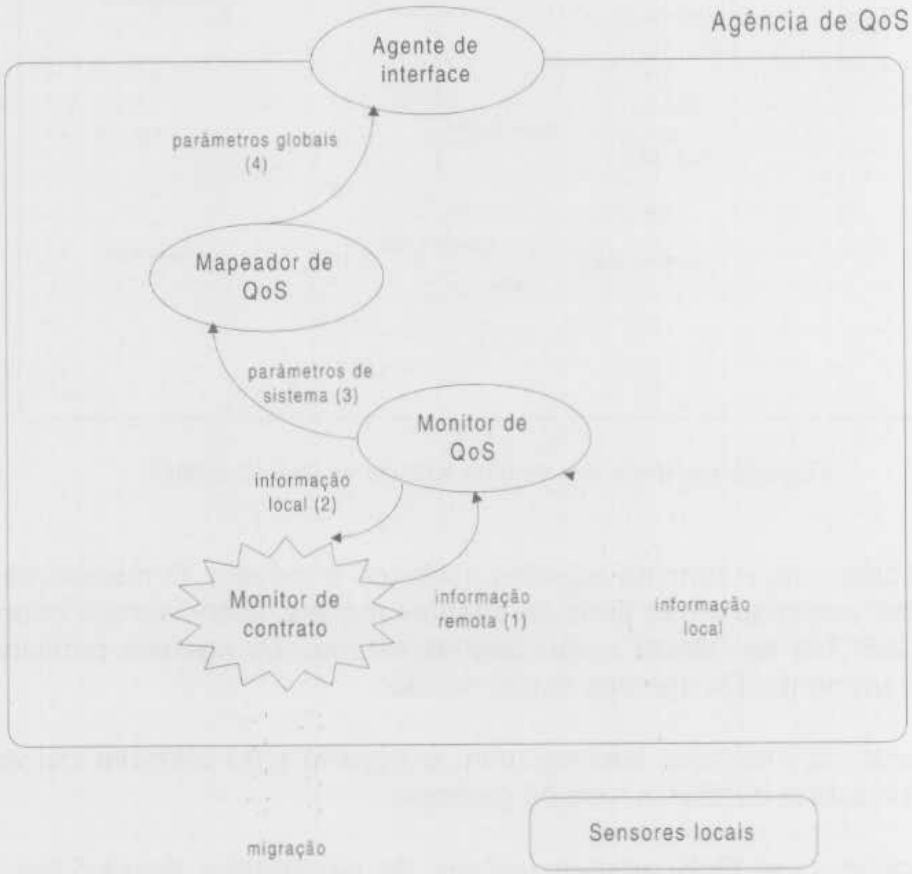

Figura 3: Cenário de monitoramento de QoS

A partir da interação com o monitor de contrato, o monitor de QoS passa as informações de qualidade de serviço dos fluxos com origem local para o mapeador de QoS (fluxo de dados 3). Este, por sua vez, calcula os valores de parâmetros no nível de usuário e os envia para o agente de interface (fluxo de dados 4), que os mostra ao usuário. Caso a qualidade de serviço não seja a desejada as etapas de adaptação ou renegociação podem ser iniciadas por outros agentes ou pelo usuário.

A próxima seção deste artigo discorre sobre o projeto e a implementação de um SBAM para monitoramento de QoS em SMD, que foi desenvolvido a partir do modelo proposto por Guedes et al. [Gued97].

4. Projeto e Implementação de um SBAM para Monitoramento de Qualidade de Serviço em Sistemas Multimídia Distribuídos

Um SBAM para monitoramento de QoS em SMD é um sistema de software que deve atender um conjunto de requisitos correntemente desejados para sistemas distribuídos, conforme apresentado em [Oliv97]. Dentre tais requisitos pode-se citar: independência de plataforma, facilidade de integração com outros sistemas e flexibilidade. Para atender estes requisitos, a arquitetura de desenvolvimento se baseia em dois elementos principais:

- infra-estrutura para suporte a sistemas baseados em agentes móveis;
- infra-estrutura de computação distribuída aberta: possibilita a interação entre componentes de um sistema distribuído independentemente de hardware, sistema operacional, protocolo de rede e linguagem de programação. Esta capacidade é chamada de interoperabilidade.

A infra-estrutura para suporte a SBAM é o núcleo da arquitetura de desenvolvimento. Os agentes componentes do modelo proposto por Guedes et al. [Gued97] são manipulados dentro de seu contexto. Para que a etapa de monitoramento de QoS possa ser realizada, agentes têm que se comunicar com programas externos à infra-estrutura para suporte a SBAM. Estes programas são o servidor de contratos e sensores locais. A comunicação de forma interoperável entre agentes e outros programas é provida pela infra-estrutura de computação distribuída aberta.

Aglets Workbench (AWB) da IBM, apresentada na seção 2 deste artigo, foi escolhida como infra-estrutura para suporte a sistemas baseados em agentes móveis. Tal escolha se fundamentou em uma série de fatores que incluem: boa documentação, facilidade de operação, abrangência, robustez, evoluções constantes e grande exposição a usuários externos. Além disso, AWB é baseada na tecnologia Java, que atualmente é a mais promissora para o desenvolvimento de sistemas baseados em agentes móveis. Um *aglet*, terminologia AWB para um agente móvel, é definido como um objeto capaz de se mover autonomamente de um nó para outro numa rede.

A infra-estrutura de computação distribuída aberta selecionada baseou-se na especificação CORBA (*"Common Object Request Broker Architecture"*) do Grupo de Gerência de Objetos [OMG97]. Trata-se de um padrão de interoperabilidade de sistemas amplamente aceito que segue a abordagem de orientação a objetos. O núcleo da especificação CORBA é o intermediário de requisições de objetos (ORB). O ORB permite que aplicações cliente requisitem serviços de objetos servidores de forma transparente num ambiente distribuído. Para tanto, cliente e servidor devem compartilhar de uma definição de interface escrita em IDL (*"Interface Definition Language"*).O produto utilizado no desenvolvimento do SBAM foi o OrbixWeb da Iona Technologies [Iona97]. Trata-se de uma implementação de um ORB com suporte ao ambiente de execução da tecnologia Java.

OrbixWeb e Aglets Workbench são baseados na tecnologia Java. A integração destas infra-estruturas para o desenvolvimento do SBAM se dá no nível de linguagem de programação, isto é, as capacidades de interoperabilidade providas pelo padrão CORBA são diretamente incorporadas no código de agentes aglets. Para tanto, OrbixWeb gera código Java interoperável seguindo a especificação CORBA de mapeamento de interfaces escritas em IDL para linguagem Java.

Da mesma forma, programas externos (servidores) que atendem as requisições de agentes podem se tornar interoperáveis se incorporarem código gerado a partir de mapeamentos IDL especificados pelo padrão CORBA (atualmente disponíveis para linguagens Java, C, C++ e Smalltalk). Agentes desempenham o papel de clientes ORB que fazem requisições para servidores locais. Um exemplo de servidor é um sensor local tal como apresentado na figura 3.

A agência de QoS é a base do modelo proposto por Guedes et al. [Gued97] e consequentemente a parte fundamental do SBAM para monitoramento de qualidade

de serviço em sistemas multimídia distribuídos. A seguir são descritos detalhes de como foram implementados os componentes da agência de QoS.

Contratos e fluxos foram implementados como objetos Java seriados, isto é, seu código interpretável e dados de instâncias podem ser migrados com agentes móveis quando necessário. Fluxos são identificados pelo tipo do meio manipulado (áudio, vídeo, etc.) e pelo nó de sua origem. Uma lista de nós destino e um conjunto de parâmetros de QoS em nível de sistema completam a estrutura de um fluxo. Cada parâmetro de QoS associado a um fluxo é descrito por um nome, valor desejado e limites de tolerância. Contratos são identificados por seu nome e versão. A estrutura de um contrato é completada por uma lista de fluxos componentes.

O servidor de contratos é um sistema responsável por gerenciar contratos armazenados em uma base. Tal sistema foi implementado na linguagem Java e algumas de suas ações foram disponibilizadas de forma interoperável. A base gerenciada pelo servidor de contratos é um arquivo seqüencial de objetos contrato. A inserção de contratos na base é realizada no modelo proposto por Guedes et al. [Gued97] por agentes de negociação e renegociação. Uma aplicação de cadastro de contratos (escrita em Java) foi desenvolvida para realizar esta tarefa, visto que tais agentes estão ainda em fase de desenvolvimento e não fazem parte do escopo de monitoramento de QoS.

A fábrica de agentes tem como função criar e terminar agentes. Este componente da agência de QoS foi implementado em Java como um agente fixo aglet presente em cada agência componente do SMD sob monitoramento. A razão desta decisão se deve a características particulares da infra-estrutura Aglets Workbench que oferece facilidades expressivas de comunicação (local e remota) entre agentes.

Com respeito aos quatro tipos de agentes que estão envolvidos na realização da etapa de monitoramento, apenas o mapeador de QoS não foi implementado. Isto porque não há ainda na área de sistemas multimídia um estudo abrangente e detalhado estabelecendo parâmetros subjetivos no nível de usuário, bem como seus respectivos mapeamentos para o nível de sistema. Assim sendo, os parâmetros no nível de sistema são diretamente apresentados ao usuário.

Agentes foram implementados em Java como aglets. Os agentes monitor de QoS e de interface foram construídos como aglets estáticos. O monitor de QoS interage com sensor locais de forma interoperável através do OrbixWeb. Uma interface IDL foi definida para isso. Agentes de interface foram implementados para cada origem de fluxo, utilizando-se das capacidades gráficas do AWT (*"Abstract Windowing Toolkit"*) Java.

O monitor de contrato foi desenvolvido de forma ligeiramente diferente do modelo proposto por Guedes et al. [Gued97]. Neste modelo há apenas um agente para monitorar todo o contrato. Como a abordagem de SBAM enfatiza paralelismo, decidiu-se que a etapa de monitoramento seria realizada por vários agentes monitores de contrato, sendo que cada um é responsável por monitorar fluxos com mesma origem (nó). Os valores dos parâmetros de QoS obtidos pelo monitor de contrato são enviados para a agência localizada na origem dos fluxos, via mensagens remotas. Desta forma o monitoramento é realizado por um conjunto de agentes móveis autônomos e cooperantes.

Para cada agente monitor de contrato é necessário estabelecer um roteiro (itinerário) de viagem. O itinerário é uma seqüência de nós que são destinos de fluxos

com mesmo nó de origem. O agente é criado pela fábrica no nó origem dos fluxos que irá monitorar. Após a sua criação, o agente monitor de contrato inicia viagem (migrando para o primeiro nó componente do seu itinerário) visando realizar a etapa de monitoramento. Em seguida, as migrações têm como destino nós subseqüentes do seu itinerário. Quando todo o itinerário é percorrido, o agente completa um ciclo de monitoramento de QoS (todos os destinos de fluxos foram visitados), e retorna ao primeiro nó componente do seu itinerário para iniciar mais um ciclo.

A figura 4 ilustra um cenário de monitoramento de QoS, contendo um nó origem e um nó destino para um fluxo multimídia. Os componentes da agência de QoS e suas relações com a arquitetura de desenvolvimento podem ser nela observados.

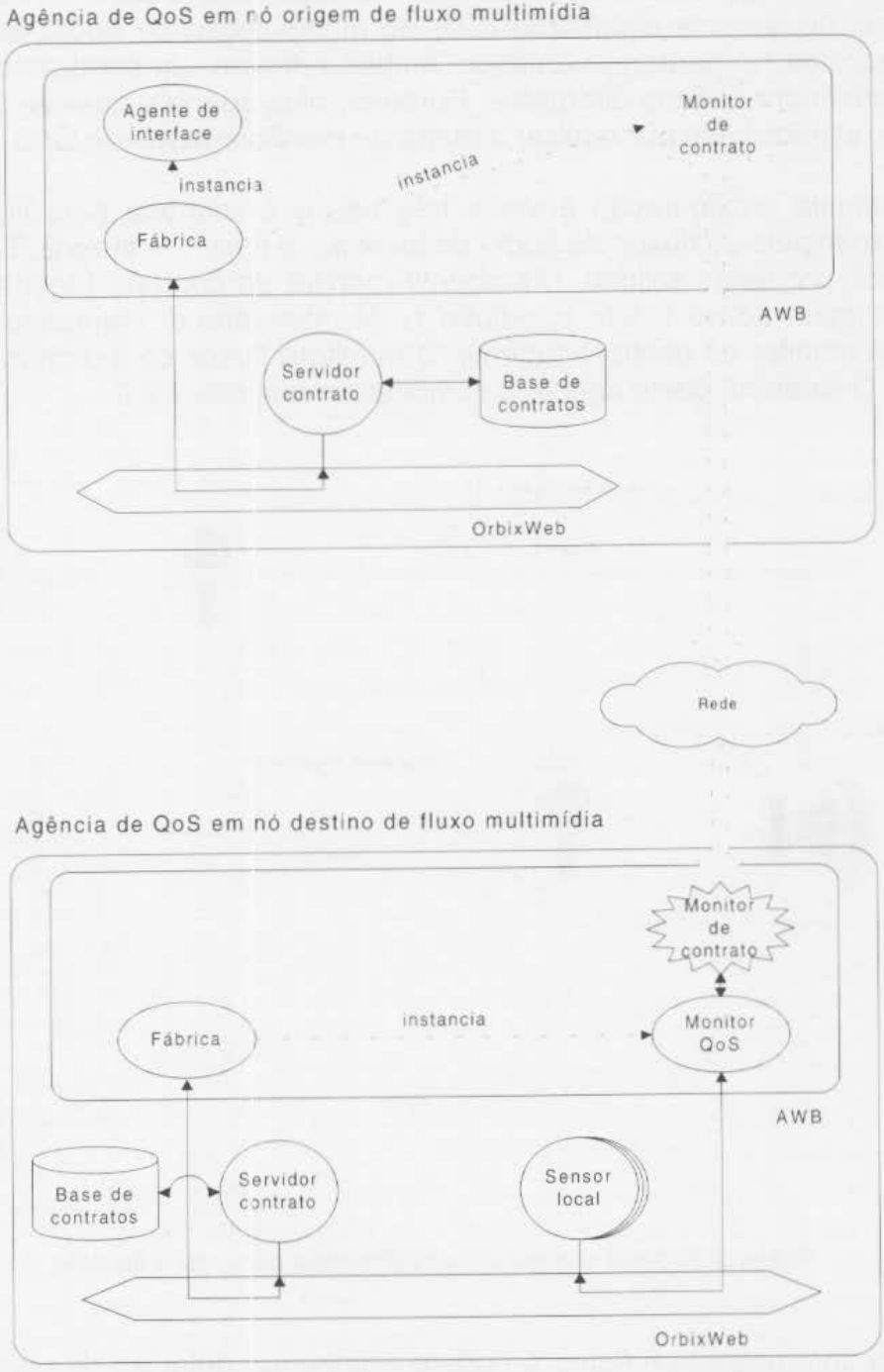

Figura 4: Cenário de monitoramento de QoS através do SBAM desenvolvido

5. Casos de teste

O sistema baseado em agentes móveis desenvolvido conforme descrito na seção anterior teve seu comportamento examinado através de casos de teste. Os casos de teste foram executados num ambiente distribuído composto por quatro estações de trabalho Sun sob sistema operacional Solaris, ligadas em rede local Ethernet. As estações dispõem de dispositivos multimídia incluindo alto-falante, microfone e câmera de vídeo.

Um sistema de manipulação de áudio em ambiente distribuído, implementado por Araújo et al. [Arau97], foi utilizado como SMD a ser monitorado. Embora atualmente manipule apenas um meio, tal sistema é suficiente para se cumprir os propósitos deste artigo. Casos de teste foram estabelecidos a partir de duas diferentes configurações do sistema distribuído (SD) de manipulação de áudio, envolvendo os recursos descritos no parágrafo anterior. Ambas refletem um contrato contendo dois fluxos de áudio com origens diferentes. Portanto, dois agentes móveis, um para cada origem, são responsáveis por realizar a etapa de monitoramento de QoS.

A primeira configuração envolve três nós e é ilustrada pela figura 5. Nesta figura o SD manipula os fluxos de áudio denominados fluxo 1 e fluxo 2. Tais fluxos são representados por setas sólidas. Um agente monitor de contrato (agente 1) monitora fluxos com origem no nó 1, isto é, o fluxo 1. Seu itinerário é composto pelo nó 2. O outro agente monitor de contrato (agente 2) monitora fluxos com origem no nó 2, isto é, o fluxo 2. O itinerário deste agente é composto pelos nós 1 e 3.

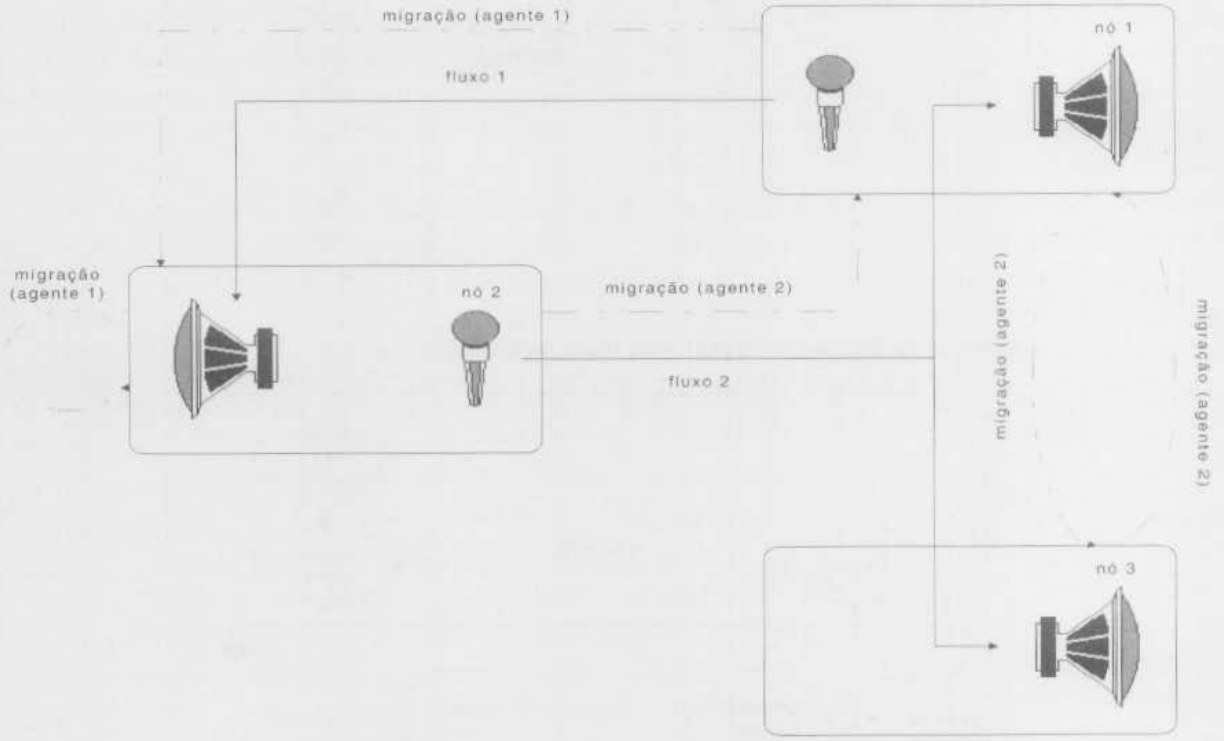

Figura 5: Primeira configuração estabelecida para casos de teste

Ainda com respeito à figura 5 podemos observar detalhes de como o itinerário dos agentes é percorrido. Estes detalhes estão ilustrados através de setas não sólidas com padrões diferenciados para cada agente. Com respeito ao agente 1, inicialmente há uma migração do nó origem do fluxo 1 (nó 1) para o único destino de seu itinerário,

o nó 2. A partir de então, ele segue realizando a tarefa de monitoramento do fluxo 1, "migrando" sempre do nó 2 para o nó 2. Para o agente 2, inicialmente há uma migração do nó origem do fluxo 2 (nó 2) para o primeiro destino de seu itinerário, o nó 1. A partir de então o agente começa a realizar ciclos de monitoramento migrando do nó 1 para o nó 3 e vice-versa.

A segunda configuração estabelecida para casos de teste envolve quatro nós e é semelhante à primeira. A única diferença é que um outro nó, nó 4, é adicionado como destino do fluxo 2. Neste caso o agente monitor de contrato para o fluxo 2 possui itinerário composto pelos nós 1, 3 e 4.

O SD de manipulação de áudio utiliza o Protocolo de Tempo Real (RTP) [Schu96] para a transmissão de fluxos multimídia. RTP transmite uma quantidade muito pequena de informação de controle, sendo sua execução leve e rápida (tornando-o de grande aplicabilidade em SMD). Além disso, RTP calcula e torna disponível valores de parâmetros de QoS em nível de sistema, tais como: taxa de transferência, jitter e taxa de perda de pacotes.

Os sensores locais presentes em cada nó envolvido no sistema distribuído de manipulação de áudio foram implementados em C++. Através da definição de uma interface IDL (seguindo-se a especificação CORBA), tais objetos se tornaram servidores interoperáveis capazes de fornecer valores de parâmetros de QoS em nível de sistema para qualquer outro programa. Os sensores locais obtêm os valores de QoS através da interação com o RTP.

A seguir são apresentados dados obtidos como conseqüência da realização de três casos de teste que utilizam o SD de manipulação de áudio. Para cada caso de teste foram levantados valores referentes a dois indicadores de tempo visando a análise de desempenho do SBAM desenvolvido. Tais indicadores são: tempo de migração e tempo de resposta.

O tempo de migração é o intervalo de tempo em segundos transcorrido na migração de um agente monitor de contrato entre dois nós consecutivos do seu itinerário. O tempo de resposta é o intervalo de tempo em segundos transcorrido entre dois recebimentos consecutivos de valores de parâmetros de QoS com respeito a um destino específico de um fluxo multimídia, pela agência localizada em sua origem. O agente monitor de contrato é responsável por enviar tais valores obtidos em agências remotas para a agência de origem.

O primeiro caso de teste se baseia na primeira configuração (ilustrada pela figura 5). O segundo e o terceiro casos de teste se baseiam na segunda configuração. A tabela 2 mostra um quadro geral dos indicadores de tempo para os casos de teste do SBAM desenvolvido.

Configuração / Tempo	1 (3 nós)	2 (4 nós)
Migração médio (segundos)	0,753	0,832
Resposta médio (segundos)	1,867	1,886

Tabela 2: Quadro geral dos indicadores de tempo para os casos de teste do SBAM desenvolvido

A análise de desempenho do SBAM desenvolvido, a partir dos valores levantados para os indicadores de tempo, deve ser feita de forma comparativa para que conclusões relevantes sejam tecidas. Uma outra série de testes foi então estabelecida, utilizando o sistema distribuído de manipulação de áudio e as mesmas configurações de caso de teste que o SBAM desenvolvido foi submetido. Entretanto, uma abordagem diferente de desenvolvimento de sistemas foi empregada: objetos distribuídos interoperáveis através do padrão CORBA.

Conforme descrito anteriormente, sensores locais presentes no SD de manipulação de áudio foram implementados como objetos (servidores) interoperáveis CORBA. Isto torna possível a qualquer aplicação local ou remota a obtenção de valores de parâmetros de QoS disponibilizados pelo servidor, desde que utilizando uma interface IDL comum. Desta forma, uma aplicação cliente foi desenvolvida em Java para coleta de valores de parâmetros de QoS, simulando a tarefa realizada pelo agente monitor de contrato. A aplicação cliente foi posicionada em nós origem de fluxos. A coleta é feita através do envio de requisições remotas (via ORB) para sensores presentes em nós destinos de fluxos de um contrato específico.

Apenas o indicador tempo de resposta foi estabelecido para análise de desempenho da abordagem CORBA, já que não faz sentido o indicador tempo de migração. Este indicador é definido como o intervalo de tempo em segundos transcorrido entre uma requisição de valores de parâmetros de QoS para um dado nó destino de um fluxo específico, e a sua resposta.

O tempo de resposta médio dos casos de teste da abordagem CORBA (t_{CORBA}) é de 0,0206 segundos. O tempo de migração médio dos casos de teste do SBAM (t_{mig}) é 0,805 s, o que eqüivale a 39,07 vezes t_{CORBA}. Os valores de t_{mig} variaram no intervalo de 24,85 a 143,49 vezes t_{CORBA}.

O tempo de resposta médio dos casos de teste do SBAM (t_{resp}) é 1,879 s, o que eqüivale a 91,21 vezes t_{CORBA}. Os valores de t_{resp} variaram no intervalo de 63,39 a 173,54 vezes t_{CORBA}. O tempo de migração médio corresponde a 42,8% do tempo de resposta médio.

Embora os tempos médios de migração e de resposta para os casos de teste do SBAM desenvolvido sejam bem elevados (quando comparados com o tempo de resposta médio dos casos de teste da abordagem CORBA), a utilização da abordagem baseada em agentes móveis não se torna inviável.

A abordagem de SBAM é emergente e a infra-estrutura Aglets Workbench está em fase de amadurecimento. Correntemente tal infra-estrutura se encontra em versão alfa teste. A IBM reconheceu recentemente, na lista de discussão eletrônica da AWB, que o seu protocolo para migração e comunicação de agentes (ATP) ainda é bastante ineficiente. Melhorias significativas em seu desempenho são esperadas para próximas versões.

A abordagem CORBA, por outro lado, vem sendo objeto de esforços de pesquisa e desenvolvimento desde 1989 quando foi estabelecido o OMG. A família de produtos Orbix, da qual OrbixWeb faz parte, teve sua primeira versão comercial lançada em meados de 1993 estando, portanto, bem madura.

A abordagem de SBAM deve continuar a ser investigada visto que agentes móveis podem carregar conhecimento consigo visando a realização de tarefas de forma eficiente. Em interações entre componentes no modelo Cliente/Servidor, como é

o caso da abordagem CORBA, apenas dados são migrados, cabendo normalmente ao cliente tratar estes dados após todo o seu volume ter sido transportado de um nó remoto.

A abordagem CORBA, para o estudo de caso aplicado, é bem mais rápida que o SBAM desenvolvido porque o volume de dados transferido é pequeno e tais dados correspondem a tipos básicos (inteiro e ponto flutuante), não sofrendo nenhum tipo de tratamento.

Para a abordagem baseada em agentes, entretanto, em média despende-se 42,8% do tempo de resposta para a migração do agente monitor de contrato, que é um processo elaborado onde código e dados são transportados. Além disso, antes da realização da tarefa de monitoramento, o agente é restaurado para execução ao chegar na agência destino de migração. Transcorre-se então mais um intervalo de tempo de preparação que tem impacto no tempo de resposta.

6. Conclusões

A abordagem de sistemas baseados em agentes móveis é potencialmente singular como paradigma para desenvolvimento de sistemas distribuídos. Entretanto, trata-se de uma abordagem emergente e de recente concentração de esforços de pesquisa e desenvolvimento.

Infra-estruturas para suporte a SBAM ainda estão em fase inicial de desenvolvimento e precisam alcançar maior maturidade para que sistemas baseados em agentes móveis possam ser desenvolvidos com robustez e empregados em larga escala. Há uma grande expectativa de que, num futuro muito próximo, uma série de infra-estruturas estarão disponíveis possibilitando assim a popularização de SBAM. Desta forma poder-se-á fazer uma avaliação mais consistente e abrangente desta abordagem, e será possível verificar se suas potencialidades são traduzidas em realidade.

Os indicadores de tempo analisados comparativamente na seção anterior mostram que o requisito de independência de plataforma traz penalidades no que diz respeito a desempenho. Isto porque ambientes de execução de agentes são baseados em interpretação de código. Entretanto, os dados apresentados não inviabilizam a abordagem de SBAM pois apenas uma infra-estrutura (Aglets Workbench) em fase alfa teste foi empregada. Além disso, a IBM reconheceu que o seu protocolo para migração e comunicação de agentes ainda é bastante ineficiente.

Como extensões deste trabalho, duas frentes são relevantes. Primeiramente, ampliar o escopo do SBAM desenvolvido para a etapa de monitoramento, de forma que este sistema contemple todo o modelo proposto por Guedes et al. [Gued97]. Esta ação visa avaliar o comportamento de SBAM compostos por um grande número de agentes, num domínio de aplicação amplo. Em segundo lugar, explorar aspectos de inteligência em agentes móveis de forma a avaliar o impacto de conciliar representação e manipulação de conhecimento com requisitos de mobilidade (código compacto e independência de plataforma).

Referências Bibliográficas

[ANSA95] Advanced Networked Systems Architecture (ANSA)
Scripts and Mobile Agents
ftp://ftp.ansa.co.uk/phase3-doc-root/bn/APM.1593.01.ps.gz
1995

[Ara97] Ara – *"Agents for Remote Action"*
The Ara Platform for Mobile Agents
Grupo de Sistemas Distribuídos
Departamento de Ciência da Computação
Universidade de Kaiserslautern
http://www.uni-kl.de/AG-Nehmer/Ara
1997

[Arau97] Araújo, D. E.; Prado, R. C. M. ; Faina, L. F.; Cardozo, E.
Implementação do Módulo DPE da Arquitetura TINA-C sobre CORBA
2.º Seminário Franco-Brasileiro em Sistemas Informáticos Distribuídos
Arquiteturas Multimídia para as Telecomunicações
Fortaleza, Brasil, 1997

[Aurr95] Aurrecoechea, C.; Campbell, A.; Hauw, L.
A Review of Quality of Service Architectures
ACM Multimedia Systems Journal, Novembro, 1995

[Camp93] Campbell, A.; Coulson, G.; Hutchim, D.
A Multimedia Enhanced Transport Service in a Quality of Service Architecture
Workshop on Network and Operating System Support for Audio and Video'93
Lancaster, Inglaterra, Novembro, 1993

[Gene96] General Magic, Inc.
Telescript Technology: Mobile Agents
http://www.genmagic.com/Telescript/Whitepapers/wp4/whitepaper-4.html
1996

[Gray95] Gray, R. S.
Agent Tcl: A transportable agent system
Proceedings of the CIKM Workshop on Intelligent Information Agents
CIKM 95, Baltimore, Maryland, 1995

[Gued97] Guedes, L. A.; Oliveira, P. C.; Faina, L. F.; Cardozo, E.
QoS Agency: An Agent-Based Architecture for Supporting Quality of Service in Distributed Multimedia Systems
IEEE Conference on Protocols for Multimedia Systems – Multimedia Networking
Santiago, Chile, Novembro, 1997

[Hafi95] Hafid, A.; Bochmann, G.
An Approach to Quality of Service Management for Distributed Multimedia Systems
International Conference on Open Distributed Processing, Australia, 1995, pp. 319-340

[Harr95] Harrison, C. G.; Chess, D. M.; Kershenbaum, A.
Mobile Agents: Are they a good idea?
Relatório de Pesquisa da IBM, Divisão de Pesquisa da IBM, 1995

[IBM96] IBM Corporation
Laboratório de Pesquisa de Tóquio
Programming Mobile Agents in Java – A White Paper
http://www.trl.ibm.co.jp/aglets/whitepaper.htm
Setembro, 1996

[Iona97] Iona Technologies Ltd.
Iona Technologies: Our Products
http://www.iona.com/Products/index.html
1997

[Java97] Sun Microsystems, Inc.
Java Home Page
http://www.javasoft.com
1997

[Joha95] Johansen, D.; Renesse, R.; Schneider, F.B.
An Introduction to the TACOMA Distributed System – Version 1.0
Relatório Técnico 95-23
Departamento de Ciência da Computação
Universidade de TromsØ, Noruega
Junho, 1995

[Kerh94] Kerhervé, B. et al.
On Distributed Multimedia Presentational Systems: Functional and Computational Architecture and QoS Negotiation
Proceedings of the 4 th. International Workshop on Protocols for High-Speed Networks
Chapman & Hall, Londres, 1994, pp. 53-67

[Ling95] Lingnau, A.; Drobnik, O.
An Infrastructure for Mobile Agents: Requirements and Architecture
Fachbereich Informatik (Telematik)
Johann Wolfgang Goethe-Universität
1995

[Mole97] Projeto Mole
Project Mole – Mobile Agents
Grupo de Sistemas Distribuídos
Instituto de Sistemas Paralelos e de Alta Performance (IPVR)
Universidade de Stuttgart
http://www.informatik.uni-stuttgart.de/ipvr/vs/projekte/mole.html
1997

[Nahr95] Nahrstedt, K.; Smith, J.
The QoS Broker
IEEE Multimedia, 1996 primavera, 1995, pp. 53-67

[Nwan96] Nwana, H. S.
Software Agents: An Overview
Knowledge Engineering Review, Vol. 11, No 3, pp. 1-40, Setembro, 1996

[Obje97] ObjectSpace, Inc.
ObjectSpace: Voyager Overview
http://www.objectspace.com/voyager
1997

[Oliv97] Oliveira, P. C.; Cardozo, E.
Mobile Agent-Based Systems: an Alternative Paradigm for Distributed Systems Development
Anais do 15.º Simpósio Brasileiro de Redes de Computadores
São Carlos, Brasil, Maio, 1997

[OMG97] Object Management Group (OMG)
OMG Home Page
http://www.omg.org
1997

[Perl97] Linguagem de Programação Perl
The www.perl.com Home Page
http://www.perl.com
1997

[Schu96] Schulzrinne, H.; Frederik, R; Jacobson, V.
RTP: A Transport Protocol for Real Time Systems
Computer Communications, Janeiro, 1996

[SunS97] Sunscript
SunScript Home Page
http://sunscript.sun.com
1997

[Voge95] Vogel, A. et al.
Distributed Multimedia and QoS: A Survey
IEEE Multimedia, verão, 1995, pp. 10-18