Gerenciamento de Objetos CORBA para a plataforma Multiware

João Augusto G. de Queiroz *Edmundo R. M. Madeira*
IC - Instituto de Computação
UNICAMP - Universidade de Campinas
13081-970 Campinas-SP Brasil
{aqueiroz,edmundo}@dcc.unicamp.br

Resumo

Este artigo apresenta um sistema de monitorização de aplicações distribuídas CORBA, no contexto da arquitetura de gerenciamento integrado da plataforma Multiware e analisa a sua implementação. Esta arquitetura, descrita suscintamente, é baseada na ODMA da ISO, cuja proposta é fundamentada no Modelo de Referência ODP, estendendo o padrão OSI de gerenciamento; e adota como infra-estrutura de suporte a CORBA, os CORBAservices e as emergentes CORBAfacilities, onde os objetos gerenciados são os componentes das aplicações e os elementos de suporte em um ambiente de processamento distribuído aberto. O primeiro passo foi instrumentar os clientes e servidores CORBA com sensores, habilitando a sua monitorização. O protótipo implementa uma Base de Informação de Gerenciamento definida em IDL, acessível por objetos CORBA, com informações de gerenciamento úteis para as áreas funcionais de desempenho e de contabilização, visualizadas em uma interface gráfica em Java. Os resultados observados com aplicações num ambiente heterogêneo, como o da Multiware, levam-nos à direção da necessidade de se adotar o conceito pleno de gerenciamento integrado de sistemas distribuídos.

Abstract

This paper presents a distributed application monitoring system in the context of the integrated system management architecture for the Multiware platform and analyses its implementation. This architecture is based on the ODMA/ISO, which is compliant with the RM-ODP to extent the OSI management standard; and uses the CORBA, the CORBAservices and the emerging CORBAfacilities, where the target objects are the application components and the system support in an ODP environment. The first step in such direction was to instrument the CORBA clients and servers with sensors, enabling the monitoring. The prototype implements a Management Information Base defined in IDL, accessible by CORBA objects, and provides management information for performance and accounting functional areas, by a Java graphical interface. The results observed from applications over a heterogeneous environment, like the Multiware, lead us to the need of a full integrated distributed system management.

Palavras-Chaves: Gerenciamento de Sistemas Distribuídos; Gerenciamento de Aplicações; Monitorização; Instrumentação; Objetos CORBA; e Multiware.

1 Introdução

Um sistema distribuído, como o ambiente da plataforma Multiware desenvolvida na Universidade de Campinas (Loyolla, 1994), consiste tipicamente de um grande número de nós de diferentes arquiteturas, conectados por redes heterogêneas de comunicação, vários sistemas operacionais e muitos serviços de suporte, estabelecendo uma infra-estrutura para o desenvolvimento e a execução de aplicações distribuídas. Estas aplicações consistem de um conjunto de componentes cooperativos de "software" distribuídos. Quando o paradigma da orientação objeto é escolhido para descrever e implementar este ambiente, seus elementos são objetos, com interfaces claramente definidas, cujos métodos podem ser invocados de qualquer nó do domínio estabelecido. Na arquitetura cliente/servidor, tais objetos residem em servidores, processos que oferecem um ou mais serviços, cujas interfaces são transparentemente chamadas por programas clientes. O crescimento e a complexidade dos sistemas distribuídos têm tornado o seu gerenciamento um tema de especial interesse para pesquisadores, projetistas, vendedores e usuários. As abordagens nessa área devem cobrir todos os componentes de tais sistemas com o propósito de entender as interações e correlações. Os processos que implementam uma aplicação distribuída também devem ser considerados como objetos gerenciados.

Em geral, a monitorização é responsável por prover informações de um sistema observado, úteis para o processo de tomada de decisões de gerenciamento, com o propósito de controlar o comportamento de seus elementos. Em sistemas distribuídos, os problemas de tais atividades são ainda mais dificultados pelos aspectos da distribuição. Adicionalmente, o conceito de monitorização é diretamente oposto ao paradigma da encapsulação que protege o estado do objeto e os procedimentos associados da observação externa, escondendo detalhes internos que podem ser de interesse para o gerente. Este artigo descreve um Sistema de Monitorização de aplicações CORBA (Common

Object Request Broker Architecture) para as áreas funcionais de desempenho e contabilização, no contexto da Arquitetura de Gerenciamento Integrado de Sistemas Distribuídos da plataforma Multiware. Embora os objetos CORBA apresentem ambas as características de distribuição e encapsulação, este sistema apresenta uma maneira simples e modular de instrumentá-los e monitorá-los.

A próxima seção apresenta o Modelo de Referência ODP (Open Distributed Processing) (ITU-T/ISO, 1995a) como um modelo de gerenciamento de sistemas distribuídos, por ser uma atividade essencialmente de processamento distribuído. A seção 3 descreve a Arquitetura de Gerenciamento Integrado da Multiware baseada na ODMA (Open Distributed Management Architecture) (ITU-T/ISO, 1995b); na CORBA (OMG, 1995c); nos CORBAservices (Common Object Services Specification) (OMG, 1995b); e nas CORBAfacilities (Common Facilities Architecture) (OMG, 1995a), propostos pelo consórcio OMG (Object Managent Group) e adotados no projeto da Multiware. Na seção 4 o modelo do sistema de monitorização é apresentado, enquanto na seção 5, alguns aspectos de implementação e resultados são mostrados e discutidos. Finalmente, os trabalhos relacionados são descritos na seção 6; e algumas observações e conclusões são apresentadas na última seção.

2 ODP e os Modelos de Gerenciamento de Sistemas Distribuídos

O uso confiável de um sistema distribuído requer um adequado gerenciamento de seus componentes: os recursos de rede, os serviços de suporte e os elementos das aplicações. Em ambientes distribuídos orientados a objetos, a alocação imprópria dos objetos das aplicações pode afetar a qualidade dos seus serviços oferecidos, a carga do sistema da máquina hospedeira e o subsistema de comunicação. Esta situação se torna ainda mais crítica com os serviços de migração, de replicação e de recuperação de falhas. Com este cenário, as especificações estabelecidas para o Gerenciamento Integrado de Sistemas Distribuídos precisam ser satisfeitas (Hegering, 1994). O seu propósito é garantir o comportamento desejado das aplicações distribuídas, monitorando e controlando os elementos de redes, os serviços de suporte e os componentes de tais aplicações, a fim de evitar os riscos e danos associados com o processamento distribuído de uma organização.

Os modelos e padrões de gerenciamento foram desenvolvidos para gerenciar componentes de redes, cujos serviços são especificados, projetados e implementados separadamente de sua funcionalidade normal. O modelo SNMP (Simple Network Management Protocol) (Case, 1990) é usado para monitorar e controlar elementos Internet, como "gateways" e roteadores, através de aplicações em estações de gerenciamento. Embora as suas necessidades de processamento e memória sejam baixas, as funções de "polling" e "trap" não são adequadas para o gerenciamento de aplicações. O modelo de gerenciamento OSI (Open Systems Interconnection) (Yemini, 1993) oferece uma abordagem mais sofisticada, usando conceitos de orientação a objeto como classes, polimorfismo e herança. Embora a linguagem GDMO (Guidelines for Definitions of Managed Objects) corresponda a uma linguagem de definição de interface, como a CORBA IDL (Interface Definition Language), ela não oferece a facilidade de pré-compilação que gera automaticamente os "stubs" para empacotar e desempacotar parâmetros. Os padrões SNMP e OSI não reconhecem que o mesmo modelo usado para especificar, projetar e implementar sistemas distribuídos também pode ser usado para o gerenciamento. Recentemente, o crescimento e sucesso do modelo de objetos CORBA como uma solução de interoperabilidade têm motivado o mapeamento das especificações dos padrões OSI/SNMP para CORBA IDL, através de adaptadores, baseado na possibilidade de CORBA se tornar um padrão de gerenciamento (Ban, 1996; Soukoti, 1997).

Como o gerenciamento de sistemas distribuídos é essencialmente uma atividade distribuída, o RM-ODP deve ser usado para modelá-lo. O Modelo de Referência é um meta-padrão que suporta os conceitos de distribuição, de interoperabilidade, de transparências e de portabilidade de aplicações distribuídas, usando o paradigma da orientação a objetos. Aborda os aspectos de distribuição de uma forma integrada, usando cinco pontos de vistas: de empresa, de informação, computacional, de engenharia e de tecnologia. Portanto, a complexidade do gerenciamento é analisada sob estas óticas, em que cada uma apresenta uma abstração diferente do processo, em uma abordagem simultaneamente "top-down" e "botton-up". Esta é uma maneira distinta de modelar o processo de gerenciamento e mover da dimensão do gerenciamento de redes para o do gerenciamento de sistemas distribuídos integrado. A proposta da ODMA é baseada no RM-ODP e estende o modelo OSI de gerenciamento.

O ponto de vista de empresa considera o sistema e o seu ambiente, focando nos seus propósitos, escopos e políticas. O ponto de vista de informação se preocupa com a informação de gerenciamento que necessita ser armazenada, processada e trocada entre os componentes do sistema. Ambas visões estabelecem especificações independentes da distribuição dos seus elementos. O modelo ODP computacional provê uma descrição orientada a objeto do sistema como um conjunto de componentes cooperativos, que potencialmente podem ser distribuídos. Sua linguagem oferece um conjunto de conceitos e regras para estruturar o sistema de gerenciamento. O ponto de vista de engenharia estabelece um ambiente no qual as interações computacionais possam ser estabelecidas, descrevendo mecanismos, funções e transparências necessárias para suportá-las. O ponto de vista de tecnologia cobre os detalhes dos componentes no qual o sistema de gerenciamento é implementado. Objetivos, políticas, informação, objetos,

modelagem e interações são palavras chaves nos cinco pontos de vista e no gerenciamento de redes, de sistemas e de aplicações.

Os pontos de vista computacional e de engenharia fazem a distinção entre projeto e implementação dos sistemas de gerenciamento. No primeiro, um sistema de gerenciamento ODP é especificado como um conjunto de objetos computacionais, cujas interfaces podem ser tanto operacionais, de sinais ou do tipo "stream". Os objetos computacionais de gerenciamento podem ter diferentes interfaces, cada uma representando uma visão diferente. Eles podem representar diferentes papéis nas suas interfaces: cliente ou servidor, produtor ou consumidor, gerenciado ou gerente. Um objeto gerenciado é uma entidade cujo comportamento pode ser monitorado e controlado por um objeto gerente. Suas múltiplas interfaces podem oferecer diferentes visões e prover informações para distintas áreas funcionais de gerenciamento: desempenho, contabilização, segurança, falha e configuração. O objeto gerenciado pode receber invocações como um servidor ou ainda emitir notificações como cliente através de suas interfaces gerenciadas. O objeto gerente pode invocar operações como um cliente ou receber notificações através de suas interfaces de gerenciamento.

A linguagem de engenharia contém os conceitos que descrevem a infra-estrutura necessária para suportar as interações entre os objetos das aplicações de gerenciamento de forma transparente. Um objeto computacional pode corresponder a um ou mais objetos básicos de engenharia (BEOs), agrupados em "clusters". Um ou mais "clusters" compõem uma cápsula que pode ser comparado a um processo, com o seu próprio espaço de endereçamento. Uma ou mais cápsulas povoam um nó sob o controle de seu núcleo que é responsável pelas funções de gerenciamento do nó. Um sistema operacional (Kernel) é um exemplo de núcleo. Os serviços de um canal garantem as interações distribuídas transparentes entre BEOs. É constituído por três objetos de engenharia: "stub", "binder" e objeto protocolo. Esta estrutura é usada para prover interações transparentes entre objetos distribuídos nos papéis de cliente-servidor ou gerenciado-gerente. O "stub" é responsável pelo "marshalling" de parâmetros em um "buffer" de comunicação. O objeto "binder" estabelece e mantém as interações distribuídas entre diferentes objetos, sendo o maior responsável pela transparência de localização. O objeto protocolo oferece o serviço de comunicação de suporte às interações remotas. Esta infra-estrutura estabelecida precisa ser gerenciada. Uma entidade gerente deve monitorar e controlar BEOs, os objetos dos canais e os núcleos dos nós através de interfaces de gerenciamento em uma visão integrada (Figura 1).

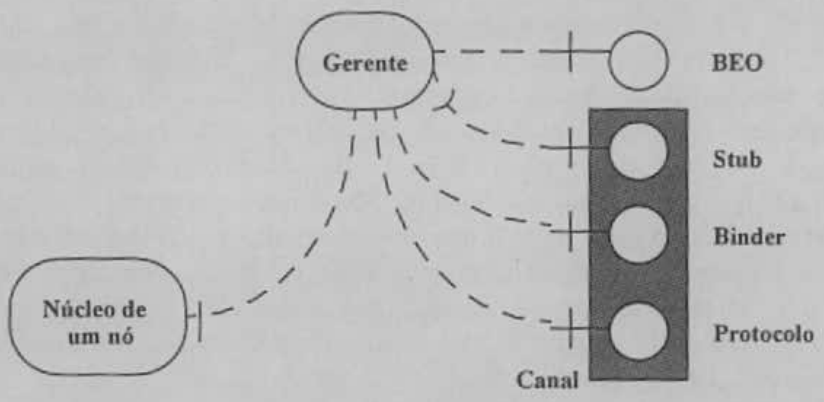

FIGURA 1. Visão Integrada do Gerente

3 Arquitetura de Gerenciamento de Sistemas Distribuídos

A arquitetura descrita nesta seção é parte da plataforma Multiware, ilustrada na Figura 2. É sedimentada em três camadas de suporte a aplicações distribuídas: uma de *Hardware/Software* básico, a Middleware e a Groupware. A camada Middleware é composta de uma ORB e de alguns serviços ODP, como o Trader (Lima, 1995), Suporte a Transação e a Grupo (Costa, 1996) e Gerenciamento de Sistemas (Queiroz, 1997). A Middleware está sendo desenvolvida sobre a Orbix da Iona Technologies (Iona, 1995). Sobre esta camada, aplicações CSCW (Computer Supported Cooperative Work) são suportados pela Groupware.

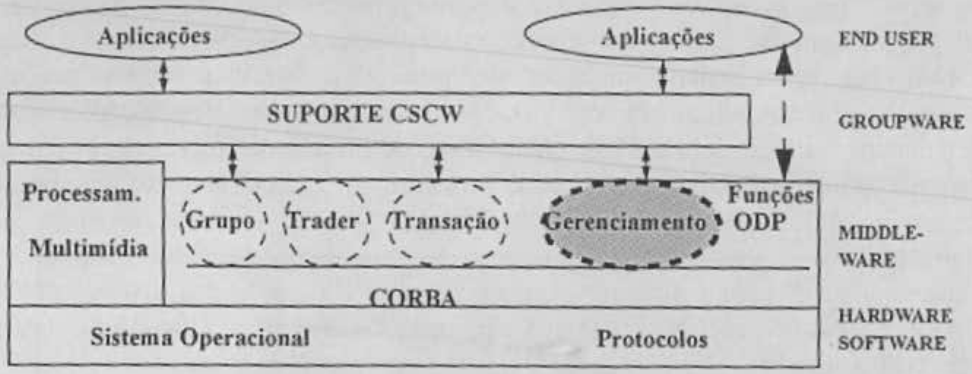

FIGURA 2. Plataforma Multiware

A arquitetura de gerenciamento integrado é baseada na ODMA e usa a CORBA, os CORBAservices e as CORBAfacilities, provendo pelo menos a mesma funcionalidade do modelo de gerenciamento OSI/ISO. Ela também foi influenciada pelos conceitos descritos em (Sloman, 1993; Sloman, 1995; Bauer, 1994). A arquitetura aborda o domínio das aplicações, integrado com os sistemas subadjacentes e os componentes de rede; permite a monitorização, a execução de ações de configuração e o controle dos recursos gerenciados; oferece uma interface gráfica comum e consistente, simplificando a interação do usuário; mantém o custo de processamento e de consumo de memória o mais baixo possível; e atende às necessidades de interoperabilidade e escalabilidade, inerentes aos sistemas distribuídos. Tal arquitetura é ilustrada na Figura 3. Não existe uma única aplicação monolítica, centralizada em uma plataforma, desempenhando todos os aspectos do processo de gerenciamento. Existe um conjunto de aplicações cooperativas oferecendo uma interface única ao usuário, abordando as áreas funcionais de desempenho, de contabilização, de configuração, de falhas e de segurança. Elas habilitam uma visão logicamente centralizada do domínio gerenciado, independente da distribuição física de seus componentes. Há ainda um conjunto de Facilidades para Monitorização, para Controle, para Configuração, para o estabelecimento de Políticas e de Domínios, usadas como pequenas peças de um "lego" para o desenvolvimento das aplicações. O nível de suporte, composto pela plataforma Multiware sobre uma ORB com os CORBAservices e as CORBAfacilities, auxilia o desenvolvimento e implementação das aplicações e das facilidades. Mais ainda, este nível permite que os objetos gerenciados e gerentes sejam compilados em diferentes linguagens, desde que exista o correspondente mapeamento para CORBA IDL. O uso dos CORBAservices ainda garante portabilidade e reusabilidade de código, permitindo que os desenvolvedores manipulem objetos que são independentes do comportamento que herdam das classes.

A monitorização consiste na observação das atividades dos componentes dos sistemas distribuídos e da aquisição de dados estáticos e dinâmicos através de requisições e notificações. A Facilidade de Monitorização é definida como um processo dinâmico para aquisição, coleta e apresentação de informações sobre os recursos gerenciados, essencial ao processo de decisão. Um serviço de filtragem é necessário para selecionar os dados coletados. A monitorização inclui as tarefas de geração, de processamento, de disseminação e de apresentação da informação monitorada (Mansouri-Samani, 1993). O Serviço de Gerenciamento de Eventos (OMG, 1995b) é usado para estabelecer comunicação assíncrona entre os objetos gerenciados e as entidades gerentes. Eventos podem ser gerados a partir de limites estabelecidos, usando os modelos "push" e "pull" através um canal de eventos. Os recursos do AWT (Abstract Windowing Toolkit) da linguagem Java são usados para criar apresentações dinâmicas e interativas das informações de gerenciamento coletadas e filtradas, através de desenhos gráficos e componentes como listas, diálogos e botões.

A Facilidade de Controle desempenha ações para evitar a degeneração de um sistema distribuído, controlando o comportamento dos objetos gerenciados. Tais situações são estabelecidas a partir da informação do estado de um componente, exigindo ações reativas, preventivas e pró-ativas. Estas condições são normalmente detetadas pela Facilidade de Monitorização. Os seus principais clientes são a Facilidade de Configuração e as aplicações de desempenho. Um aspecto muito importante é garantir que somente as ações permitidas pela Facilidade de Políticas sejam executadas por gerentes ou por sistemas devidamente autorizados e autenticados.

A Facilidade de Configuração é capaz de estabelecer a configuração inicial de um sistema distribuído, acompanhar as mudanças e executar as modificações, quando necessárias. Em grandes sistemas, intervenções humanas podem criar erros e causar falhas. Tal facilidade é derivada do Serviço de Ciclo de Vida para permitir a criação, destruição, cópia e movimentação dos objetos gerenciados. Um serviço de migração é desejável para migrar objetos para outros nós por motivos de balanceamento de carga ou para garantir qualidade de serviço.

Um aspecto importante é a especificação de quais operações de gerenciamento um gerente pode executar através de permissões, obrigações e proibições com o objetivo de guiar o processo de tomada de decisão. Existe portanto uma necessidade de se especificar, representar e manipular políticas. A especificação começa com políticas abstratas que são refinadas, após muitas interações, em ações que possam ser interpretadas por sistemas computacionais. A Facilidade de Políticas oferece operações para a criação e destruição de políticas, para ler e escrever seus atributos, para verificar políticas associadas a um domínio e determinar os domínios sujeitos a uma determinada política. A Facilidade de Domínios é usada para agrupar objetos sujeitos a uma política.

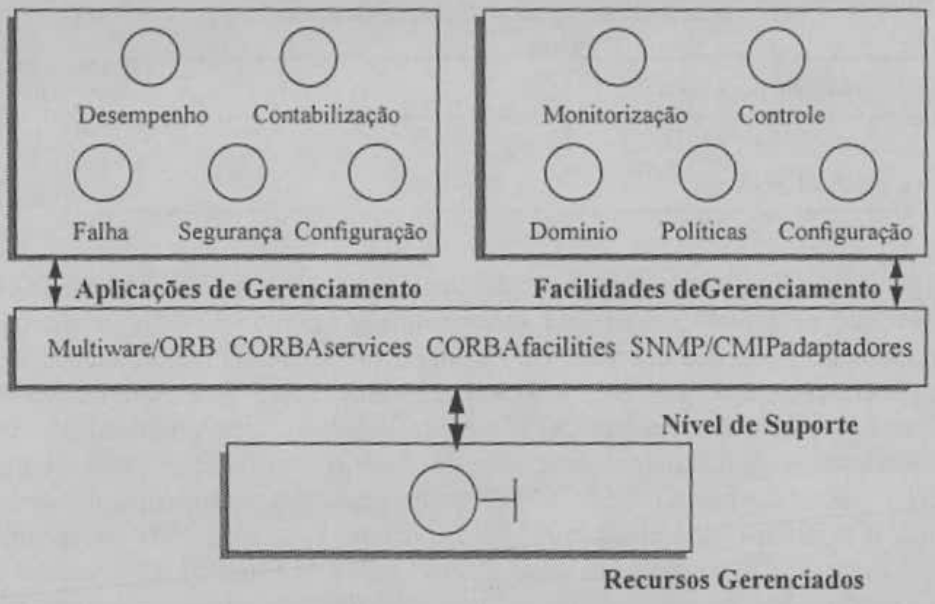

FIGURA 3. Arquitetura de Gerenciamento

Em um ambiente ODP, existem múltiplas visões de gerenciamento e diferentes limites de responsabilidades (Yemini, 1993). O gerenciamento deve ser estruturado em domínios para dividir responsabilidade e autorização entre diferentes gerentes (Sloman, 1995). Tal estruturação pode refletir a conectividade física da rede, um domínio da plataforma ou ainda o organograma administrativo. Os domínios estabelecidos não encapsulam seus membros. Eles são entidades passivas que mantêm referências para seus membros que são especificados explicitamente e não em termos de um critério de agrupamento. A Facilidade de Domínio provê operações para a escrita e leitura de atributos de um domínio, para a criação e destruição de domínios, para listar os seus componentes e inserir ou remover membros de um domínio.

O nível de suporte, composto pela plataforma Multiware sobre uma ORB e alguns CORBAservices, permite o desenvolvimento de aplicações de gerenciamento, simplificando a implementação e garantindo interoperabilidade. O Serviço de Tempo é desejável para se ter um tempo padrão acurado e preciso para sincronizar atividades de gerenciamento e mensagens. O Serviço de Nome é empregado para resolução de nomes em um contexto de gerenciamento. O Serviço de Eventos é concebido para desacoplar as comunicações entre objetos fornecedores que geram eventos e os consumidores que os recebem e os processam, usando os modelos "push" ou "pull". O Serviço de Ciclo de Vida é usado para permitir as operações para criação, destruição, cópia e movimentação de objetos. O Serviço de Persistência oferece um conjunto de interfaces para os mecanismos usados para gerenciar o estado persistente dos objetos gerenciados, a partir dos quais pode-se reconstruir o estado dinâmico. Finalmente, o Serviço de Segurança controla o acesso às informações de gerenciamento, autentica as partes envolvidas e criptografa mensagens quando necessário.

4 Modelagem e Projeto do Sistema de Monitorização

Este trabalho aborda as áreas funcionais de desempenho e de contabilização, motivado por freqüentes questões surgidas durante a operação de um sistema distribuído. Essas questões buscam identificar as possíveis causas de um excessivo tempo de resposta de uma aplicação e os usuários responsáveis pelo consumo de recursos num ambiente heterogêneo como o da Multiware. A maioria das métricas obtidas para o gerenciamento desempenho são também usadas para contabilização. O nosso foco está nas interações operacionais entre objetos nos papéis de cliente e servidor.

Para se especificar um sistema de monitorização flexível, genérico e eficiente, é necessária uma abordagem modular. A arquitetura proposta, derivada de um modelo ODP (Hoffner, 1994), é composta dos módulos de geração, observação, coleta, armazenamento, processamento e apresentação (Figura 4).

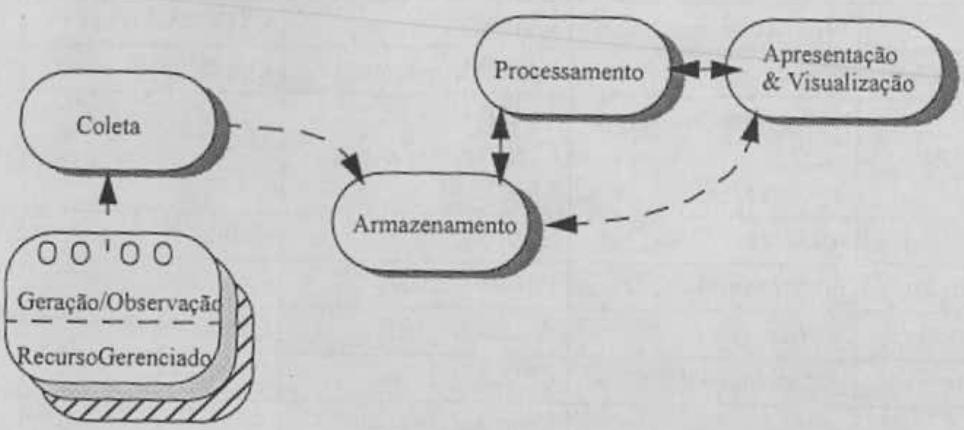

FIGURA 4. Modelo de Monitorização

O módulo de geração é responsável pela aquisição dos dados brutos de monitorização como resultado de um importante evento interno do recurso gerenciado. Esta atividade é desempenhada pela instrumentação de sensores para gerar uma determinada métrica. A observação oferece a interface para que os dados instrumentados sejam acessados e ainda pode oferecer um ponto de controle. O módulo de coleta adquire dados de vários objetos gerenciados, permitindo as atividades subseqüentes. O módulo de armazenamento se responsabiliza pela manipulação dos dados persistentes, usando o serviço disponível. O módulo de processamento é necessário para transformar, combinar, correlacionar, filtrar e analisar diferentes dados de gerenciamento. Esta atividade é dependente da área funcional e do escopo da monitorização. Os componentes do módulo de apresentação e visualização oferecem interfaces gráficas ao usuário. Os recursos GUI (Graphical User Interface) de Java e do seu AWT oferecem as facilidades necessárias. "Applets" Java em uma página Web permitem ao gerente navegar num sistema de gerenciamento integrado, usando as ferramentas disponíveis, de acordo com o conceito de aplicações Intranet. Em geral, a execução dos métodos de uma aplicação AWT requer métodos de inicialização, de pintar, parar e destruir objetos gráficos.

Um sistema de gerenciamento requer uma maneira de obter dados dos componentes instrumentados das aplicações distribuídas. Esses dados são gerados por sensores, criados no espaço de endereçamento dos processos cliente/servidor, que são objetos especializados incorporados para a criação de métricas de interesse. Dados dinâmicos são criados por sensores, como contadores e temporizadores, instanciados para adquirir uma determinada métrica. Tais sensores devem ser instalados onde ocorrem transições de eventos significativos. Um outro tipo de sensor, conhecido como composto, é definido para retornar atributos específicos como a identificação do recurso gerenciado, o seu estado operacional e outros. O primeiro passo em direção à implementação do nosso protótipo foi investigar uma maneira de se instrumentar as aplicações CORBA. A Figura 5 mostra o esquema estático do ponto de vista ODP de informação, usado para especificar o sistema de monitorização, apresentando as hierarquias de classe, de acordo com a notação gráfica da Técnica de Modelagem de Rumbaugh.

As métricas de desempenho escolhidas basicamente foram tempo de resposta, vazão e utilização (Rolia, 1993) traduzidas para o ambiente CORBA. Tal tradução foi baseada na especificação, apresentada em (Friedrich, 1995b), para monitorização de aplicações DCE (Distributed Computing Environment), exigindo o uso de contadores, temporizadores e sensores do tipo composto. Do ponto de vista do cliente, as métricas coletadas são: o número de requisições enviadas, o tempo de "marshalling" dos parâmetros, o número de repostas recebidas, o tempo de "unmarshalling" dos parâmetros e o tempo de resposta. Da perspectiva do servidor, elas são: o número de requisições recebidas, o tempo de "unmarshalling" dos parâmetros, o tempo de residência, o número de respostas enviadas, o tempo de "marshalling" dos parâmetros, a utilização e a vazão. A instrumentação calcula dados estatísticos como os valores máximos e mínimos, o somatório simples e o somatório dos quadrados para computar a média e a variância durante um intervalo de tempo. Os serviços oferecidos por cada elemento da aplicação devem ser monitorados e caracterizados separadamente. Outras informações são também necessárias para os propósitos de desempenho e contabilização. Esses dados podem ser visualizados na Tabela 1, onde se pode verificar os métodos usados na instrumentação. Apenas a operação *get_principal()* é disponível na arquitetura CORBA, as demais são

específicas da Orbix. No Anexo A, é apresentada a interface do objeto instrumentado, similar a uma MIB (Management Information Base).

TABELA 1. - Atributos da Interface

Dados	Operação	Classe	CORBA/Orbix
nome da operação	*operation()*	*CORBA::Request*	Orbix
nome da implementação	*myImplementationname()* *myServer()*	*CORBA::BOA* *CORBA::ORB*	Orbix
identificador	*myMarkerName()*	*CORBA::BOA*	Orbix
modo de ativação	*myactivationmode()*	*CORBA::BOA*	Orbix
máquina hóspede	*myHost()*	*CORBA::ORB*	Orbix
usuário cliente	*get_principal()*	*CORBA::BOA*	CORBA
pid	*getpid()*	Unix system call	--

5 Implementação do Protótipo

A plataforma Multiware está sendo implementada em estações IBM RS/6000 e Sun Sparc, com os sistemas operacionais AIX, SunOS e Solaris, conectados por redes ethernet, fast ethernet e FDDI. Este ambiente está disperso em dois laboratórios, inter-conectados por um enlace Internet de 10 Mbits. Portanto, o sistema de monitorização deve ser portável para tais sistemas heterogêneos.

Os filtros de processos, disponíveis na antiga Orbix1.3, na atual Orbix2.1 Multi-threaded e na recente OrbixWeb2.0 com o mapeamento IDL-Java, permitem que um código adicional seja executado antes ou depois do código normal de uma operação. Eles podem ser usados como mecanismos de auxílio para autenticação, para criação de um "lightweight thread" e para monitorização (IONA, 1995). São capazes de monitorar todas as requisições e respostas do espaço de endereçamento de um cliente ou de um servidor. Tais filtros desempenham as funções dos sensores, especificados na seção anterior, podendo ser instalados em cadeia.. Tais mecanismos, não previstos na especificação CORBA, têm a característica modular exigida pela instrumentação. Um filtro no lado cliente pode adicionar dados para o "buffer" de saída de uma requisição, de tal modo que possa ser recuperado no filtro correspondente do lado servidor. Similarmente, um filtro pode adicionar dados a uma reposta.

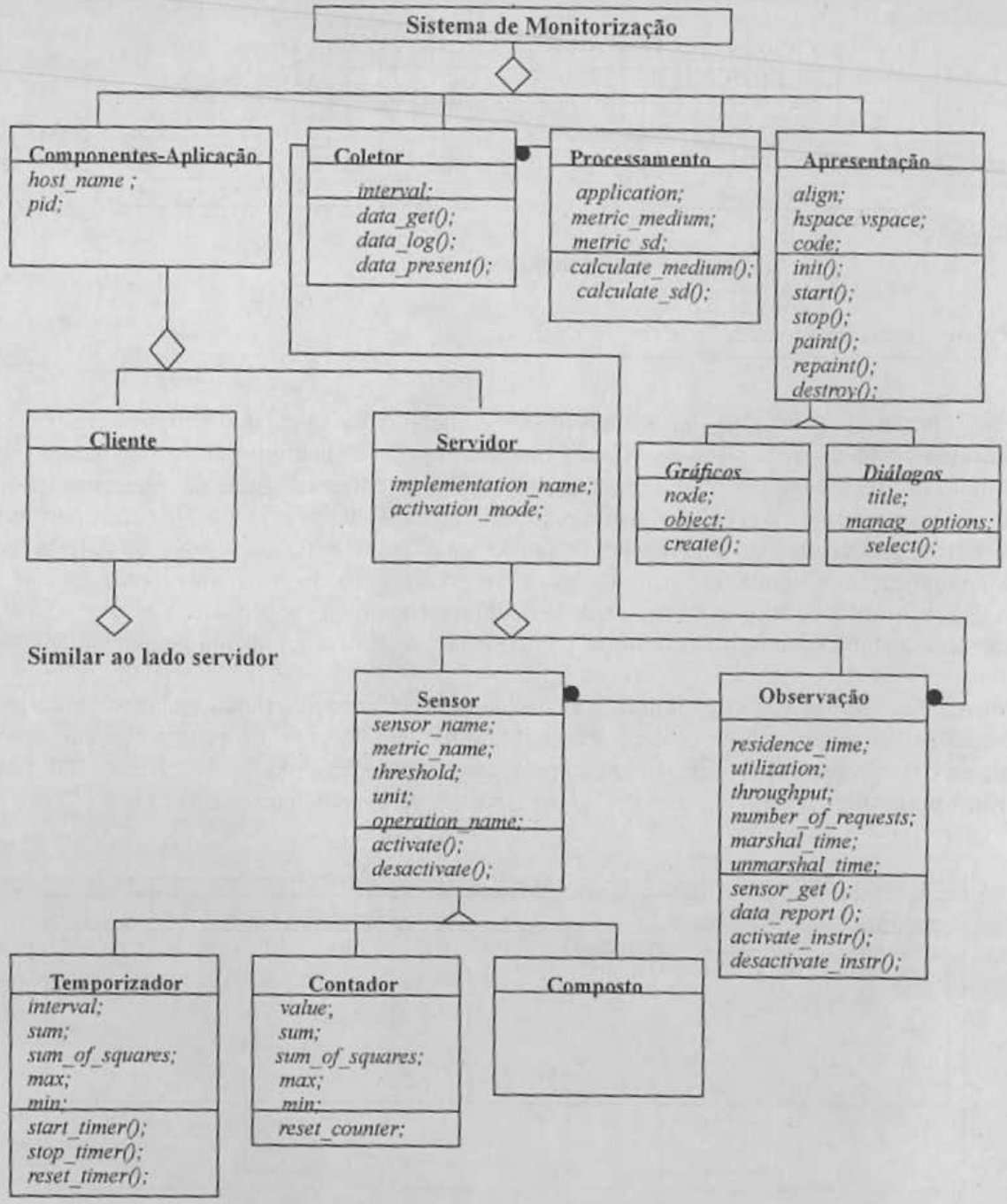

FIGURA 5. Esquema Estático

Os sensores implementados são derivados da super-classe *CORBA::Filter,* permitindo a instrumentação em dois pontos de uma requisição e em outros dois pontos de uma reposta, antes e depois do "marshalling" e "unmarshalling" dos parâmetros nos lados cliente e servidor. Foram redefinidos alguns métodos para executar as operações especificadas. No lado servidor, cinco sensores foram instanciados, três como temporizadores, um como contador e um outro como composto para retornar dados adicionais (Figura 6). Um temporizador foi instalado para medir o tempo de resposta do servidor e os outros dois para calcular os tempos de "marshalling" e "unmarshalling". Foram ainda criados objetos do tipo relógio para oferecer um serviço de tempo. Da mesma forma, cinco sensores foram instalados no lado cliente: três temporizadores, um contador e um composto. O tempo de resposta de um cliente é calculado por um temporizador, instalado no seu espaço de endereçamento, com o disparo do relógio instanciado por uma requisição enviada, sendo parado pela resposta esperada do servidor. O código em C++ desse temporizador é mostrado no Anexo B. Nós estimamos que o número potencial de sensores usados possa ser maior ainda, na medida que se deseje adquirir mais dados.

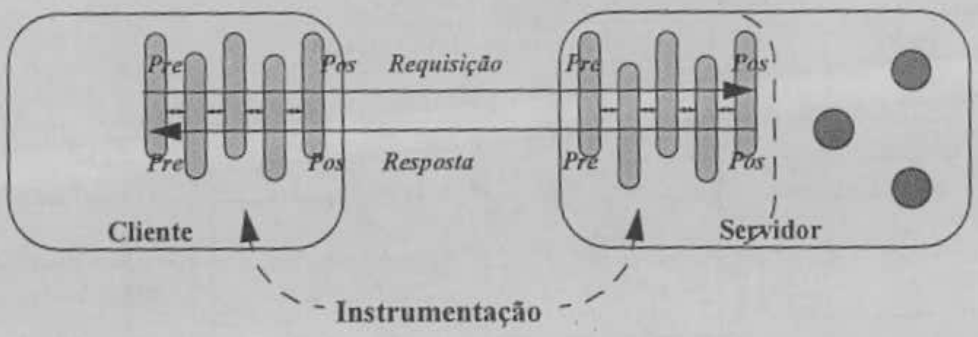

FIGURA 6. Cadeia de Sensores

Inicialmente, um temporizador foi instanciado no lado cliente para calcular o tempo de resposta entre uma requisição enviada e uma resposta recebida, visando medir o impacto da instrumentação inserida no servidor. Essa medida foi muito útil para controlar o "overhead" associado durante todas as fases da implementação. Nós ainda notamos que os maiores tempos de resposta são de requisições para o "daemon" *orbixd* que manipula as referências de objetos CORBA. Tais valores são dependentes de fatores estáticos e dinâmicos dos nós, do sistema operacional e da rede de comunicação. O ambiente heterogêneo e sobrecarregado da Multiware permitiu confirmar estas observações, levando-nos à conclusão da necessidade de integração do gerenciamento. Um alto valor de tempo de resposta pode ser ocasionado tanto pelos próprios componentes da aplicação, quanto pela máquina hospedeira ou pela rede.

Paralelamente, nós verificamos a possibilidade de que algumas informações monitoradas poderiam ser úteis para a área funcional de gerenciamento de configuração como o número da porta de escuta, o modo de ativação e o identificador; ou o nome do usuário cliente para a área de segurança. Os atributos da interface IDL (Anexo A) de um servidor instrumentado apresenta apenas os dados essenciais para o nosso propósito.

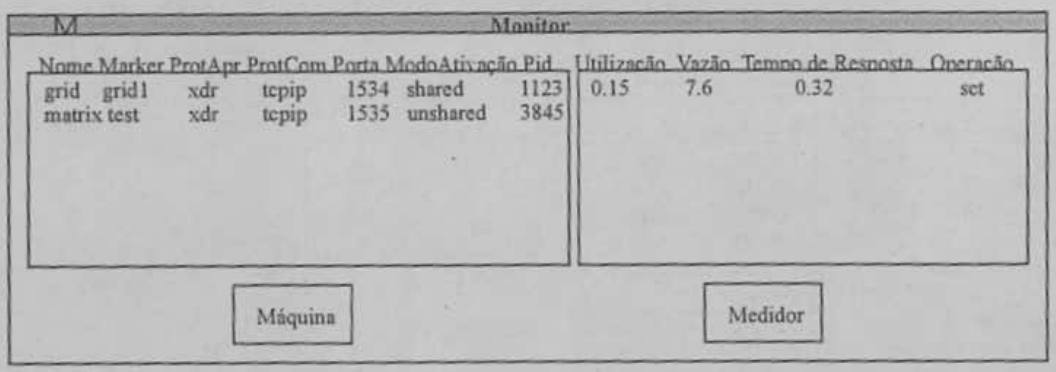

FIGURA 7. Interface Gráfica em Java

Estes dados são visualizados através de um cliente Java, no ambiente da OrbixWeb, através da interface gráfica mostrada na Figura 7. O painel é composto basicamente de dois botões e de duas listas, cujos eventos criados estão associados dois a dois. O botão Máquina admite a escolha do nó do domínio a ser monitorado. Este domínio é facilmente configurado através da facilidade do localizador, existente na Orbix, que estabelece uma relação de prioridade de ativação dos servidores com nomes das máquinas escolhidas. O evento associado a esse botão lista à esquerda os dados de todos objetos servidores que estão sendo executadas naquele "host". Estes dados são obtidos através do método *IT_daemon::listActiveServers()*, da interface IDL do "daemon" orbixd, retornando a estrutura do tipo *serverDetails*. A seleção de um determinado objeto CORBA apresentado e do botão de medição dispara uma requisição para a interface do servidor instrumentado (Anexo A), cuja resposta é apresentada na lista à direita. Estas interações podem ser vistas na Figura 8.

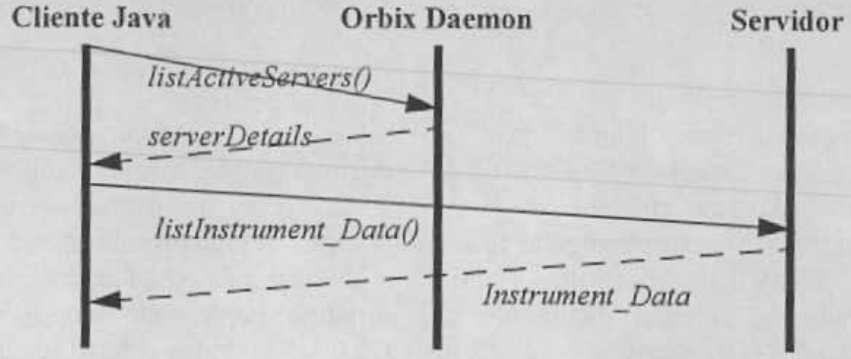

FIGURA 8. Interações dinâmicas

O uso de Java permite que este sistema seja integrado com outras ferramentas de gerenciamento de redes e de sistemas em um "Web browser". Dessa forma, um gerente pode navegar através de "applets", visualizando outros dados de interesse como carga de CPU, atividades de disco e de paginação dos nós sobre os quais os componentes das aplicações CORBA estão sendo executados; e monitorando os elementos do subsistema de comunicação.

O ambiente heterogêneo da Multiware permitiu a monitorização de aplicações instrumentadas em diferentes configurações. A Tabela 2 apresenta um exemplo de amostragem de dados coletados de dois objetos CORBA no ambiente Multiware. O tempo de residência do servidor é apenas uma componente do tempo de resposta do cliente. A carga do canal ODP, distribuída pelos objetos "stubs", "binder" e protocolo, degrada a resposta do cliente, sem afetar o tempo de residência do servidor. Por exemplo, os valores de média e desvio padrão do tempo de resposta do cliente foram 5,24 ms e 0,81, enquanto os valores do tempo de residência foram 0,36 ms e 0,12 respectivamente.

TABELA 2. - Amostragem

Nome do Server	Máquina	pid	Utilização	Tempo de Residência	Vazão	Operação mais requisitada
matrix	marumbi	4483	0.11	0.32	7.8	set
matrix	marumbi	4483	0.13	0.35	8.0	set
grid	trancoso	1181	0.21	0.29	9.8	get
grid	trancoso	1181	0.20	0.30	8.0	get

A Tabela 3 mostra as diferenças entre os tempos de respostas do ponto de vista do cliente e do servidor, com os valores de média e desvio padrão, em três situações distintas: o cliente e o servidor na mesma máquina, em diferentes máquinas e em diferentes laboratórios.

TABELA 3. - Diferenças de Tempos de Respostas

Localização	Mesma máquina	Diferentes Máquinas	Diferentes LAN's
Média	4.18	5.01	8.53
D.P.	1.09	1.15	3.39

6 Trabalhos Relacionados

A questão do gerenciamento integrado ODP é também apresentada em diferentes trabalhos (ITU-T/ISO, 1995b; Farooqui, 1995). Friedrich et all em (Friedrich, 1995a) descrevem uma arquitetura e apresentam um protótipo para monitorização de aplicações baseadas em DCE. Embora mostre informações que auxiliem a entender o comportamento operacional do DCE, tais dados são também úteis para gerenciamento. (Schade, 1996) propõe um método para integrar uma aplicação distribuída em um ambiente de gerenciamento, baseado em uma interface Management IDL, extensão da CORBA IDL, e uma biblioteca de instrumentação para suportar o desenvolvimento de aplicações capazes de serem gerenciadas. (Uslander, 1996) também apresenta uma arquitetura para o gerenciamento de aplicações CORBA, integrada com os atuais padrões de gerenciamento de redes e de sistemas. O

nosso trabalho apresenta uma maneira modular de instrumentar objetos CORBA e de monitorá-los, no contexto maior da arquitetura de gerenciamento integrado para uma plataforma ODP.

7 Conclusão

Este artigo apresentou um sistema de monitorização ODP sobre CORBA, usando os mecanismos diponíveis para instrumentar os componentes das aplicações CORBA de uma forma simples e direta e apresentando esses dados em uma interface gráfica em Java no ambiente da OrbixWeb. Descreveu um modelo de objetos, discutiu alguns aspectos de implementação e mostrou alguns resultados. Esses resultados observados com aplicações num ambiente heterogêneo, como o da Multiware, levam-nos à direção da necessidade de se adotar o conceito de gerenciamento integrado de sistemas distribuídos. O protótipo implementa uma Base de Informação de Gerenciamento definida em IDL, acessível por objetos CORBA. Os dados coletados são informações úteis de gerenciamento para as áreas funcionais de desempenho e de contabilização. A experiência obtida com CORBA mostrou que os esforços de padronização também devem ser concentrados em gerenciamento. A maioria dos mecanismos e operações utilizadas na instrumentação são disponíveis apenas na plataforma Orbix. Elas não são parte da especificação CORBA. Recentemente foi aprovada pela OMG a submissão do consórcio X/OpenSysMan (X/OPEN, 1995), com uma proposta de uma infra-estrutura para o desenvolvimento de aplicações de gerenciamento de sistemas distribuídos. Embora a própria X/Open tenha reconhecido que o documento abordava apenas um subconjunto das facilidades necessárias, ele foi aceito, além de não ter absorvido a especificação de outros importantes CORBAservices, como o Serviço de Eventos, e as CORBAfacilities.

O objetivo da arquitetura de gerenciamento de sistemas distribuídos descrita é oferecer uma estrutura e um conjunto de serviços úteis para o gerenciamento de aplicações CORBA para alcançar a integração dos três domínios: redes, sistemas e aplicações. Os componentes da arquitetura são objetos CORBA, acessíveis por operações ORB como protocolo de gerenciamento. Por outro lado, o RM-ODP provê uma infra-estrutura para aplicações distribuídas, capaz de ser usada para o seu gerenciamento, como um ponto de partida para a integração. Contudo, a padronização está mais concentrada no desenvolvimento de sistemas distribuídos do que no seu gerenciamento. Embora a técnica de modelagem do padrão OSI/ISO não seja adequada para o gerenciamento de sistemas distribuídos, a experiência obtida deve ser usada para o gerenciamento da arquitetura ODP. Os esforços da padronização ODMA estão tentando suprir essa deficiência.

Agradecimentos:

Agradecemos à FAPESP e ao CNPq pelo apoio financeiro na realização deste trabalho.

8 Referências

Ban, B. (1996) Towards an Object-Oriented Framework for Multi-Domain Management. *Proceedings of ECOOP'96 Workhop on System and Network Management,* Linz, Austria, july.

Bauer, M.A. et al (1994) Reference architecture for distributed systems management. *IBM Systems Journal vol 33, no 3*, pp 426-444.

Case J., Fedor M. and Davin J. (1990) A Simple Network Management Protocol (SNMP). *RFC 1157 Networking Group*, may.

Costa, F.M. and Madeira, E.R.M. (1996) An object group model and its implementation to support cooperative applications on CORBA. *Distributed Platforms (*Ed. A. Schill, C. Mistach, O. Spaniol and C. Popien), Chapman & Hall, pp 213-227.

Farooqui, K. (1995) OSI Management in the ODP Architectural Framework. *Proc. IFIP/IEEE DSOM'95-* pp 1-13, october.

Friedrich, R.; Martinika, J.; Sienknecht, T. and Saunders, S. (1995a) Integration of Performance and Modelling for Open Distributed Processing. *Proc. of ICODP'95,* pp 341-352, february.

Friedrich, R.; Saunders, S.; Zaidenweber, G.; Bachman, D. and Blumson, S. (1995b) Standardized Performance Instrumentation and Interface Specification for Monitoring DCE Based Applications. *OSF DCE RFC33.0,* may.

Hegering, H.G. and Abeck, S. (1994) Integrated Network and System Management. *Data Communication and Network Series.*Addison-Wesley.

Hoffner, Yigal. Monitoring in Distributed Systems. *ANSA Phase III APM 1018.01.*outubro.

ITU-T Rec.X901/2/3 | ISO/IEC 10746-1/2/3 (1995a) ODP Reference Model, Part 1, Overview and Guide

to use; Part 2. Foundations; Part 3. Architecture.

ITU-T | ISO/IEC (1995b) Open Distributed Management Architecture, Working Draft 3, november.

Iona Technologies Ltd. (1995) Orbix 1.3 advanced programmer's guide, release 1.3.1.

Soukoti, N. e Hollberg, Ulf (1997) Joint Inter Domain Management: CORBA, CMIP and SNMP. próximo *Fifth IFIP/IEEE International Symposium on Integrated Network Management*, maio.

Lima, L.A.P. and Madeira, E.R.M. (1995) A model for a Federative Trader. *In Proc. of the ICODP'95*, pp 155-166, february.

Loyolla, W.; Madeira, E.R.M.; Cardozo, E.; Magalhães, M.F. and Mendes, M.J. (1994) Multiware Platform: An open distributed environment for multimedia cooperative applications. *IEEE COMPSAC'94*, november.

Mansouri-Samani, M. e Sloman, M. (1993). Monitoring Distributed Systems. *IEEE Network magazine*, pp 20-30, 7(6), novembro.

OMG (1995a) CORBAfacilities: Common Facilities Architecture, rev. 4.0, january.

OMG (1995b) CORBAservices: Common Object Services Specification, rev. march.

OMG (1995c) Common Object Request Broker: Architecture and Specification, rev 2.0, july.

Queiroz, J.A.G. e Madeira, E.R.M (1997) Management of CORBA objects monitoring for the Multiware platform. próximo *IFIP Joint International Conference on Open Distributed Processing (ICODP) and Distributed Platforms (ICDP)*, maio. www.ocri.ca/events/icodp/icodprog.html

Rolia, J.A. (1993) Distributed Application Performance, Metrics and Management. *Proc. of ICODP'93*, pp 205-216.

Schade, A.; Trommler, P. and Kaiserswerth, M. (1996) Object Instrumentation for Distributed Applications Management. *Distributed Platforms (*Ed. A. Schill, C. Mistach, O. Spaniol and C. Popien), Chapman & Hall, pp 173-185.

Sloman, M.; Magee, J.; Twidle, K. and Krammer, J. (1993) An Architecture for Managing Distributed Systems. *Proc. 4th IEEE Workshop on Future Trends of Distributed Computing Systems,* pp 40-46.

Sloman, M. (1995) Management Issues for Distributed Services. *Proc. IEEE SDNE'95*, pp 52-59, june.

Usländer, T. and Brunne, H. (1996) Management View upon CORBA Clients and Servers. *Proc. of ICDP'96,* Industrial and Poster Session, pp 165-169.

X/Open Company (1995) System Management: Common Management Facilities, Volume 1, Version 2 - OMG 95-12-05, december.

Yemini, Y. (1993) The OSI Network Management Model.*IEEE Communications Magazine*, pp 20-29, 31(5), may.

Anexo A - Interface IDL

```
interface Server_Managed_Object {
  struct Instrument_Data {
    enum state {idle, unmarshalling_request, processing, marshalling_response, committing,};
    readonly attribute string operation_name;
    readonly attribute string implementation_name;
    readonly attribute string marker_name;
    readonly attribute string activation_mode;
    readonly attribute string hostname;
    readonly attribute string principal;
    readonly attribute long number_of_request;
    readonly attribute double unmarshalling_time;
    readonly attribute long number_of_response;
```

```
        readonly attribute double marshalling_time;
        readonly attribute double throughput;
        readonly attribute double utilization;
        readonly attribute double residence_time;
    };
    void listInstrument_Data(out Instrument_Data);
};
```

Anexo B - Temporizador

```
// este sensor mede o tempo resposta do cliente
class Response_timer :: Filter {
//  instanciação do relógio
        Clock* cl1;
//  estrutura de gerenciamento
        Instrument_Data i_d;
public:
// uma requisição recebida dispara o relógio
        unsigned char outRequestPreMarshal (CORBA::Request &r, Environment &) {
                cl1.start_timer();
                return 1;
        };
// uma resposta enviada para o relógio
        unsigned char inResponsePosMarshal (CORBA::Request &r, Environment &) {
                cl1.stop_timer();
                i_d.residence_time = cl1.diff_time;
                i_d.sum_residence_time += i_d.residence_time;
                i_d.sum_of_sq_residence_time += i_d.residence_time**2;
                delete[] cl1;
        return 1;
};
}
```