UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Ângela Mello Barotto

GERÊNCIA DE ALARMES EM AMBIENTES WEB, WAP E SMS USANDO AS TECNOLOGIAS SNMP, WBEM E CORBA

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos necessários para obtenção do grau de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Carlos Becker Westphall

Florianópolis, agosto de 2002

GERÊNCIA DE ALARMES EM AMBIENTES WEB, WAP E SMS USANDO AS TECNOLOGIAS SNMP,WBEM E CORBA

Ângela Mello Barotto

Esta dissertação foi julgada adequada para obtenção do título de Mestre em Ciência da Computação na Área de Concentração Sistemas de Computação e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

Prof. Dr. Fernando A. O. Gauthier - Coordenador

Prof. Dr. Carlos Becker Westphall - Orientador

Banca Examinadora:

Prof. Dr. Carla Merkle Westphall

Prof. Dr. Mario Dantas

Prof. Dr. Roberto Willrich

AGRADECIMENTOS

Eu gostaria de agradecer primeiramente aos meus pais pela oportunidade que me deram de poder estudar. Também agradeço a meu irmão André pela força e incentivo que ele me deu. Ao meu namorado Alexandre que me ajudou nas horas mais difíceis do mestrado. Aos meus irmãos e amigos que sempre estiveram por perto incentivando e dando apoio moral e ao meu orientador Carlos Westphall.

SUMÁRIO

Índice de Figuras

Lista de Abreviações

ASN.1	*Abstract Syntax Notation 1*
CDMA	*Code Division Multiple Access*
CDPD	*Cellular Digital Packet Data*
CIM	*Common Information Model*
CMIP	*Common Management Information Protocol*
COM +	*Component Object Model plus*
CORBA	*Common Object Request Broker Architecture*
CSD	*Circuit-Switched Cellular Data*
DCOM	*Distributed Component Object Model*
DMTF	*Distributed Management Task Force*
GPRS	*General Packet Radio Services*
GSM	*Global System for Mobile Communications*
HTML	*Hypertext Markup Language*
HTTP	*Hypertext Transfer Protocol*
IDL	*Interface Definition Language*
IP	*Internet Protocol*
ISO	*International Organization for Standardization*
LAN	*Local-Area Network*
MOF	*Managed Object Format*.
MTS	*Microsoft Transaction Server*
NMS	*Network Management Stations*
ODBC	*Open Database Connectivity*
OMG	*Object Management Group*
ORB	*Object Request Broker*
OSI	*Open System Interconnection*
PCS	*Personal Communications Service*
PDA	*Personal Digital Assistants*
SIM	*Subscriber Identity Module*

SMFA	*Specific Management Functional Areas*
SMI	*Structure of Management Information*
SMS	*Short Message Service*
SNMP	*Simple Network Management Protocol*
TCO	*Total Cost of Ownership*
TCP	*Transmission Control Protocol*
TDMA	*Time-Division Multiple Access*
TLS	*Transparent LAN Service*
UDP	*User Datagram Protocol*
UML	*Unified Modeling Language*
USSD	*Unstructured Supplementary Services Data*
WAE	*Wireless Application Environment*
WAP	*Wireless Application Protocol*
WDP	*WAP Datagram Protocol*
WBEM	*Web-Based Enterprise Management*
WMI	*Windows Management Instrumentation*
WML	*Wireless Markup Language*
WMLScript	*Wireless Markup Language Script*
WSP	*WAP Session Protocol*
WTA	*Wireless Telephony Application*
WTLS	*Wireless Transport Layer Security*
WTP	*WAP Transaction Protocol*
XML	*Extensible Markup Language*

Resumo

A operação eficiente e ininterrupta das redes de computadores é um requisito que tem sido cada vez mais cobrado atualmente. Uma falha na rede pode deixar instituições inteiras inoperantes. Por este motivo, detectar eventuais falhas e tomar ações corretivas de modo rápido e eficiente tem se tornado uma exigência de mercado. Deste modo, o Gerenciamento de Falhas em redes de computadores tem se tornado cada vez mais importante. Dentro deste contexto este trabalho propõem a implementação de uma aplicação destinada a detecção das falhas que mais prejudicam um ambiente de produção. Esta ferramenta foi desenvolvida em um ambiente WEB, WAP e SMS se propondo a ser um diferencial através da união de tecnologias como WBEM, SNMP e CORBA. O objetivo básico desta aplicação é prover um meio eficiente para a detecção e notificação de falhas, assim como, o acompanhamento das eventuais soluções para o problema.

Abstract

The efficient and uninterrupted operation of the computer networks is a requirement that has been getting more importance. A failure in the Network can leave inoperative entire institutions. For this reason, to detect eventual failures and to take action corrective in efficient way had become a market requirement. In this way, the Management of Failures in computer networks has become very important. In this context, this work did an implementation to detect failures that more harm a production environment. This tool was developed in a WEB, WAP and SMS environment. It is considered to be a differential through the union of technologies as WBEM, SNMP and CORBA. The basic objective of this application is to provide an efficient way for the detention and notification of failures, as well as, the accompaniment them eventual solutions for the problem.

1 Introdução

Atualmente, as redes de computadores exercem um papel fundamental para as pessoas que a utilizam, pois é através das redes que é disponibilizado acesso aos mais diversos tipos de recursos e serviços, criando por conseqüência uma certa dependência entre os usuários e os recursos da rede no sentido de que quando seja necessário a utilização de algum serviço de um determinado recurso, este deverá estar disponível.

Com o objetivo de prover tal disponibilidade, foram criados os padrões e tecnologias necessárias para que aplicações de gerência pudessem interagir com os recursos da rede. Estas aplicações proporcionam ao usuário, o conhecimento do funcionamento da rede, facilitando a busca por possíveis problemas e auxiliando na resolução destes.

O desenvolvimento de uma aplicação para o gerenciamento de redes, normalmente é movido pelas necessidades de um ambiente. De maneira a atender estas necessidades com eficácia, é preciso levantar os pré-requisitos que deverão ser atendidos para a aceitação da aplicação.

Atualmente já existem diversas ferramentas que implementam funções de gerência, mas que são dispendiosas e nem sempre atendem por completo determinadas necessidades de uma ou outra corporação. Isso faz com que aumentem o número trabalhos que visam a criação de ferramentas mais simples, mas que atendam as necessidades de um determinado ambiente.

Neste trabalho, foi analisado um ambiente que tem, em sua essência, recursos da plataforma Windows, pois esta plataforma é dominante [MIL01], mas como em qualquer outro ambiente, existem também recursos de outras plataformas. Essas características são encontradas em algumas instituições que requerem total confiabilidade na rede, pois uma falha pode deixar instituições inteiras inoperantes. Dentro deste contexto, este trabalho propõe uma solução eficiente e inovadora para a gerência de alarmes disparados em reação à falhas ocorridas na rede.

É proposta uma aplicação que faz a gerência dos alarmes de uma rede garantindo que as notificações das falhas ocorridas sejam repassadas aos responsáveis

de forma rápida e eficaz. A captação destas falhas abrange todas as plataformas da rede, mas possui um enfoque especial na plataforma Windows.

Para atender tais características, foi necessária a utilização de tecnologias como WBEM, SNMP e CORBA e a interação da aplicação com o usuário foi utilizada a WEB e WAP.

O WBEM (*Web-Based Enterprise Management*) é uma iniciativa da DMTF (*Distributed Management Task Force*) que objetiva a unificação de mecanismos para descrever e compartilhar informações de gerência, sem definir as regras de implementação, permitindo que a indústria desenvolva soluções de gerência padronizadas e baseadas nas tecnologias WEB emergentes. Desta forma, é possível utilizar diversas tecnologias (como por exemplo CORBA/Java) para estender esta estrutura para um objetivo próprio e até mesmo fazer a integração com o SNMP, que é o protocolo mais popular no que diz respeito a tecnologias de gerenciamento de redes [MAL01].

Fazendo uso das características de implementações baseadas na iniciativa WBEM, juntamente com os já consagrados SNMP e CORBA, através de uma aplicação baseada nos conceitos da componentização[1] foi possível criar uma aplicação completa para gerenciar uma rede, flexível para criação de aplicativos com diversos enfoques.

Com a interface dessa aplicação sendo feita baseada na WEB, é possível gerenciar uma rede a partir de qualquer lugar do mundo onde se tenha uma conexão à Internet. As vantagens de usar esse tipo de interface são várias, indo desde a substituição da linha de comando até a economia em treinamento de pessoas especializadas [MAC01]. Além disso, para proporcionar um melhor acesso a informações gerenciais, a utilização de dispositivos WAP vem crescendo ao longo dos últimos anos devido à demanda existente [STU02]. Este protocolo, mesmo com as dificuldades existentes no ambiente *wireless*, mostra-se bastante útil se bem utilizado, principalmente como forma de apoio a ferramentas de gerência.

---

[1] Componentização: implementação da aplicação utilizando-se do conceito de componentes da engenharia de *software*.

1.1 Motivação

Os ambientes de rede são compostos por diversos tipos de equipamentos interligados entre si. Geralmente essa interligação é proveniente da necessidade de dependência entre os equipamentos, como por exemplo, a utilização de um servidor de aplicações, onde várias máquinas dependem deste servidor para que possam ser executadas as aplicações nele armazenadas. Quanto maior for a dependência entre estes equipamentos maior será a necessidade de que estes estejam sempre em perfeito funcionamento, para que o ambiente se mantenha operacional como um todo. Este é o objetivo maior da gerência de redes de computadores, que é uma área da computação destinada ao estudo e desenvolvimento de soluções para a manutenção de ambientes de redes. O gerenciamento destes ambientes pode ser feito manualmente ou através de ferramentas desenvolvidas especialmente para este propósito. No decorrer dos anos, várias ferramentas foram surgindo para o auxilio na gerência, mas existem algumas questões que geralmente impedem a sua aquisição, como por exemplo:

- As ferramentas são bastante robustas e possuem diversas funcionalidades que nem sempre são utilizadas. Conseqüentemente, estas ferramentas têm um nível de complexidade elevado, o que implica na necessidade da contratação ou treinamento de pessoas especializadas.
- O preço costuma ser bastante elevado e os responsáveis por redes menores não acreditam na necessidade de se comprar ferramentas deste porte para suas redes.

Pelo fato destas redes menores, na sua maioria utilizando sistemas operacionais Windows [MIL01], não necessitarem de uma ferramenta de gerência tão especializada, elas acabam não fazendo o uso de nenhuma outra forma de gerenciamento mais sofisticada. O que ocorre normalmente é a detecção de problemas pelo administrador da rede, através de comandos próprios do sistema operacional ou simples observações dos equipamentos. Pequenos problemas que ocorrem nestes ambientes podem vir a atrasar o

trabalho de pessoas que dependem de seus computadores, do bom funcionamento dos servidores, roteadores e demais dispositivos da rede.

Ambientes com esse tipo de característica necessitam de ferramentas que sejam focadas na detecção de pequenos problemas de hardware e software e de problemas que prejudicam o funcionamento dos equipamentos fundamentais da rede. Ao detectar uma falha no sistema, o administrador deve ser avisado de forma rápida para que a rede volte ao seu estado de normal.

Para gerir ambientes como o descrito, este trabalho se propõe a fazer um estudo das tecnologias disponíveis para encontrar uma forma eficiente de implementar uma aplicação com os seguintes pré-requisitos:

- Simplicidade para a utilização de um administrador: não deve haver a necessidade de conhecimento profundo nas tecnologias envolvidas, apenas uma interface simples que esconde toda a lógica do funcionamento do sistema gerente.
- Tenha facilidade de incorporação de novas funcionalidades: caso surjam novas necessidades no ambiente, novas funcionalidades poderão ser adicionadas sem interferir no sistema existente.
- Seja considerada uma solução de baixo custo: esta característica é fundamental para ambientes de pequeno é médio porte mas que ainda necessitam de gerenciamento de forma automática.
- Notificação de responsáveis: deve possuir um sistema de notificação confiável e automatizado, de modo a garantir que um responsável seja avisado na ocorrência de problemas.
- Fácil acesso: o sistema deve permitir acesso de qualquer ponto onde se tenha uma conexão com a rede ou um dispositivo wireless.

1.2 Objetivos

1.2.1 Objetivo Geral

O objetivo geral deste trabalho é a apresentação de uma solução para gerência de alarmes para um ambiente de rede com notificação imediata na ocorrência destes. Para este objetivo, o sistema foi dividido em Módulo Gerenciador COM, Módulo Gerenciador CORBA,Módulo Gerenciador SNMP, Módulo Notificador, Gerador de Relatórios, Interface WEB e Interface WAP.

1.2.2 Objetivos Específicos

Como objetivos específicos deste trabalho, destacam-se:

- Conhecimento dos principais conceitos da gerência de redes, com foco na Gerência de falhas;
- Estudo dos padrões utilizados para construção de aplicações destinadas à gerencia de redes como SNMP, CORBA, WBEM;
- Estudar a arquitetura WMI, da Microsoft;
- Conhecimento do funcionamento do ambiente *Wireless* para poder avaliar viabilidade, necessidades, performance de uma aplicação que é executada neste ambiente;
- Dominar os conceitos do protocolo WAP, principalmente as linguagens WML e WMLScript, que juntamente com o ASP serão as linguagens que foram utilizadas na implementação do sistema;

- Caracterizar dentro da aplicação a importância da divisão em componentes, que neste caso foi feito utilizando a arquitetura Windows DNA;
- Desenvolver o sistema de gerência de alarmes conforme as necessidades do ambiente apresentados anteriormente, de forma a utilizar os conceitos previamente estudados para chegar a uma solução eficiente, fácil de utilizar e com baixo custo;
- Validar o sistema em um ambiente de produção de forma a avaliar suas vantagens e desvantagens comparadas à ferramentas de gerenciamento de falhas já existentes no mercado;

1.3 Organização do Trabalho

Este trabalho está organizado da seguinte forma:

- Capítulo 1 - Introdução: Apresenta os aspectos introdutórios do trabalho, a motivação e os objetivos gerais e específicos da dissertação.
- Capítulo 2 - Gerência de Redes: detalha os conceitos da gerência de redes juntamente com os protocolos e arquiteturas utilizados. São apresentadas também algumas ferramentas existentes no mercado e trabalhos que são realizados nesta área.
- Capitulo 3 – Tecnologias: são apresentadas as tecnologias necessárias para o desenvolvimento do sistema. No caso, a tecnologia *Wireless*, com seu protocolo WAP para desenvolvimento de aplicações e o SMS para envio de mensagens curtas, e a arquitetura de objetos distribuídos como CORBA e DCOM.
- Capitulo 4 - Proposta: conceitua a proposta do trabalho relacionando-o com os trabalhos já existentes na área.
- Capitulo 5 – Implementação: é descrito como foi feita a implementação e apresenta a validação do mesmo em um ambiente de testes.

- Capitulo 6 – Conclusão: é feita uma conclusão do trabalho relatando os aspectos positivos e negativos da ferramenta juntamente com as sugestões de trabalhos futuros.

2 Gerência de Redes

As redes foram concebidas, inicialmente, como um meio de compartilhar dispositivos periféricos mais caros como impressoras, *modems* de alta velocidade, painéis pc-fax, entre outros. Entretanto, à medida que as redes crescem e tornam-se integradas às organizações, o compartilhamento dos dispositivos toma aspecto secundário em comparação às outras vantagens oferecidas. As redes passaram a fazer parte do cotidiano dos usuários como uma ferramenta que oferece recursos e serviços que permitem a interação e o aumento de produtividade.

Considerando este quadro, torna-se cada vez mais importante a gerência do ambiente de redes de computadores para mantê-lo funcionando de forma adequada.

Dependendo da ênfase atribuída aos investimentos realizados no ambiente de processamento de dados, as funções de gerência de rede podem ser centralizadas no processador central ou distribuídas em diversos ambientes locais. Dependendo da heterogeneidade dos ambientes - por exemplo, um *mainframe* IBM com rede SNA e redes locais com ambiente UNIX, *Netware* da *Novell* ou LAN *Manager* da IBM/*Microsoft* -, a dificuldade para gerenciamento de rede pode ser muito grande, implicando, assim, no uso de várias ferramentas inseridas em uma estrutura complexa, com os limites de atuação definidos (se possível, padronizados) entre os componentes envolvidos.

Esta estrutura pode definir aspectos como a estratégia empregada no atendimento/chamadas dos usuários, a atuação do pessoal envolvido nas tarefas de gerenciamento de rede, os supridores de serviços (inclusive externos), entre outros. Portanto, o ponto-chave para as atividades de gerência de rede é a organização e aspectos tais como o atendimento do usuário se caracterizam como primordiais para o sucesso da estrutura. É desejável que o usuário dos serviços de rede tenha um único ponto de contato para reportar problemas e mudanças.

Os limites de atuação desta gerência devem levar em conta a amplitude desejada pelo modelo implantado na instalação que, além de operar a rede, deve envolver tarefas como:

- Controle de acesso à rede ;

- Disponibilidade e desempenho;
- Documentação de configuração ;
- Gerência de falhas;
- Outros.

A ênfase relativa atribuída em cada uma dessas categorias por uma instalação depende do tamanho e complexidade da rede.

Para tal tarefa, tendo o objetivo de garantir a interoperabilidade de múltiplos e diversificados sistemas computacionais e redes de computadores, a ISO (*International Organization for Standardization*) desenvolve seus padrões de gerenciamento. Neste contexto, a ISO dividiu a atividade de gerenciamento em cinco áreas funcionais específicas (*SMFA´s - Specific Management Functional Areas*): Gerenciamento de Configuração, Gerenciamento de Falhas, Gerenciamento de Contabilização, Gerenciamento de Desempenho e Gerenciamento de Segurança. Dentro de cada Área Funcional foram desenvolvidos padrões de funções (incluindo requisitos, modelos e serviços) para o gerenciamento das redes. Essas funções são processos de aplicação de gerenciamento que utilizam os serviços oferecidos pela camada de aplicação.

Com o passar dos anos foram criadas muitas padronizações, protocolos e sistemas de gerenciamento, sempre objetivando a melhor forma de gerenciamento [FES99]. Neste capítulo serão apresentadas as áreas funcionais de gerência, juntamente com algumas das tecnologias e padronizações existentes no mercado.

2.1 Gerenciamento de Configuração

A função do Gerenciamento de Configuração envolve a manutenção e monitoração da estrutura física e lógica da rede, incluindo a existência de componentes e sua interconectividade. Corresponde ao conjunto de processos que exercem o controle sobre os objetos gerenciados, identificando-os, coletando e fornecendo dados sobre os mesmos a fim de dar suporte a funções de [BOU02]:

- atribuição de valores iniciais aos parâmetros do sistema;
- início e encerramento de operações sobre objetos gerenciados;
- alteração da configuração do sistema;
- associação de nomes a conjuntos de objetos gerenciados.

O objetivo principal do Gerenciamento de Configuração é manter um registro detalhado das configurações de rede antigas, atuais e propostas. Dependendo do ambiente, esse conhecimento detalhado pode compreender uma lista muito grande de informações sobre configuração.

Documentar a configuração de rede irá auxiliar o administrador a entender os efeitos das alterações e falhas da mesma. Como todas as redes necessitam de manutenção, é fundamental ter uma visão completa das mesmas.

2.2 Gerenciamento de Falhas

O Gerenciamento de Falhas é responsável pela manutenção e monitoração do estado de cada um dos objetos gerenciados e pelas ações necessárias ao restabelecimento ou isolamento das unidades com problemas [BOU02]. As informações coletadas podem ser usadas em conjunto com o mapa da rede, para indicar quais elementos da rede estão funcionando, quais operam precariamente ou quais permanecem fora de operação.

Também é possível que o Gerenciamento de Falhas gere um registro das ocorrências, um diagnóstico de falhas e uma correlação entre os resultados do diagnóstico, com as subseqüentes ações de reparo.

O Gerenciamento de Falhas utiliza hardware e software para alertar os gerentes a respeito de falhas, auxiliando também nos seus respectivos reparos. Também podem ser utilizados hardware e software de tolerância a falhas ou redundantes, que podem continuar a fornecer serviços de rede mesmo quando ocorrer falha.

2.3 Gerenciamento de Contabilização

O Gerenciamento de Contabilização preocupa-se com a manutenção e monitoração de recursos, verificando quais e quanto desses recursos estão sendo utilizados. Estas informações podem ser utilizadas para estatísticas e/ou para faturamento.

As informações colhidas pelo Gerenciamento de Contabilização podem ser utilizadas para alocar novos recursos na rede ou simplesmente para planejar melhorias.

Utilizando o Gerenciamento de Contabilização pode-se compreender os custos reais da rede, definir suas capacidades e estabelecer políticas e procedimentos para torná-la mais eficiente

2.4 Gerenciamento de Desempenho

Enquanto o Gerenciamento de Falhas é principalmente reativo, o Gerenciamento de Desempenho é ativo, envolvendo a coleta e interpretação das medições periódicas dos indicadores de desempenho, identificando gargalos, avaliando tendências, e fazendo previsões do desempenho futuro da rede.

O Gerenciamento de Desempenho preocupa-se com o desempenho corrente da rede, incluindo parâmetros estatísticos tais como atrasos, vazão, disponibilidade e número de retransmissões [BOU02]. Consiste em um conjunto de funções responsáveis por manter e examinar registros com histórico dos estados do sistema para fins de planejamento e análise.

2.5 Gerenciamento de Segurança

Gerenciamento de Segurança refere-se à proteção e controle de acesso das informações de gerenciamento [BOU02]. Isso pode envolver a geração, distribuição e armazenamento de chaves de criptografia. Senhas, autorizações e outros controles de acesso devem ser mantidos de forma a proteger qualquer informação de gerenciamento de cada elemento da rede.

O Gerenciamento de Segurança aborda os aspectos de segurança essenciais para operar uma rede corretamente e proteger os objetos gerenciados. O sistema de gerenciamento deve providenciar alarmes para o administrador da rede quando eventos relativos a segurança forem detectados.

A tarefa de Gerenciamento de Segurança pode abranger o seguinte:

- Identificar os riscos de segurança e suas conseqüências.
- Implementar projetos e equipamentos de rede seguros.
- Administrar grupos e senhas de usuários.
- Usar equipamentos de monitoração da rede para registrar o uso, relatar violações ou fornecer alarmes para atividades de alto risco.

2.6 Modelo de Gerência SNMP

O SNMP foi desenvolvido nos anos 80 como resposta para os problemas de gerenciamento em ambiente TCP/IP Internet, envolvendo redes heterogêneas. Inicialmente foi concebido para ser apenas uma solução provisória até o completo desenvolvimento de um protocolo de gerenciamento mais completo, o CMIP (*Common Management Information Protocol*), que é o protocolo de gerenciamento baseado na arquitetura OSI. Esta característica de simplicidade, somada ao fato de que o protocolo já supria as principais necessidades no gerenciamento de redes, fez com que o SNMP

passasse a ser largamente utilizado, sendo hoje a base do gerenciamento de redes de computadores [MAL01].

O *Simple Network Management Protocol* (SNMP) é um protocolo da camada de aplicação, como mostrado na Figura 2.1, desenvolvido para facilitar a troca de informações de gerenciamento entre dispositivos de rede. Estas informações transportadas pelo SNMP, como pacotes por segundo e taxa de erro na rede, permitem que os administradores da rede gerenciem o desempenho da rede de forma remota, encontrando e solucionando problemas, bem como planejando o crescimento da mesma.

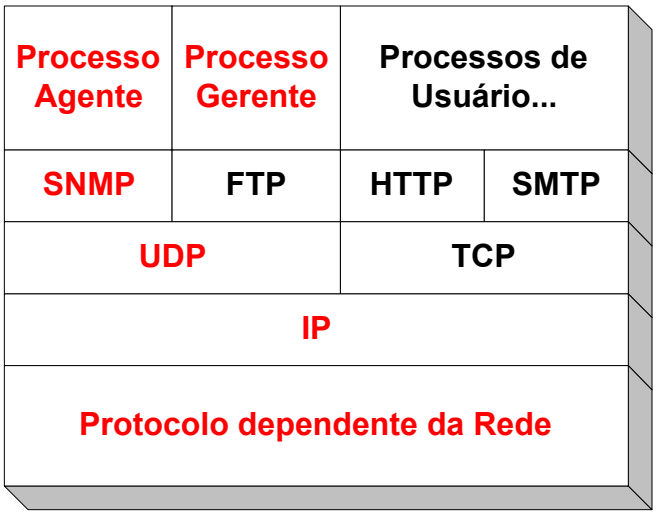


Figura 2.1 - Protocolo SNMP sobre as camadas do TCP/IP

Basicamente, o SNMP provê um padrão de *framework* para definição de informações gerenciáveis com um protocolo para fazer a troca dessas informações. O modelo SNMP assume a existência de agentes e gerentes, como pode ser visto na figura 2.2. Um gerente é o módulo do *software* responsável pelo gerenciamento das informações dos equipamentos de uma rede. O agente é o módulo responsável por gerenciar as informações localmente em um dispositivo e enviá-las ao gerente via SNMP. A troca da informação gerenciada pode ser inicializada de duas formas: pelo gerente, através do *polling,* ou pelo agente, através do *trap* .

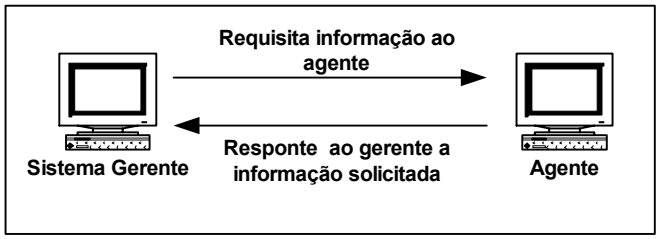


Figura 2.2 - Interação Agente/Gerente

2.6.1 Trabalhos que utilizam o protocolo SNMP

Devido às suas características, o SNMP é a melhor opção a ser usada em certos ambientes. Nesta sessão, serão apresentados trabalhos que exemplificam esta situação.

Em [MAL01], é apresentada uma proposta de gerência de redes móveis através do protocolo SNMP. A escolha deste protocolo foi feito após comparações com outras tecnologias de gerência, como por exemplo CORBA. Diante das características de uma rede sem fio, o SNMP se mostrou a melhor opção principalmente pela sua simplicidade e funcionalidades como a geração de *trap,* onde a grande vantagem é de não ter a necessidade de utilizar o *pooling* para recuperar informações de gerência, o que causaria muito tráfego, prejudicando a performance da rede.

Na implementação foi utilizado o modelo COM, que oferece também uma maior simplicidade, reusabilidade, flexibilidade entre outros benefícios. De acordo com [MAL01], COM+ e SNMP juntos formam uma sólida tecnologia para a criação de modernos e flexíveis sistemas de gerenciamento baseados em componentes.

No caso de [HON01], opção pelo protocolo SNMP é pelo fato deste ser o mais usado no gerenciamento de sistemas. Conseqüentemente a maioria dos dispositivos a serem gerenciados já vêm com suporte ao SNMP. De acordo com sua proposta a junção com a WEB para a integração em dispositivos embutidos, podem tornar os produtos de diferentes fabricantes muito mais competitivos.

2.7 Iniciativa WBEM para gerência de redes

O WBEM (*Web-Based Enterprise Management*) é uma iniciativa da DMTF (*Distributed Management Task Force*) para a unificação de mecanismos que descrevam e compartilhem informações de gerência. Essa iniciativa foi patrocinada por companhias como a BMC Software, Cisco Systems, Compaq Computer, Intel e Microsoft [THO98]. Hoje em dia, muitas companhias já suportam essa funcionalidade, como Computer Associates, IBM/Tivoli e HP.

O WBEM define um mecanismo comum para o compartilhamento das informações de gerência, mas não define as regras de como os fabricantes devem implementar suas soluções. Para isso, foi criada uma série de tecnologias padronizadas de gerência e *Internet*, desenvolvidas para unificar o gerenciamento de ambientes de computadores, permitindo que a industria desenvolva soluções de gerência padronizadas e baseadas nas tecnologias WEB emergentes.

O modelo de informação é capaz de descrever qualquer ambiente de gerência existente. O sistema foi descrito de uma maneira compatível com protocolos de gerência existentes, incluindo o SNMP, *Definition of Managemnt Information* (DMI) *e Common Management Information Protocol* (CMIP). A intenção desta iniciativa não é tirar a posição já consagrada destes protocolos, mas sim coexistir com os mesmos e até fazer a extensão do WBEM para fazer a unificação destas tecnologias [THO98].

A base de padrões que constituem o WBEM inclui:

- O modelo de dados: o padrão *Common Information Model* (CIM)
- A especificação: xmlCIM *Encoding Speficication*
- Mecanismo de transporte: CIM *Operations over* HTTP

O CIM é a linguagem e a metodologia para descrever os dados de gerência. O CIM *schema* inclui modelos para sistemas, aplicações, redes locais (LAN) e dispositivos, permitindo que aplicações de diferentes desenvolvedores em diferentes plataformas descrevam dados de gerência em um formato padrão, que podem ser

compartilhados por diferentes aplicações de gerência. O xmlCIM *Encoding Specification* define os elementos XML, escritos em um DTD (*Document Type Definition*), e pode ser usado para representar as Classes CIM e suas instâncias. O CIM *Operations over* HTTP define o mapeamento das operações CIM em HTML, fazendo com que implementações do CIM tenham interoperabilidade de forma aberta e padronizada, completando, assim, as tecnologias que suportam o WBEM.

Através desta estrutura, que pode ser considerada como um *framework*[2] extensível para diversas outras implementações, torna-se possível a utilização de diversas tecnologias, como por exemplo CORBA/Java, que pode estender esta mesma estrutura para um objetivo próprio. Exemplo disso é o caso da implementação do SUMO [BEN00], que cria um *framework* baseado no WBEM, utilizando o *middlewere* CORBA para o gerenciamento dos seus recursos.

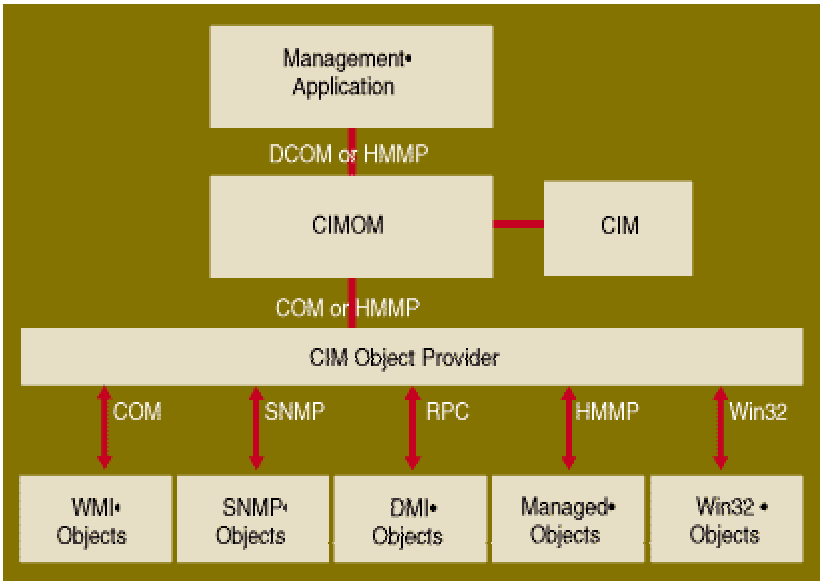


Figura 2.3 - Arquitetura WBEM [DMT02]

Na figura 2.3 é apresentada a arquitetura básica do WBEM. Na base da arquitetura estão as diferentes origens de dados gerenciais, que são integrados através

[2] *Frameworks* são estruturas de classes que constituem implementações incompletas que estendidas permitem produzir diferentes artefatos de software. A grande vantagem desta abordagem é a promoção do reuso de código e projeto, que pode diminuir o tempo e o esforço exigidos na produção de software [SIL00].

do CIM *Object Provider*. A comunicação entre o CIM e a aplicação de gerência é responsabilidade do CIMOM (CIM *Object Manager*), que age como uma camada tradutora de informação entre estes diferentes blocos.

2.7.1 CIM – Common Information Model

O CIM representa o modelo de todos os tipos de objetos físicos e lógicos em um ambiente de gerência. Os objetos gerenciados são representados utilizando o conceito de orientação a objetos, através de classes. Estas classes incluem propriedades que descrevem os dados e métodos que descrevem o comportamento dos objetos. O CIM define 3 níveis de modelos [BEN00]:

- ***Core Model*** - é formado por algumas classes, associações e propriedades que contêm o vocabulário básico para analisar e descrever sistemas em todas as áreas de gerenciamento.
- ***Common Models*** - esta classe procura englobar características de áreas comuns de gerenciamento, mas completamente independentes de uma tecnologia ou plataforma em particular. A informação é específica o suficiente para permitir o desenvolvimento das aplicações de gerência.
- ***Extension Models*** – representam objetos gerenciados com a adição de uma tecnologia específica aos *Common Models*. Geralmente são utilizados em plataformas específicas como UNIX ou em ambientes Microsoft Win32.

A partir da padronização de cada modelo, os fabricantes podem desenvolver, a partir da extensão destes, suas próprias ferramentas de gerência, como é o caso do WMI da Microsoft, que desenvolveu classes como Win32_Processor, para monitorar o processador, ou Win32_LogicalDisck, para a monitoração de discos, físicos ou lógicos, entre muitas outras classes.

As classes do CIM são definidas por uma linguagem especial chamada MOF - *Managed Object Format*. O MOF é uma linguagem compilada baseada na *Interface Definition Language* (IDL). É através desta linguagem que são definidas as classes para cada tipo de tecnologia. Por exemplo, quando a Microsoft define a classe Win32_Processor, todas as propriedades e métodos para o gerenciamento do processador são definidos usando esta linguagem.

2.7.2 WMI – Windows Management Instrumentation

O WMI é implementação da Microsoft para o WBEM que utiliza o *Common Information Model* – CIM, para representar objetos gerenciáveis em ambientes *Windows*. O WMI utiliza a sintaxe MOF – *Managed Object Format* para definir a estrutura e conteúdo armazenados no CIM, fazendo com que o acesso as informações de gerenciamento sejam simplificadas podendo ser acessadas através de componentes (COM – *Componente Object Model*) ou linguagens script, como VBScript, Javascript ou Windows Script.

Os principais componentes da arquitetura WMI incluem o CIM, CIM *Object Manager* (CIMON), *providers,* os objetos gerenciáveis e as aplicações de gerência [MIC00]:

- **Aplicações de gerência:** são aplicações windows ou serviços windows NT/2000 que processam ou mostram dados de objetos gerenciados. Uma aplicação de gerência pode fazer várias tarefas como medição de performance, alertar para possíveis paradas do sistema, analise de performance.
- **Objetos Gerenciados:** são componentes físicos ou lógicos operando em um ambiente. Os objetos gerenciados são modelados usando o CIM e são acessados por aplicações de gerência usando o WMI. Um objeto gerenciado pode ser qualquer componente do sistema de um pequeno hardware, como o *cooler* da máquina até uma aplicação grande como um sistema de banco de dados.
- ***Providers:*** são DLL's ou executáveis (EXE) que servem a camada de abstração entre o objeto gerenciado e a infraestrutura WMI. Os *Providers* escondem os detalhes de implementação que são únicos pra uma metodologia ou protocolo de gerência, como por exemplo APIs Win32, Windows Driver Model (WDM), SNMP, Desktop Management Interface(DMI). Um *provider*

utiliza API's nativas para fazer a comunicação com os objetos e uma interface WMI para se comunicar com o CIMON (CIM Object Manager).

- **Infraestrutura de gerenciamento CIMON (*CIM Object Manager*):** Consiste no *software* WMI e o repositório CIM **.** O WMI permite que aplicações de gerenciamento e *providers* se comuniquem. Essa comunicação é feita através de COM APIs, que por sua vez acessam as informações armazenadas estática ou dinamicamente no CIM .

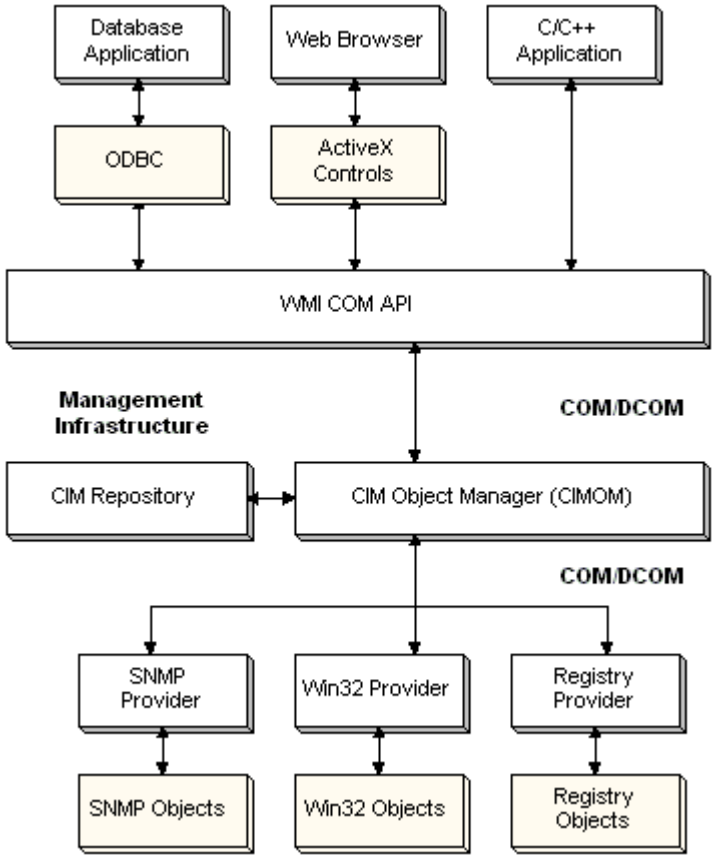


Figura 2.4 - Arquitetura WMI [MIC00]

A figura 2.4 mostra o relacionamento dos componentes da tecnologia WMI. No topo, aparecem alguns exemplos de *providers* de aplicações de gerência. Algumas aplicações acessam os COM API diretamente para interagir com o WMI e o repositório CIM, outros utilizam métodos de acesso como *Open Database Connectivity* (ODBC) e HTML para fazer essas requisições. Na parte inferior da figura, são ilustrados exemplos de objetos gerenciados e seus respectivos *providers* .

2.7.3 Trabalhos relacionados à iniciativa WBEM

A utilização da padronização criada pela iniciativa WBEM tem sido considerada uma boa opção principalmente para ambientes que operam com tecnologias variadas de gerenciamento. Isto se deve ao fato da arquitetura WBEM disponibilizar meios que facilitam a integração de tecnologias e padrões de gerência.

Um exemplo desta integração pode ser visto em [BEN00] que apresenta uma proposta de *framework* para o gerenciamento global e distribuído de sistemas envolvendo tecnologias orientadas a objetos como:

- Arquitetura WBEM
- Modelo de informações gerenciáveis CIM
- CORBA como intraestrutura de comunicação
- Java como linguagem de programação

A escolha dessas tecnologias foi resultado de uma pesquisa onde certos pré-requisitos deveriam ser atendidos, como homogeneização de sistemas de gerenciamento, aumento da competitividade através da utilização de tecnologias padrão, reuso e modularização (utilização de componentes) e integração com o legado. Para atender estes pré-requisitos, o sistema de gerência do projeto SUMO tem sua arquitetura baseada na arquitetura WBEM e a implementação das funcionalidades é feita com CORBA. Desta forma, os objetos gerenciados e os *providers* são objetos CORBA. As informações gerenciais são mapeadas através de uma implementação do CIM.

Como resultado desta implementação, foi criado um *framework* para gerenciamento de sistemas simples de utilizar e configurar devida a utilização do modelo CIM. Este *framework* oferece serviços que dificilmente seriam encontrados nas soluções já existentes, especialmente no que se trata de segurança e tolerância a falhas, que foram algumas funcionalidades implementadas no projeto SUMO.

Outro exemplo é descrito em [FES99] que fez um trabalho voltado para integração de tecnologias para o gerenciamento, que no caso é a integração de tecnologias WBEM e o CIM como repositório de informações gerenciáveis, juntamente como gerenciamento OSI.

A necessidade desta integração surge quando a rede interna é gerenciada por agentes WBEM mas *links* de longas distâncias são gerenciadas por sistemas baseados em TMN. Para isso [FES99] propõe um mapeamento de WBEM para TMN, resultando no Q-Adapter, que é capaz de gerenciar o ambiente OSI com agentes WBEM. O Q-Adapter pôde ser criado pela facilidade de mapeamento proporcionada pelo modelo CIM e pela proximidade das duas formas de gerenciamento.

Já [WUN02] faz um estudo sobre as metodologias de gerenciamento de redes. A base desta pesquisa é a utilização do WBEM como estratégia de gerenciamento, tendo em vista que a WEB já controla boa parte do comércio e atividades acadêmicas, tornando lógica a utilização de tecnologias baseadas na WEB, como o WBEM.

Esta opção tornou-se adequada pelas características do modelo CIM, que tem boa escalabilidade, arquitetura e configuração genérica, performance e é destinado a integração com outras tecnologias.

Segundo [WUN02], há uma necessidade de integração no gerenciamento de redes corporativas e domésticas. Tal nível de integração é possibilitada através de uma implementação com tecnologias genéricas como WBEM, CORBA, SNMP.

2.8 Trabalhos/Ferramentas de gerência existentes

A gerência de redes é uma tarefa complexa e, devido às novas tecnologias emergentes e à heterogeneidade existente num mesmo ambiente, está sempre em crescimento. Para auxiliar nesta tarefa existem várias ferramentas que objetivam gerenciar redes. Neste capítulo serão apresentadas algumas ferramentas e suas principais características, juntamente com os trabalhos que vem sendo feitos na área de gerência

2.8.1 Netview

O Tivoli Netview é uma solução bastante robusta apresentada pela IBM para o gerenciamento de redes TCP/IP.

Após a instalação, o gerente da rede pode gerar um mapa com todos os nós da sua rede a partir do *auto Discovery* do Netview e, a partir desse mapa, gerenciar graficamente todos os nós da rede [IBM01]. Com os nós que tiverem o agente SNMP instalado, é possível monitorar todas as variáveis da MIB II, e se existir, utilizar-se das MIB´s proprietárias para ter mais detalhes do nó. Dentre as principais características dessa ferramenta estão:

- Interface Gráfica amigável;
- Mapeamento automático da rede;
- Isolamento de falha de roteador, reduzindo, assim, o tempo de resolução do problema;
- Interface WEB, que permite a visualização os nós da rede e fazer o gerenciamento dos mesmos a partir de qualquer computador ligado à rede;
- Permite a correlação e gerenciamento de eventos e *traps* SNMP;
- Gerenciar a performance dos nós da rede;
- A nova versão já permite que possam ser gerenciados os nós que têm o WBEM instalado;
- Entre muitas outras características gerenciais importantes.

O NetView é uma ferramenta completa para gerenciamento de grandes e complexas redes heterogêneas. É importante para grandes empresas, universidades e qualquer outra grande instituição que precise garantir um bom funcionamento de suas redes e, para isso, podem investir em uma aplicação robusta, com um preço relativamente elevado e treinamento de pessoa qualificada para a utilização da mesma .

2.8.2 WhatsUp

O WhatsUp é uma poderosa e intuitiva ferramenta que oferece aos administradores de rede maior controle e entendimento da rede, facilitando a missão de manter a rede funcionando adequadamente [WHA01]. O WhatsUp é fornecido pela Ipswitch e, apesar de ser uma ferramenta com um preço mais acessível que o NetView, ainda possui um custo relativamente elevado.

Por ser uma ferramenta gráfica, oferece facilidades no gerenciamento da rede, pois através de mapas que são descobertos automaticamente, pode-se gerenciar toda uma rede de maneira centralizada, em um único computador. Suas principais características são :

- Mapeamento da rede: a ferramenta tem uma função para mapear os nós da rede tais como roteadores, servidores, estações de trabalho e *switches*, de forma a apresentar uma interface simples e amigável para quem estiver administrando a rede;
- Monitoramento de serviços e dispositivos: inicializa a auditoria e visualização de alarmes quando dispositivos e serviços do sistema apresentam problemas.
- Mecanismo de notificação: quando é detectado algum problema, são enviadas notificações via SMS, e-mail, telefone e/ou *Pager*, configuradas para cada dispositivo em um mapa*;*
- Medição da performance do sistema como geração de gráficos de utilização de CPU e geração de relatórios para análise da utilização da rede;
- Acesso via Interface WEB, que permite que sejam visualizadas remotamente as mesmas informações apresentadas pelo console;

Assim como o NetView, a utilização desse tipo de ferramenta é de interesse para grandes instituições, onde os nós da rede são bastante espalhados e torna-se importante a centralização da administração. Além do investimento com a ferramenta, também seria necessário investir em um gerente treinado, para fazer o melhor uso da mesma.

2.8.3 NTOP

NTOP é uma ferramenta de gerência que permite a verificação de alguns comportamentos da rede que podem vir a gerar falhas. Suas principais funcionalidades são [NTO02] :

- Medida de tráfego: faz a medição de banda utilizada, dados recebidos e enviados, estatística do fluxo da rede, entre outras métricas;
- Detecção de violação de segurança na rede;
- Monitoramento de tráfego: faz a monitoração de mau comportamento do trafego da rede, como utilização de IPs duplicados, mau uso de protocolos, excesso de utilização de largura de banda;
- Otimização e planejamento de rede;

O NTOP é uma ferramenta que pode ser utilizada via também via WEB e uma interface WAP. Basicamente ela oferece uma forma de monitorar tráfego, utilização de banda na rede. Ele não utiliza nenhum tipo de notificação de falhas, é possível apenas uma analise para a prevenção de erros.

O NTOP é uma ferramenta gratuita e aconselhável para instituições que tem problemas com largura de banda ou superutilização de protocolos.

2.8.4 Trabalhos Realizados na Área de Gerência de Redes

Os trabalhos na área de gerência de redes, surgem de acordo com as necessidades de um ambiente, ou a ascensão de novas tecnologias que podem vir a melhorar aplicações já existentes. Neste capítulo serão apresentados alguns trabalhos que foram feitos com base nestas premissas. Como [VOU01] que descreve um trabalho de implementação de uma aplicação que é executada em dispositivos móveis como

celulares e *Palms*. Esta aplicação tem como objetivo, o gerenciamento de falhas de uma rede através de envio de comandos de gerenciamento dentro de mensagens SMS. Como exemplo, ela pode através do celular, após a detecção de uma falha, reiniciar uma máquina remotamente. Como continuidade deste trabalho, é proposto a união com a tecnologia WAP como mecanismo de comunicação.

Já [GUI01] propõe a implementação de uma API Java para gerenciamento de falhas de um sistema, baseado no JMX (*Java Management Extension*), que foi desenvolvido pela *Sun* para prover uma forma padronizada para o gerenciamento de recursos e para desenvolvimento de agentes dinâmicos para gerir esses recursos. A API proposta por [GUI01], tem como principais funcionalidades o monitoramento da performance da rede e detecção de falhas. É um trabalho que pretende ter uma continuidade com a implementação do gerenciamento de alarmes, localização da falha e resolução da mesma.

No caso de [KRI01], é apresentada uma ferramenta para geração de alarme baseado na performance de uma rede. NwsAlarm é uma ferramenta que faz leituras da performance em relação a largura de banda, latência, CPU entre outras variáveis. Ao detectar uma queda de performance, automaticamente é enviada uma notificação via *e-mail* ao administrador da rede. Os valores limites de performance são configuráveis no sistema. NwsAlarm também permite que a partir de um histórico armazenado, seja gerado, através de cálculos matemáticos, uma estimativa futura de performance. Essa informação também pode ser visualizada de maneira gráfica. NwsAlarm é desenvolvida em *applets* Java, e é de extrema importância em um ambiente que execute aplicações que necessitem rodar um ambiente de alta performance.

Também na área de gerência, são feitos estudos da melhor forma de apresentação das informações gerenciadas ao usuário. Atualmente, muitas aplicações são viabilizadas através da WEB. A gerência de redes não poderia ser diferente. Muitas são as vantagens em ter a WEB como forma de interface nas aplicações voltadas para o gerenciamento como pode ser visto em [MAC01], que reporta a experiência da introdução de interfaces WEB em plataformas de gerenciamento distribuídas provendo uma forma consistente e uniforme para acessos a serviços como gerenciamento de falhas, gerenciamento de performance, gerenciamento de negócios e de serviços. O uso de interfaces WEB melhorou de forma significante os serviços prestados aos clientes.

2.9 Conclusão

Com crescimento das redes de computadores, fez-se necessária uma organização na forma de gerenciamento. Com este intuito, a ISO dividiu a atividade de gerenciamento em cinco áreas funcionais Gerenciamento de Configuração, Gerenciamento de Falhas, Gerenciamento de Contabilização, Gerenciamento de Desempenho e Gerenciamento de Segurança. Dentro de cada Área Funcional foram desenvolvidos padrões de funções (incluindo requisitos, modelos e serviços) para o gerenciamento das redes. Essas funções são processos de aplicação de gerenciamento que utilizam os serviços oferecidos pela camada de aplicação. Dentre os padrões de gerenciamento existentes, podem ser citados como exemplos os protocolos SNMP e o WBEM como uma arquitetura padrão para desenvolvimento de aplicações de gerência..

O *Simple Network Management Protocol* (SNMP) é um protocolo da camada de aplicação desenvolvido para facilitar a troca de informações de gerenciamento entre dispositivos de rede. Devido a sua abrangência de funcionalidades de maneira simplificada, o SNMP é o padrão mais utilizado para o gerenciamento de redes. Já o WBEM define um mecanismo comum para o compartilhamento das informações de gerência. Sua arquitetura provê meios de unificação do gerenciamento de ambientes de computadores, permitindo que se desenvolva soluções de gerência padronizadas e baseadas nas tecnologias WEB emergentes. O modelo de informação WBEM é capaz de descrever qualquer ambiente de gerência existente, e até ser integrado com protocolos de gerência existentes, incluindo o SNMP, DMI, CMIP

Esses dois padrões são focos de vários estudos dentro da área de gerência de redes. No caso do SNMP algumas vezes torna-se a melhor solução a ser utilizada para determinados ambientes, por esse motivo, muitos trabalhos feitos na área de gerencia de rede, adotam o SNMP como padrão obrigatório de gerenciamento. Da mesma forma o WBEM que pelas suas características de integração fazem com que vários trabalhos utilizem este padrão, pois com o mesmo, é possível criar uma aplicação capaz de gerenciar ambientes que necessitem de diferentes padrões de gerência.

3 Tecnologias Utilizadas Para a Aplicação

3.1 *Wireless Aplication Protocol* - WAP

É um protocolo de aplicação destinado a redes que operam sem a utilização de cabos (redes *wireless*), permitindo a interação de aplicações e serviços entre redes móveis e a *Internet*.

O WAP foi desenvolvido pela *Phone.com,* juntamente com a Nokia, Ericsson e Motorola. Como suas especificações são abertas, ou seja, qualquer pessoa ou empresa pode desenvolver recursos para terminais móveis como celulares, PAD's, *Pagers*, entre outros, foi estabelecido o órgão WAP Fórum para administrar e cuidar da sua padronização. As especificações são amplas e atingem todas as tecnologias de redes sem fio e as tecnologias Internet. A meta é dar suporte tecnológico aos operadores, fabricantes e desenvolvedores, permitido-os a execução e a implementação de serviços na rede móvel, de maneira rápida e flexível.

O protocolo WAP permite o acesso, via celular, do conteúdo da Internet através de um *Gateway*, que busca a informação da WEB e entrega para o celular (WAP *Browser*). Para acessar um *site* WAP, é necessário que o telefone celular seja compatível com a tecnologia WAP e as informações devem estar no formato WML (*Wireless Markup Language*). Existe também o *WMLScript* para WML, semelhante ao *JavaScript* para HTML.

A especificação da arquitetura do WAP também atua como modelo para a compreensão das tecnologias de redes móveis, servindo de referência e especificação de novos detalhes técnicos adicionais [WAP01].

Com o crescimento atual do mercado em relação a tecnologia *Wireless*, os desenvolvedores de aplicações, devem estar atentos na utilização deste protocolo em suas soluções [LEE01].

Para um melhor entendimento de como implementar uma aplicação utilizando o protocolo WAP, este capítulo mostrará uma comparação entre o modelo WWW e o modelo WAP, seguido da descrição de sua arquitetura

3.1.1 Arquitetura WAP

O Conceito da formulação do protocolo WAP, é a de ter a grande proximidade das camadas da *Internet* padrão. Com o diferencial de que o WAP, como seu próprio nome diz, é destinado a ser utilizado em uma rede sem fios, enfrentando empecilhos como baixa largura de banda, condições de alta latência, taxa de erro elevada e com relação aos terminais móveis, interfaces limitadas, baixo poder de processamento e pouca memória.

Para enfrentar tais empecilhos a pilha de protocolo definida no WAP aperfeiçoa os padrões de protocolos WEB, como HTTP. Alguns desses aperfeiçoamentos são:

- Os textos de cabeçalho do HTTP são traduzidos em código binário, o que reduz a quantidade de dados que devem ser transmitidos.
- Um protocolo leve de restabelecimento de sessão é definido para permitir que sessões possam ser suspendidas e restauradas sem o *overhead* de estabelecimento inicial . Isto permite suspender uma sessão, enquanto se espera por recursos da rede ou poupa potência da bateria.
- WAP provê um serviço de pacote de dados confiável através do Protocolo de Transação Sem Fio (WTP), que mantém algumas características importantes do TCP tradicional, mas sem o comportamento que o faz inadequado nas redes sem fio. Por exemplo, TCP transmite uma grande quantidade de informação para cada transação de pedido e resposta, incluindo informações necessárias ao controle de entrega de pacote fora de ordem. No WAP só há uma rota possível entre o *Gateway/proxy* WAP e o dispositivo portátil, então não há nenhuma necessidade de controle desta situação. O WTP elimina esta informação

desnecessária e reduz a quantia de informação para cada transação de pedido e resposta.

As melhorias feitas na pilha de protocolo WAP significaram uma economia na largura de banda sem fio. Este utiliza menos que a metade do número de pacotes que são necessários para disponibilizar um conteúdo com protocolos HTTP/TCP/IP. Esta melhoria é essencial para melhor utilização da largura de banda que é limitada nas redes sem fio.

Ainda em decorrência da natureza das redes *wireless*, algumas características úteis de quando se está programando para WEB, são bastante prejudicadas utilizando-se WAP. Mas a demanda dos usuários móveis tem tido um crescimento considerável [STU002], portanto existem muitos estudos com o intuito de encontrar soluções para problemas como segurança nas transações, armazenamento de valores em variáveis de sessão[3].

A questão da segurança é apresentada por [THA00] como não suficiente para certos tipos de aplicações como transações bancárias e pagamentos via dispositivos *wireless,* e partindo deste problema, apresenta uma solução com a implementação de um *proxy* para utilização em transações que necessitem de maior segurança.

Já para o armazenamento de variáveis de sessão, que são constantemente utilizadas na programação WEB para validação de sessão de usuários, *cookies,* entre outros, fica bastante debilitado devido ao ambiente de intensa perda de conexão causadas pelos problemas já citados [HAG02]. Para esse tipo de situação, [CAN01] propõem uma solução através ambiente componentizado que implementa uma aplicação WAP, que visa tratar o problema do uso de sessões perante estas várias perdas de conexão.

Muitos outros trabalhos nesta área visam melhorar a viabilização das aplicações que é o caso de [OJA00], que realizou uma pesquisa na tentativa de melhorar a performance da transmissão através da compressão dos dados de um arquivo WML. Mas esse tipo de solução ainda não foi aperfeiçoada, podendo tornar as perdas mais

[3] Variáveis de sessão são aquelas que armazenam valores de uma determinada sessão criada pelo Browser. Assim que a janela do browser é fechada, a variável é perdida. Os Cookies, são exemplos de variáveis de sessão.

significativas devido ao tempo e processamento de comprimir/descomprimir os arquivos.

Já não é novidade que existem atualizações da versão atual da pilha de protocolos que visam a melhora do mesmo, que é o caso do WAP 2.0, terminado em 2001, mas os celulares mais recentes utilizam a versão 1.x e ainda será utilizado por anos [STU02].

3.1.2 Componentes da Arquitetura WAP

Os componentes da arquitetura de WAP criam um ambiente escalonável e extensível para o desenvolvimento de aplicações voltados para o ambiente de comunicação *wireless*. Cada camada da arquitetura é acessada pelas camadas imediatamente superiores, ver figura 3.3, assim com por outros serviços e aplicações.

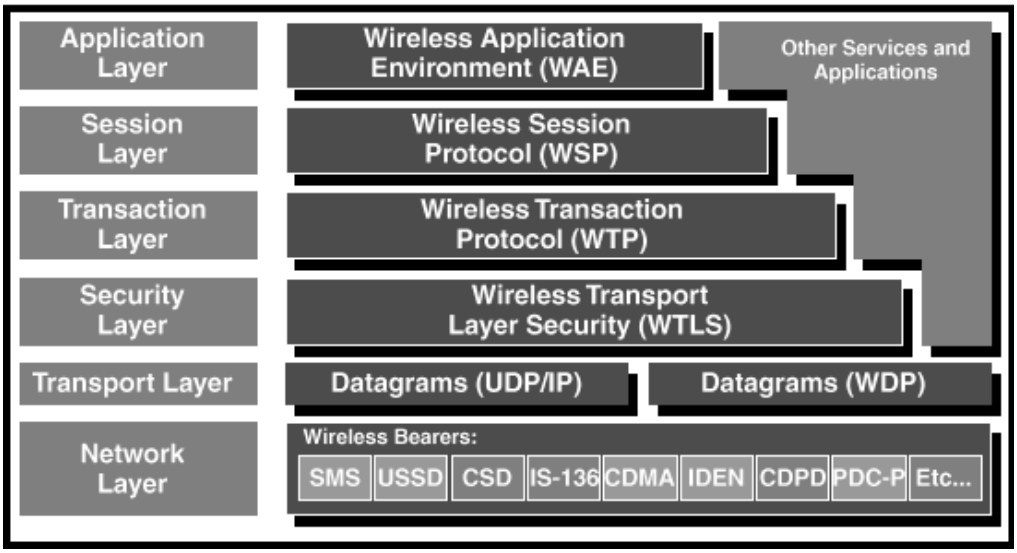


Figura 3.1 - Camadas do Protocolo WAP [WAP01]

Utilizando-se de princípios de interoperabilidade entre camadas, já consagrados por outros modelos recomendados, a arquitetura oferece as aplicações externas, os

mecanismos necessários para estabelecimento de uma sessão, transação, política de segurança e de transporte .

As camadas da arquitetura WAP são:

- Camada de Aplicação (WAE)
- Camada de Sessão (WSP)
- Camada de Transação (WTP)
- Camada de Segurança (WTLS)
- Camada de Transporte (WDP)
- *Bearers* (Serviços de Entrega)

Wireless Aplication Environment - WAE, fornece estrutura para o ambiente de interação entre o modelo WWW e tecnologias dos dispositivos móveis Todo conteúdo é especificado em formatos que são semelhantes aos formatos padrões da Internet. O conteúdo é transportado usando protocolos padrões no domínio WWW, e no domínio *wireless* são usados os protocolos WAP.

O WAE assume a existência de um *Gateway* responsável em codificar e decodificar dados transferidos de e para o cliente móvel. O propósito da codificação do conteúdo entregue ao cliente é minimizar o tamanho dos dados enviados ao cliente, como também, minimizar a energia computacional requerida pelo cliente, para processar esses dados. A funcionalidade do *Gateway* pode ser adicionada a servidores de origem ou pode ser colocada em *Gateways* dedicados.

O objetivo da WAE é estabelecer um ambiente que permita aos operadores e os fornecedores de serviços construírem aplicações que possam alcançar uma gama de plataformas diferentes *wireless* de redes, de maneira eficiente e útil [WAP01].

A camada de sessão (WSP) fornece protocolo responsável pela garantia de estabelecimento da conexão e desconexão de uma sessão entre dois agentes (servidor e terminal), através de dois tipos de serviços:

- Orientado à conexão (confiável): opera sobre um protocolo da camada de transação.
- Sem conexão (não confiável): opera sobre um serviço de transporte de pacote de dados seguro ou não. É indicado para aplicações que não precisam de uma de entrega de dados confiável e não se importam com a confirmação.

Wireless Transaction Protocol - WTP tem o objetivo de prover transação com confiabilidade. Durante uma sessão de navegação, o cliente requisita informação de um servidor, que pode ser fixo ou móvel, e o servidor responde com a informação [WAP01]. O WTP opera sobre os serviços de datagrama, nas redes *wireless* seguras ou não.

O protocolo da camada de segurança na arquitetura WAP é chamado de camada de transporte segura sem fio, WTLS. Esta camada opera sobre o protocolo de transporte (WDP). É uma camada modular, dependente do nível de segurança exigido por determinada aplicação, que pode habilitar ou desabilitar seletivamente[WAP00b]. Os serviços que esta camada provê são :

- Integridade dos dados;
- Privacidade;
- Autenticação
- Detecção e rejeição de dados enviados incorretamente

O protocolo de transporte do WAP é um serviço de datagrama através da camada WDP, que opera acima dos serviços de transporte suportados pelas redes de comunicação de dados [WAP01]. Como qualquer camada de transporte, em geral, o WDP oferece um serviço consistente aos protocolos das camadas superiores do WAP e comunica-se de forma transparente sobre os diferentes portadores de serviços das redes sem fio, como GSM, CDMA, PHS, entre outros. Mantendo a interface da camada de transporte e as características básicas consistentes, a interoperabilidade global pode ser alcançada usando portais (*Gateways*) mediadores.

O protocolo WAP foi projetado para interagir com uma variedade de serviços diferentes, que são os Bearers, incluindo :

- Mensagens curtas (SMS – *Short Message Service);*
- Comutação de Dados pr Circuito (CSD – *Circuit-switched Data);*
- Serviço de Pacotes Via Rádio (GPRS - *General Packet Radio Service);*
- Troca de Dados em Circuito de Alta Velocidade (HSCSD – *High Speed Switch Data*), entre outros.

Este serviços apresentam diferentes requisitos quanto a qualidade, taxa de erros, retardos. A intenção do WAP é compensar ou tolerar estas variações, da maneira mais eficiente e clara possível. Na especificação da camada WDP, existe uma lista dos portadores que são suportados e as técnicas necessárias ao funcionamento de cada portador, sendo volatibilidade a principal característica desta lista, dada a evolução da tecnologia das redes *wireless* [WAP01].

3.2 SMS - Short Message Service

O *Short Message Service* é um serviço que provê um mecanismo para envio e recebimento de mensagens curtas entre dispositivos *Wireless*. Este serviço utiliza o SMSC (*Short Message Service Center*), que é um *software* residente nas operadoras, responsável pela gerência das mensagens curtas, fazendo a recepção, recebimento e entrega das mensagens entre o SME e o dispositivo móvel . Como pode ser visto na figura 3.5.

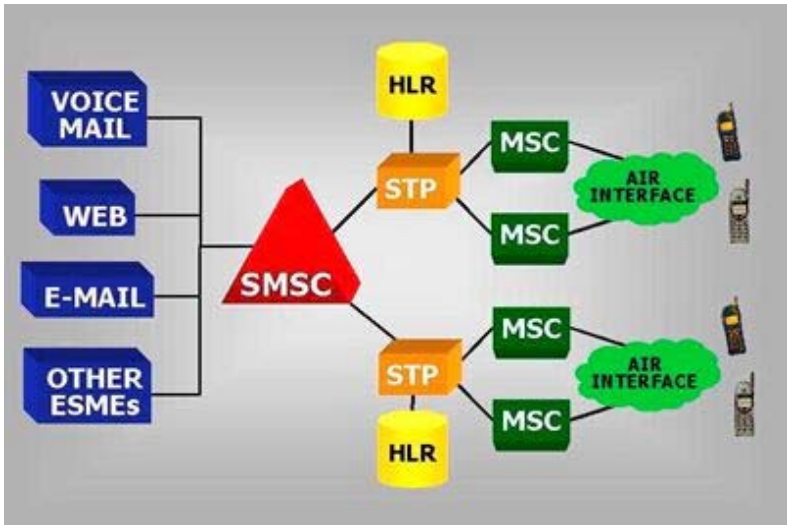


Figura 3.2 - Arquitetura SMS

A figura 3.5 representa uma arquitetura básica para um SMSC com múltiplas entradas, incluindo V*oice-Mail System* (VMS), mensagens baseadas em WEB, e-mail, e outras *External Short Message Entities* (ESMEs). A comunicação com os elementos de rede *wireless* como a *Home Location Register* (HLR) e *Mobile Switching Center* (MSC) é alcançado através do *Signal Transfer Point* (STP).

O SMS é um serviço que garante a entrega da mensagem mesmo que o destino esteja indisponível no momento em que a mesma foi enviada. Neste caso a mensagem é armazenada em um SMSC até que o destino esteja novamente disponível.

A rede *wireless* provê os mecanismos para encontrar as estações destino e transportam as mensagens entre o SMSCs e as estações *wireless*.

O SMS foi o primeiro serviço a integrar a WEB ao telefone, antes mesmo do WAP. Criado primeiramente para fazer com que dois aparelhos celulares enviassem mensagens entre si, e a operadora enviasse mensagens para seus assinantes, não foi difícil integrar a *Internet* ao serviço, onde um computador, conectado á *Internet*, envia uma mensagem para um aparelho celular.

Com as características de competitividade do mundo atual, qualquer diferenciação de serviços é relevante. O SMS é um fator de diferenciação bastante importante a oferecer a um cliente. Garantindo os seguintes benefícios:

- Pode ser usado para fazer notificações e/ou alertas;
- Entrega garantida de mensagem;
- Mecanismo confiável e de baixo custo para uma mensagem importante;
- Possibilidade de filtrar as mensagens e retorna-las, se necessário de uma maneira facilitada;
- Aumenta a produtividade do assinante;
- É uma alternativa aos serviços de *paging*

3.3 Objetos Distribuídos

O uso de objetos distribuídos tem sido rapidamente expandido devido a necessidade da busca de informações entre ambientes computacionais e também a necessidade da divisão de tarefas entre computadores, também chamada de *load balancing* [ABD02].

Na programação distribuída usando a arquitetura cliente-servidor, clientes e servidores podem ser implementados usando qualquer paradigma de programação. Assim, é possível que um serviço específico seja executado por um método de algum objeto. No entanto, mesmo que o cliente também tenha sido desenvolvido orientação a objetos, na comunicação entre o cliente e o servidor esse paradigma deve ser esquecido, devendo ser utilizado algum protocolo pré-estabelecido de troca de mensagens para a solicitação e resposta ao serviço.

Um sistema de objetos distribuídos é aquele que permite a operação com objetos remotos. Dessa forma é possível, a partir de uma aplicação cliente orientada a objetos, obter uma referência para um objeto que oferece o serviço desejado e, através dessa referência, invocar métodos desse objeto, mesmo que a instância desse objeto esteja em uma máquina diferente daquela do objeto cliente.

Um ponto comum entre as diferentes abordagens para sistemas distribuídos orientados a objetos é a existência de um elemento responsável por disponibilizar transparentemente os objetos das aplicações servidoras às aplicações clientes: O ORB (*Object Request Broker*, ou Analisador de Requisições a Objetos) permite que um cliente utilize objetos *proxies* para acessar um objeto existente em uma outra aplicação.

Os objetos são implementados segundo o conceitos da programação orientada a objetos, sendo executados em ambiente que suportam serviços como localização transparente de objetos, invocação de métodos em objetos locais ou remotos e a migração de objetos.

Atualmente existem diversas tecnologias que suportam o desenvolvimento de aplicações baseadas em objetos mas apenas duas serão citadas neste trabalho: CORBA que é a proposta do OMG e o DCOM, a solução da Microsoft .

3.3.1 CORBA (*Common Object Request Broker Architecture*)

É uma especificação desenvolvido pelo *Object Management Group* - OMG, consórcio de industrias responsáveis por publicar padronizações de uma estrutura de alto nível para computação distribuída para que as várias implementações sejam compatíveis umas com as outras. Hoje em dia, é uma das arquiteturas de *middlewere* mais populares em utilização [ABD02].

A utilização do CORBA faz com que o sistema inteiro seja auto descritivo, pois a especificação dos serviços é feita de forma separada da implementação, o que permite a incorporação de sistemas existentes, independência de plataforma e de linguagem de programação.

Um objeto distribuído CORBA pode estar em qualquer lugar da rede, sendo acessado por clientes remotos através de métodos de invocação. Tanto a linguagem quanto o compilador utilizado para a geração do código do objeto servidor são totalmente transparentes para o cliente. O cliente não precisa saber onde o objeto distribuído está localizado, qual sistema operacional está sendo utilizado para executá-lo ou em que linguagem foi escrito, a única coisa que o cliente necessita saber é a interface de acesso fornecida pelo servidor.

Object Management Architecture (OMA) é a arquitetura geral dos componentes necessários para desenvolver este ambiente portável e interoperável, isto inclui a necessidade de interfaces orientadas à objetos, transparência de distribuição, uma forma comum de modelar objetos, uma base comum para todos os componentes, suporte total para todos os estágios do ciclo de vida do software, uma natureza flexível e dinâmica, alta performance, implementações robustas, e compatibilidade com padrões existentes dentro da indústria de software[OMG02].

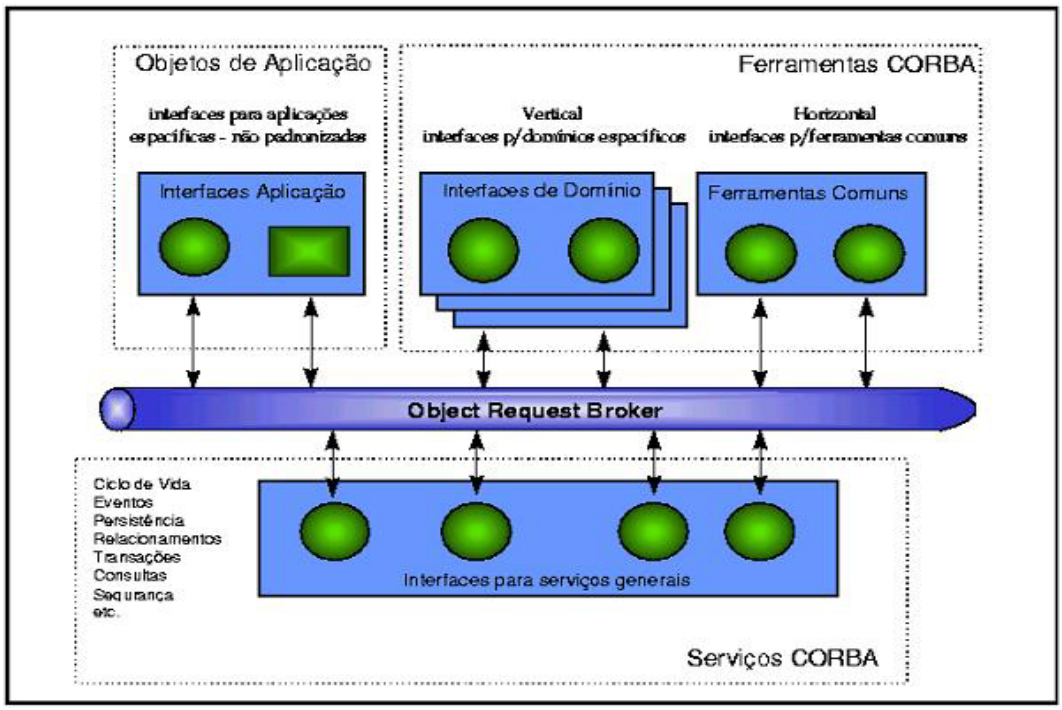


Figura 3.6 - Modelo de Referência OMA [OMG02].

O modelo de referência OMA (Arquitetura de Gerência de Objetos) é mostrada na figura 3.6, este modelo de referência provê *frameworks* que servirão como guia na adoção da tecnologia desenvolvida pela OMG.

A OMA é usada como um mapa, que diz a direção na qual os elementos deverão ser desenvolvidos. O modelo é composto de quatro categorias de componentes: *Object Request Broker*, Objetos de Aplicação, Serviços CORBA e Ferramentas CORBA.

Objetos de Aplicação estão relacionados às aplicações específicas do usuário, que serão integradas ao ambiente. As Ferramentas CORBA compreendem as funcionalidades de propósito geral que estarão disponíveis no ambiente distribuído. Os Serviços CORBA fornecem as funcionalidades básicas do ambiente distribuído como ciclo de vida de objetos, consulta e transações. Objetos comunicam-se com outros objetos via *Object Request Broker* (ORB), que é o elemento chave de comunicação.

O *Object Request Broker* - ORB é o *middleware* que estabelece a relação cliente/servidor entre os objetos. Usando um ORB, um cliente pode invocar um método num objeto servidor de forma transparente, que pode ser numa máquina local ou na rede. O ORB intercepta a chamada e é responsável por encontrar o objeto que

implementa aquela requisição, passar os parâmetros, chamar o método e retornar os resultados.

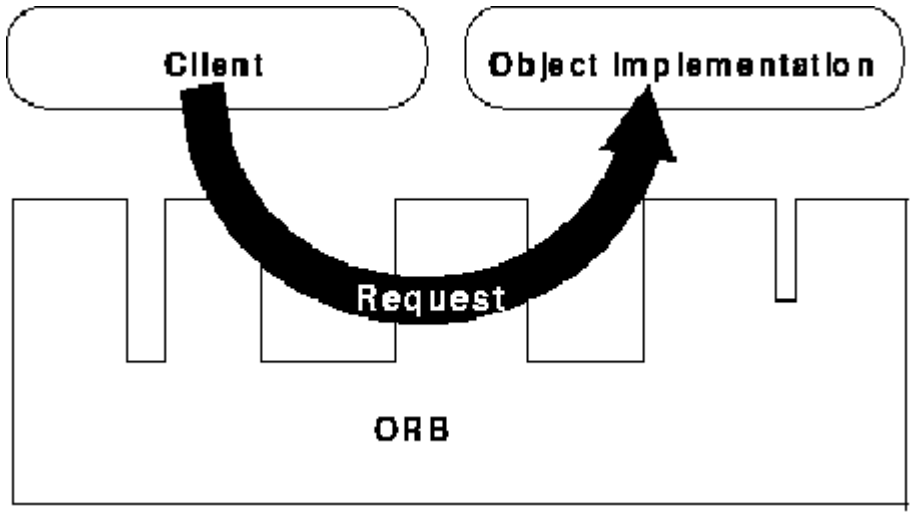


Figura 3.7: Requisição de uma implementação de Objeto [OMG02].

Na figura 3.7, pode-se ver como um cliente faz uma requisição a um serviço a ser fornecido por um objeto. O cliente possui uma referência ao serviço deste objeto, podendo portanto utilizá-lo, no entanto, o cliente não tem noção da localização do objeto, não sabe que linguagem foi utilizada em sua implementação, ou qualquer outro detalhe referente a implementação deste objeto.

A *Interface Definition Language* (IDL), define o tipo de objetos através da especificação de suas interfaces. Por definição, todos os serviços OMG devem ser especificados através do uso de uma linguagem declarativa, com ênfase na separação da interface e implementação [OMG02].

A IDL provê um sistema de encapsulação em duas camadas: tipos de dados (básicos e definidos pelo usuário) e objetos, que permite uma modelagem sofisticada para domínios distribuídos.

A CORBA IDL é puramente declarativa, não fornecendo detalhes de implementação. Métodos especificados em IDL podem ser ligados a vários tipos de linguagens, entre elas C, C++, Smalltalk, COBOL e Java. IDL fornece interface independente de sistema operacional e linguagem para todos os serviços e componentes

que são fornecidos pelo CORBA, permitindo que clientes e servidores desenvolvidos em diferentes linguagens e sistemas operacionais interajam

Para melhor entender o funcionamento do CORBA é importante saber a definição da arquitetura de interface, que composta de três componentes específicos: Interface de cliente, interface de implementação de objetos e ORB *core* .

1. **Interface Cliente:** A interface de cliente fornece as seguintes interfaces para o ORB e objetos servidores:
    - **IDL *stubs*** –representa a interface da composição das funções geradas pela definição da interface IDL e ligada ao programa cliente. Esta interface de invocação estática representa o mapeamento entre a linguagem cliente e a implementação ORB. A interface *stub* IDL traz o ORB direto para o domínio da aplicação do cliente, o cliente interage com o servidor remoto invocando suas operações exatamente como se invocasse operações em objetos locais;
    - ***Dynamic Invocation Interface*** **(DII)** – O ORB fornece um mecanismo de invocação dinâmica, permitindo as aplicações clientes a construção de uma requisição em tempo de execução. Usando o mecanismo DII, um objeto é acessado por uma chamada ao ORB ou por uma série de chamadas ao ORB no qual o objeto, métodos e parâmetros estão especificados. É responsabilidade do cliente especificar os tipos de parâmetros e retornos esperados;
    - **Repositório de APIs de Interface –** permite obter e modificar a descrição de todos os componentes de interface registrados, os métodos que são suportados e os parâmetros requeridos. Estas descrições são chamadas de assinaturas de métodos;
    - **Interface ORB** – A interface ORB permite que o código do cliente acesse diretamente as funções do ORB. Esta interface fornece somente umas poucas operações, tais como transformar em texto uma referência a objeto. A interface ORB é um dos

componentes que é compartilhado pelo lado cliente e pelo lado da arquitetura da implementação.

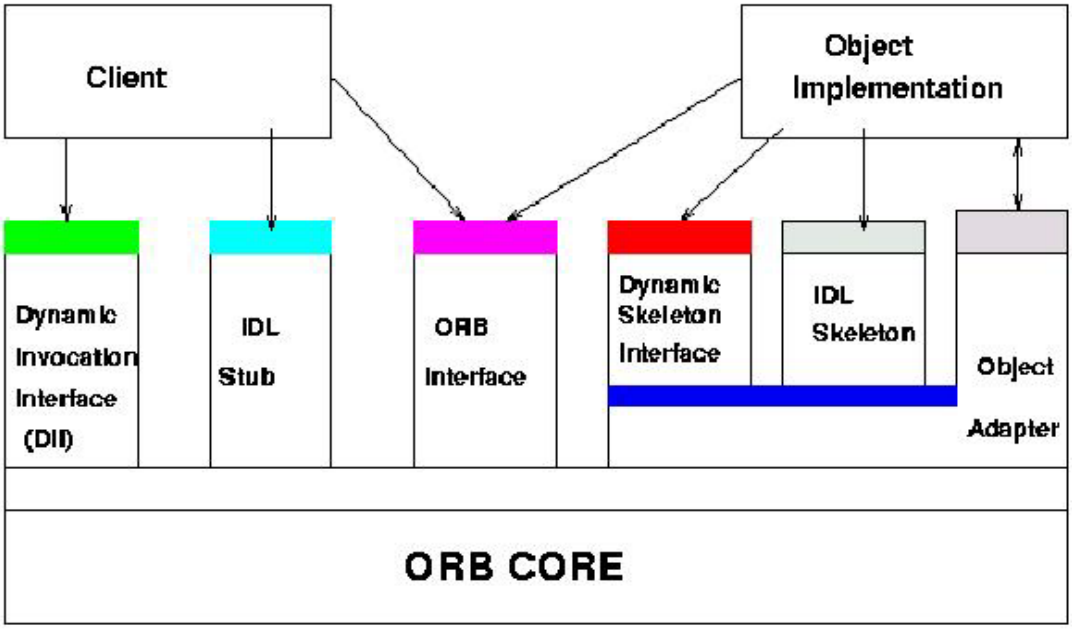


Figura 3.8: Estrutura de Interfaces CORBA [OMG02].

2. **Interface Servidor:** A interface de implementação é composta por interfaces de chamada para cima, permitindo chamadas do ORB para as implementações dos objetos.
   - **IDL *skeleton* –** É a parte do lado servidor correspondente ao stub da interface IDL;
   - ***Dynamic Skeleton Interface* (DSI)** – A DII especificada no CORBA 1.2 é um mecanismo do lado cliente, não existindo correspondente no lado servidor. Já no CORBA2, o *Dynamic Skeleton Interface* (DSI) é requerido pela interoperabilidade do ORB. O DSI é o lado servidor correspondente ao DII, provendo um mecanismo de ligação para os servidores entregarem requisições ao ORB para uma implementação cliente que não tenha o conhecimento em tempo de compilação do tipo do objeto. O DSI pode receber tanto invocações dinâmicas quanto estáticas;

- ***Object Adapter*** – ajuda o ORB com a entrega de requisições para o objeto e com a ativação dos objetos. Um *object adapter* associa a implementação de objetos com o ORB. *Object Adapter* podem ser especializados em prover suporte para certos estilos de implementações de objetos como OODB, *object adapter* para armazenamento persistente das referências aos objetos.
- **Repositório de Implementações** – provê um repositório de informações sobre as classes suportadas por um servidor, objetos instanciados e os seu identificadores. Também usado para armazenar informações associadas com implementações de ORBs;
- **Interface ORB –** Esta interface é compartilhada pelas implementações dos objeto e pelos objetos clientes.

3. ORB Core: o ORB Core é responsável pela manipulação da comunicação básica das requisições dos vários componentes, podendo ser visto como uma camada básica de transporte, como pode ser visto na figura 3.8.

3.3.2 DCOM

O *Distributed* COM (DCOM) é uma extensão do *Component Object Model* (COM), feito para suportar a comunicação entre objetos em computadores diferentes, seja em uma LAN, WAN ou até mesmo na Internet.

O COM define como os componentes e seus clientes interagem. Esta interação é feita de forma que o cliente e o componente possam conectar-se sem a necessidade de um componente intermediário do sistema.

Nos sistemas operacionais de hoje, os processos são protegidos uns dos outros. Um cliente que necessita estabelecer comunicação com um componente em outro processo não pode chamar o componente diretamente, necessitando fazer uso de alguma forma de comunicação inter-processos fornecida pelo sistema operacional. COM

fornece a comunicação inter-processos de forma transparente, interceptando chamadas do cliente e direcionando esta chamada para o componente em outro processo. A figura 3.9 ilustra como as bibliotecas *run-time* do COM/DCOM fornecem a conexão entre cliente e componente.

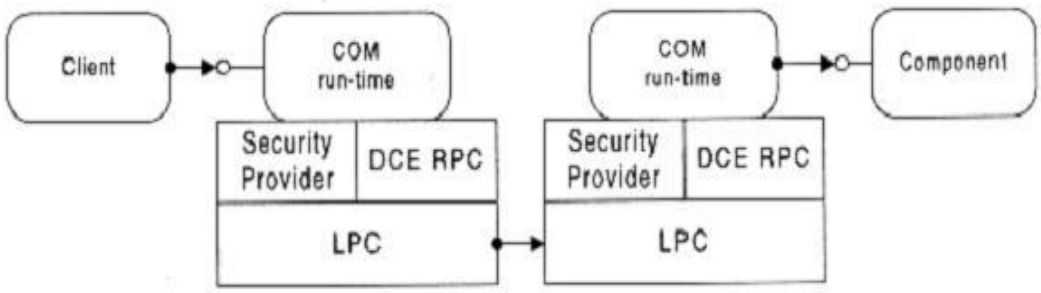


Figura 3.9: Bibliotecas *run-time* COM/DCOM [MIC96]

Quando o cliente e o componente residem em máquinas diferentes, a chamada inter-processos é substituída por um protocolo de rede pelo DCOM de forma transparente para cliente e componente, que não precisam tomar conhecimento do meio de comunicação utilizado para conexão.

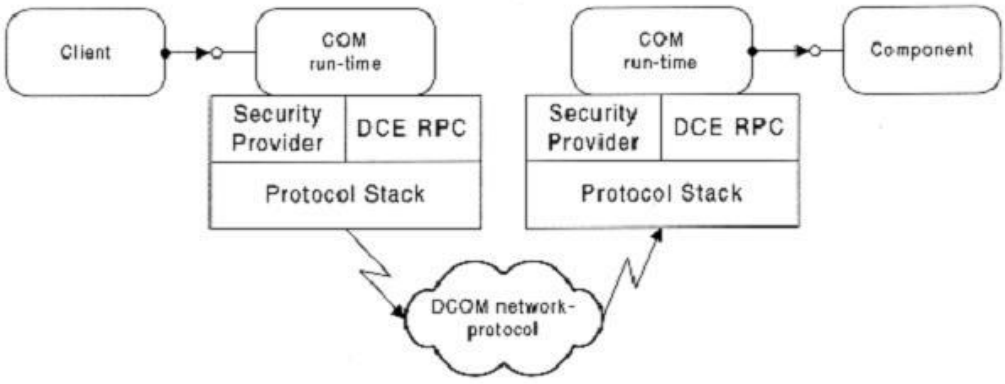


Figura 3.10: Arquitetura DCOM [MIC96]

O protocolo DCOM trás algumas características fundamentais para seu sucesso, sendo elas:

- **Independência de Localização**: Utilizando-se o DCOM, a localização dos componentes fica completamente escondida, podendo estar tanto dentro do próprio processo quanto em qualquer máquina ao redor do mundo, em qualquer uma das situações a forma de conexão entre o componente e o cliente é a mesma, não sendo necessária recompilação do programa ou mudança no código fonte, sendo necessária somente a realização de reconfiguração na forma de conexão. A independência de localização do DCOM simplifica a forma de distribuir componentes de uma aplicação de forma a otimizar a performance, os componentes podem rodar na máquina onde faz mais sentido sua presença, interface de usuário e validação próximo ao cliente, acesso a base de dados no servidor próximo ao banco de dados.
- **Gerenciamento de Conexões**: As conexões via rede são sempre mais frágeis do que as conexões dentro da máquina. Os componentes de uma aplicação distribuída precisam saber se um cliente não está mais ativo, mesmo em caso de falha de hardware ou rede. O gerenciamento de conexões é feito pelo DCOM através da manutenção de um contador de referência para cada componente. Quando um cliente estabelece uma conexão a um componente DCOM incrementa o contador de referência para este componente. Quando o cliente libera a conexão o contador é decrementado. Se o contador for zerado o componente pode ser liberado. Através de um protocolo de *ping*, o DCOM detecta se o cliente ainda está ativo. A conexão é considerada com problemas se no intervalo de três períodos de *ping* o componente não receber nenhuma mensagem de *ping*, neste caso, o DCOM decrementa o contador de referência para o componente liberando o componente caso o contador tenha chegado a zero.
- **Escalabilidade** Um fator crítico para uma aplicação distribuída é a habilidade de crescer de acordo com o número de usuários e a quantidade de dados. A aplicação necessita ser pequena e rápida quando a demanda é mínima, mas deve estar preparada para atender demandas adicionais sem perda significativa de performance ou confiança. O DCOM faz uso da capacidade do suporte a multiprocessamento do Windows NT/Windows 2000, otimizando a execução da aplicação de acordo com o número de processadores disponíveis. Caso o

uso de uma máquina não seja suficiente para acomodar a aplicação, a independência de localização do DCOM faz com que os componentes desta aplicação possam ser distribuídos entre outras máquinas da rede, bastando-se adicionar mais máquinas a medida em que a demanda cresce.

- **Acesso aos Componentes**: No COM e no DCOM, o cliente nunca tem acesso ao próprio objeto servidor, mas ao mesmo tempo o cliente nunca está separado do servidor por um componente do sistema a não ser que isto seja absolutamente necessário. Esta transparência é conseguida através da seguinte forma: o único meio do cliente conseguir falar com um componente é através de chamada de métodos. O cliente obtém os endereços destes métodos de uma tabela de endereços de métodos. Quando um cliente quer chamar um método em um componente isto é feito através de um ponteiro para o endereço deste método Quando o componente não está próximo ao cliente, o COM substitui o método de chamada pelo seu próprio método de chamada de procedimentos remotos (RPC), fazendo com que a chamada de um método do lado cliente seja empacotada e enviada ao lado servidor, onde será desempacotada e a chamada será feita ao método original.
- **Neutralidade de Protocolos de Comunicação**: O DCOM pode usar qualquer protocolo de transporte, incluindo TCP/IP, IPX/SPX e NetBIOS. O DCOM fornece um *framework* de segurança para todos estes protocolos, incluindo protocolos orientados a conexão e não orientados a conexão. O DCOM utiliza como protocolo de transporte preferido o UDP, a natureza sem conexão deste protocolo permite ao DCOM a realização de várias formas de aumento de performance, através da junção de muitos pacotes de confirmação de baixo nível com as mensagens de dados e *ping*. Mesmo sendo executado sobre protocolos orientados a conexão, tal como o TCP, o DCOM ainda oferece vantagens significantes sobre protocolos customizados especificamente para aplicações
- **Segurança**: o uso da rede para distribuição de aplicações não é um desafio somente por causa das limitações de largura de banda e latência, mas também devido a segurança de acesso aos componentes e aos clientes, visto que as aplicações estão acessíveis a qualquer um que tenha acesso físico ao meio de

transporte. O DCOM faz uso do *framework* de segurança fornecido pelo Windows NT, que suporta múltiplos mecanismos de identificação e autenticação

- **Tolerância a falhas**: Tolerância a falhas é vital para aplicações críticas que requerem alta disponibilidade. O DCOM fornece suporte básico para tolerância a falhas a nível de protocolo através do mecanismo de *ping*, que detecta falhas de rede e hardware.

Com a utilização do modelo COM, o desenvolvedor pode criar uma aplicação totalmente baseada em componentes, o que trás diversas vantagens no sentido da reusabilidade, organização da aplicação, encapsulação do código de forma a manter independente a complexidade da linguagem de programação, podendo ser usados *Visual Basic*, C++ entre outras e o acesso aos métodos é feito apenas via sua interface. Todos estes fatores são úteis no sentido de diminuir a complexidade de uma aplicação[MAL01]. Dependendo do ambiente em que esta aplicação esteja executando, pode ser útil, ou não a utilização do DCOM. Se a aplicação já está toda componentizada, a utilização de um ambiente distribuído passa a ser a escolha da melhor solução, bastando apenas fazer a configuração do ambiente.

3.4 Conclusão

Várias são as tecnologias existentes que podem ser usadas para auxiliar no desenvolvimento de aplicações para o gerenciamento de redes. A escolha de cada uma dessas tecnologias deve ser feita baseada em um para a captação das vantagens e desvantagens de cada uma delas seguido da construção da estrutura da aplicação. A estrutura básica da aplicação de gerência proposta é composta de interação com o usuário (interface e meios de comunicação), implementação das regras e armazenamento de dados.

Como uma boa opção para a interface da aplicação, pode ser citada a utilização do WAP, que tem algumas desvantagens relacionadas a problemas com o ambiente

Wireless como elevada taxa de erros, pouca largura de banda, ou até problemas ligados a interface como tamanho do *display*, dificuldade de digitação entre outros. Tais problemas foram bastante estudados e é certo que para a construção de uma aplicação utilizando WAP exigem-se certos cuidados, como tamanho de páginas, volume de processamento em cada página. Com tais cuidados, os benefícios podem superar as desvantagens do ambiente proporcionando um meio eficiente de acesso a informações.

Em se tratando de ambiente *wireless,* uma boa opção é o SMS, que são mensagens curtas que podem ser enviadas através de uma rede sem fio para fazer a comunicação entre o usuário e a aplicação. Esse tipo de tecnologia é extremamente funcional quando se trata da necessidade de pequenas notificações, como o aviso de uma falha ocorrida em algum objeto gerenciado.

Para a implementação da aplicação, é interessante se considerar a utilização de objetos distribuídos, que facilitam a busca de informações entre ambientes computacionais e a divisão de tarefas entre computadores, também chamada de *load balancing.* Dentre as implementações de objetos distribuídos podem ser citados como exemplos importantes, CORBA, que é um padrão desenvolvido pela OMG, e DCOM, que é uma implementação feita pela Microsoft. CORBA é um padrão bastante utilizado para a gerência de redes, pode ser desenvolvido em várias linguagens e é portável para várias plataformas. Já o DCOM é exclusivo par ambientes Microsoft, mas extremamente importante principalmente pela derivação do modelo COM, que é o modelo de desenvolvimento baseado em componentes, que permite diversas vantagens como reusabilidade, flexibilidade, segurança de código, entre outras.

4 O Sistema gerenciador de alarmes

4.1 A Proposta

A proposta deste trabalho é a criação de uma aplicação para gerência de falhas em redes de computadores que exigem alta confiabilidade. O objetivo básico desta arquitetura é prover uma infra-estrutura eficiente para a detecção e notificação de falhas, assim como, o acompanhamento das eventuais soluções para o problema.

Para cumprir este objetivo, a aplicação foi implementada com a capacidade de detectar falhas em qualquer ambiente, independente de plataforma. Um enfoque especial, foi dado a falhas decorrentes de recursos que utilizam a plataforma Windows, pois esta plataforma é freqüentemente usada em várias instituições. De forma a permitir que se fizesse a gerência de diferentes plataformas e dispusesse de uma boa integração com outras aplicações, o sistema teve que dispor de várias tecnologias para ser implementado, como pode ser visto a seguir:

1. **Gerência de recursos com plataforma Windows**: existem padrões que podem fazer a gerência deste ambiente de uma forma geral, como o próprio SNMP. Mas o ideal seria ter uma maior interação com o sistema operacional. Para possibilitar tal interação foi utilizado o WMI. O WMI dispõe de várias classes, herdadas do modelo CIM, que podem dar a uma aplicação de gerência inúmeras funcionalidades de gerência da plataforma Windows que não poderiam ser acessadas através de outros padrões.
2. **Gerência de recursos de outras plataformas**: como o WMI é puramente usado para o ambiente Windows, seria necessário a utilização de padrões para diferentes plataformas. Para este fim, foram utilizadas os padrões SNMP e CORBA. O SNMP é o padrão mais utilizado na Internet, portanto grande parte de equipamentos de rede já possuem suporte a ele, o que facilita a criação de uma aplicação que necessite de suas funcionalidades. Já CORBA é um padrão desenvolvido para ser usado em ambientes distribuídos. Com a utilização do

CORBA é possível a criação de objetos clientes responsáveis pela detecção de falhas em diferentes plataformas.

3. **Possibilidade de integração com outras aplicações**: no caso de já existir alguma aplicação com objetivos de detecção de falhas de software ou hardware, seria de extrema utilidade a integração desta aplicação com a aplicação desenvolvida neste trabalho. Para possibilitar esta funcionalidade, pode ser utilizada a interface CORBA que foi implementada. Desta forma, a aplicação existente só teria que invocar a interface de envio de alarme através da construção de um ORB *Client* para se comunicar com o ORB *Server* desenvolvido para o Gerenciador de Alarmes.

Assim que atendidas as necessidades do ambiente com relação as tecnologias usadas, o sistema precisou de uma forma de união dos alarmes gerados por essas diversas abordagens. Essa união foi feita de duas formas. A primeira é o armazenamento centralizado das informações. A segunda é a interface com o usuário, que apresenta de forma unificada os alarmes provenientes de todos os recursos da rede.

Para que o armazenamento fosse centralizado, cada uma das abordagens de detecção de falhas deve fazer o envio das características principais da falha para uma base de dados única assim que a mesma for captada. Ao ser inserida na base de dados, a falha é reconhecida como um alarme e deve chegar ao conhecimento de um responsável. A partir daí as demais funções do gerenciador de alarmes farão a interação com usuário através de uma notificação, objetivando a resolução rápida do problema. Todo processo feito com o alarme é gravado como forma de histórico para o gerenciamento das atitudes tomadas pelos responsáveis e geração de relatórios.

A interação visual do sistema e configuração é feita através da WEB e WAP, que são meios simples e bastante úteis para a interação com o usuário.

É através da WEB que o usuário poderá acompanhar todos os alarmes ocorridos na rede, quais as providências que estão sendo tomadas e quem está responsável pela resolução dos problemas. Também é pela WEB que são feitas as principais configurações do sistema.

Através da interface WAP é possível receber os alarmes, resolver, prorrogar ou apenas visualizá-lo. Esta foi uma opção para permitir maior agilidade na interação com o usuário, já que atualmente é muito comum administradores terem celulares.

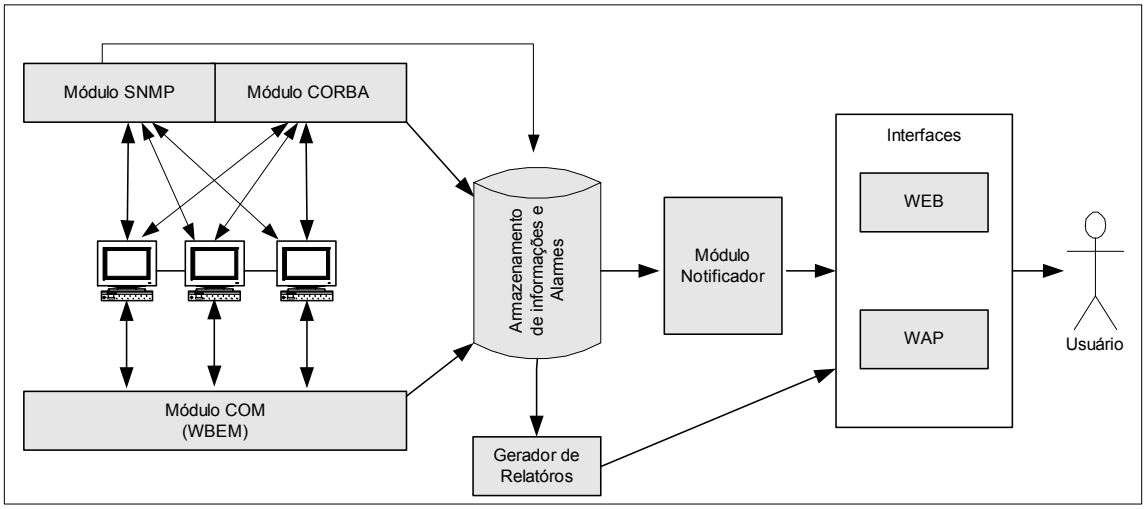


Figura 4.1 - Funcionamento do Sistema Gerenciador de Alarmes

Para simplificar a complexidade da aplicação, ela foi dividida em módulos, como pode ser visto na figura 4.1, que ilustra como o sistema deve funcionar. Dentre estes módulos podem ser citados:

- **Módulo Gerenciador COM:** este módulo é responsável pela gerência das falhas usando as tecnologias apresentadas pela iniciativa WBEM, apresentada com mais detalhes no capítulo 2. Sua funcionalidade principal é fazer a monitoração da rede em busca de alguma falha de *hardware* ou software em equipamentos que utilizam a plataforma Windows.
- **Módulo Gerenciador SNMP:** este módulo é responsável pela detecção de falhas ocorrida em equipamentos que têm suporte ao protocolo SNMP, independentemente de plataforma.
- **Módulo Gerenciador CORBA:** módulo responsável principalmente pela integração do Gerenciador de Alarmes com outras aplicações.
- **Módulo Notificador:** este módulo tem a funcionalidade de manter uma iteração com o usuário através de e-mail e mensagens SMS de forma a garantir que os responsáveis estarão cientes dos problemas ocorridos na rede.

- **Interfaces WEB e WAP:** são as interfaces utilizadas para fazer a interação do sistema com o usuário.
- **Gerador de Relatórios:** módulo responsável pela geração de relatórios sobre as ações tomadas com relação às falhas ocorridas no ambiente.

Os três módulos gerentes têm a função de monitorar a rede com o objetivo de detectar falhas. Assim que estas falhas são encontradas, as informações sobre os problemas são armazenadas em uma base de dados centralizada e o módulo notificador é acionado. As interfaces WAP e WEB, fazem a comunicação amigável com o usuário, que recebe o alarme e se torna responsável pela resolução do problema. Toda operação que o usuário fizer com o alarme recebido estará sendo armazenada em uma base de dados. Com essas informações o Gerador de Relatórios pode disponibilizar os relatórios de alarmes gerados, dispositivos que mais falharam, usuários que mais resolveram problemas e a velocidade em que os problemas foram resolvidos.

4.1.1 Facilidade de acesso

Para garantir a facilidade de acesso, o sistema tem dois tipos de interfaces, a WEB e WAP. Na WEB é onde serão feitas as principais configurações do sistema, toda parte de visualização do *status* da rede, envio de alarmes específicos e visualização dos relatórios. A interface WAP trará apenas a recepção dos alarmes e visualização dos mesmos. Com esses dois tipos de interface, o sistema garante que de qualquer lugar do mundo um administrador poderá estar em contato com a configuração do ambiente gerenciado e o funcionamento do mesmo.

Segundo trabalho realizado por [MAC01], existem muitos benefícios com a utilização de interface WEB para o gerenciamento de redes, que podem ser resumidos em:

- Substituição da linha de comando, que antes era usada para estabelecer uma comunicação com os elementos da rede;

- O treinamento de pessoas foi praticamente eliminado, uma vez que não é mais necessário saber a sintaxe de cada comando e o uso das tecnologias Internet são muito mais fáceis de se aprender;
- O acesso aos *swiches* passaram a ser controlados;
- Aumento do número de pontos de acesso;
- Interface amigável com o usuário.

4.1.2 Divisão em camadas

A implementação do sistema gerenciador de alarmes, é dividida em três camadas, sendo elas a interface com o usuário, a camada de negócios, que é a lógica do sistema e a camada de dados. Com esse tipo de subdivisão é possível implementar funcionalidades na camada interna sem interferir nas outras camadas. Essa abordagem permite a criação de uma ferramenta modularizada que possibilita o acréscimo de novas funcionalidades sem a necessidade de alteração nas demais camadas.

A camada de negócios (lógica) é a camada que contém a implementação que o sistema necessita para o gerenciamento de uma rede, geração de relatórios, configurações entre outros. Dentro da camada de negócios, serão implementados vários módulos, cada um com a responsabilidade de uma tarefa do sistema somados a dois módulos de interface que serão responsáveis pela interação com o usuário. Os módulos do sistema são:

- Módulo Gerenciador COM
- Módulo Gerenciador CORBA
- Módulo SNMP
- Módulo Notificador
- Gerador de Relatórios
- Interface WAP
- Interface WEB

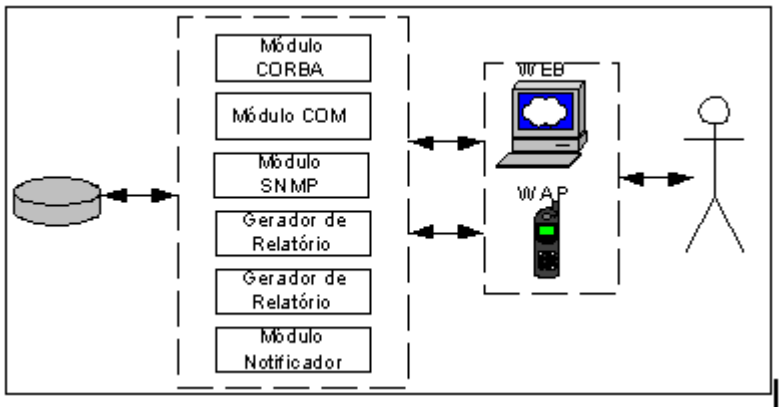


Figura 4.2 - Módulos do Sistema

A figura 4.2 mostra os módulos funcionais dos sistemas independentes um dos outros, interagindo com o usuário conforme a necessidade. Todos os dados são armazenados em base de dados única, conseqüentemente todas as operações são registradas como histórico. Com a implementação feita desta forma o sistema pode facilmente interagir com diferentes sistemas operacionais, apenas construindo-se um módulo com tal responsabilidade. Se houver a necessidade da criação de novos módulos, estes podem ser adicionados ao sistema sem causar problemas para os demais módulos.

4.1.3 Notificação eficiente

Para que o sistema de alarmes garanta que o responsável saiba da ocorrência de um problema, foi implementado um módulo específico cuja responsabilidade é a notificação do administrador responsável.

A principal idéia é que na ocorrência de uma falha, serão disponibilizados meios de comunicação do sistema com o administrador que receberá o alarme. O sistema deverá ter a informação de recebimento de alarme para tomar a próxima providência, que pode ser reenviar a notificação, caso não receba nenhuma resposta ou cessar o

processo de envio de notificação para o alarme em questão, caso um administrador envie a resposta de alarme recebido com sucesso.

4.2 Trabalhos Relacionados

Atualmente há uma tendência em se fazer o desenvolvimento baseado em componentes, que pode ser visto em trabalhos como [BEN00], [MAL01] onde são apresentadas as várias vantagens dessa abordagem. No Gerenciador de Alarmes, foi utilizada a estrutura Windows DNA, que é a arquitetura de desenvolvimento proposta pela Microsoft, que separa a camada de interface, camada de negócio e armazenamento de dados.

A camada de interface do Gerenciador é composta pelas interfaces WEB e WAP. Estas interfaces permitem maior interação entre os problemas da rede e o usuário, sem a necessidade do conhecimento de comandos ou dos principais conceitos da própria gerência. Nem sempre essa é uma preocupação dos autores que realizam trabalhos nesta área. Por exemplo [VOU01], onde apesar de ser baseado em um dispositivo móvel, que tem uma interface simples, o usuário tem a necessidade de ter plenos conhecimentos dos comandos que deverão ser utilizados para poder enviá-los via SMS. Em [GUI01], o usuário necessita ter conhecimentos na linguagem Java para fazer uso da API criada. Já as interfaces do Gerenciador, foram planejadas para não haver a necessidade de conhecimento em gerência de redes. Na interface WEB, todas as configurações são simples e permitem que qualquer usuário possa configurar os objetos a serem gerenciados. A interface WAP foi desenvolvida com os cuidados necessários para diminuir problemas causados por dispositivos móveis, como tamanho do *display,* dificuldade de digitação e velocidade das páginas. Nos trabalhos pesquisados não foi encontrada uma solução onde eram apresentadas as duas possibilidades de interface para interagir com o usuário. Normalmente são utilizadas aplicações *desktop* ou WEB ou WAP, o que é um diferencial para o sistema implementado nesta dissertação, que

possui as duas interfaces aumentando a interação com o usuário, que chega mais rapidamente a informação da falha e conseqüentemente à solução da mesma.

Na camada de negócios são implementadas as funcionalidades do sistema. Estas funcionalidades foram divididas em componentes que tem responsabilidades como configuração do sistema, criação de usuários, implementação da segurança das páginas da interface WEB, os módulos de gerência e o sistema de notificação. A junção destes componentes para um objetivo único formam os módulos. Esse tipo de divisão permite que sejam utilizadas diferentes tecnologias nos módulos.

No módulo COM, foi utilizado o WMI, que é uma implementação do modelo CIM, padrão da iniciativa WBEM, para fazer a gerência de toda rede Windows. O WMI já tem integração com algumas formas de gerência como SNMP, gerenciamento WIN32 entre outros. Este módulo gerencia os principais problemas que podem ocorrer em uma rede Windows, sendo estes problemas de software ou hardware. Os trabalhos encontrados que utilizam os padrões apresentados pela iniciativa WBEM, como [BEN00] ou [FES99], fazem a integração deste padrão com outras formas de gerência, como OSI, CORBA, de acordo com suas necessidades. A implementação feita por estes trabalhos, não é adequada ao gerenciamento do ambiente proposto nesta dissertação. Por outro lado, o módulo COM, desenvolvido com esta tecnologia, torna-se útil apenas para problemas causados em redes Microsoft. Por este motivo foram desenvolvidos como um adicional, os módulos SNMP e CORBA. O módulo SNMP tem como objetivo fazer a utilização do protocolo SNMP para a recepção de falhas ocorridas em qualquer dispositivo que tenha um agente SNMP instalado, independente de plataforma. O módulo CORBA é a implementação de uma interface que pode ser chamada de qualquer objeto gerenciado da rede para a geração de um alarme. Grande parte dos trabalhos na área de gerência de redes utilizam o SNMP para fazer o gerenciamento, como por exemplo [MAL01] e [HON01], outro preferem WBEM com CORBA, como é o caso de [BEN00]. A vantagem do Gerenciador de Alarmes proposto nesta dissertação, é a união destes três módulos, que é um diferencial em relação aos outros trabalhos.

As falhas capturadas pelos três módulos, são tratadas e armazenadas em base de dados, formando assim a terceira camada da arquitetura DNA, a camada de dados. Juntamente com as falhas, são armazenadas todas as informações do andamento da solução das falhas ocorridas no ambiente. A notificação do usuário é feita com esquema

cíclico de notificação para que o sistema possa garantir que pelo menos um responsável foi notificado de uma falha. Quando a notificação é recebida, o usuário receptor passa a ser o responsável pelo encaminhamento da solução da falha.

Esta característica de gerenciamento das atitudes tomadas pelos responsáveis e geração de relatórios se difere do enfoque dos trabalhos realizados na área de gerência de falhas, como os apresentados no capítulo 2.8.4, pois estes trabalhos focam apenas na detecção e correção do erro. Já o Gerenciador de Alarmes proposto nesta dissertação, tem como objetivo armazenar o andamento da resolução das falhas e relaciona-las aos usuários que as receberam/resolveram. Com esta interação, é passada uma responsabilidade maior aos usuários com o intuito de garantir que alguém será o responsável pela solução do problema. Com as informações armazenadas no sistema, tem-se a possibilidade de averiguação de responsáveis e das atitudes tomadas pelos mesmos dentro do ambiente.

4.3 Conclusão

A proposta da dissertação é a criação de uma aplicação para a gerência de alarmes ocorridos em reação a falhas em redes de computadores que exigem alta confiabilidade. A realização desta tarefa num ambiente heterogêneo com predominância da plataforma Windows, exige a utilização de várias tecnologias. A junção destas em uma aplicação de gerência permite o gerenciamento das falhas de um ambiente de modo bastante eficiente. A união das informações de gerência recuperadas pela utilização destas tecnologias são armazenadas em uma base de dados central, que é acessada por toda aplicação, disponibilizando para o usuário, o *status* do funcionamento da rede através das interfaces WEB e WAP.

Existem outros trabalhos que implementam ferramentas de gerência de falhas e alarmes, mas o enfoque destes é diferente, pois cada um se propõe a resolver problemas de seu próprio ambiente. O diferencial deste trabalho, é o tipo de ambiente a ser gerenciado que junta tecnologias que normalmente são usadas separadamente. Outro

diferencial são as novas características, como a forma de notificação utilizada e a geração de informações que fazem que os administradores tenham uma maior responsabilidade pela resolução dos problemas encontrados na rede

5 Ambiente de Desenvolvimento

Este capítulo tem como objetivo apresentar o sistema desenvolvido para a validação da proposta de resolução de problemas relacionados a gerência, anteriormente apresentados nesta dissertação.

5.1 Implementação do Sistema Gerenciador de Alarmes

O Gerenciador de Alarmes foi implementado usando componentes de software Esta estratégia de programação permite a criação de diferentes módulos para diferentes funções sem haver interferência entre eles.

A decisão da arquitetura a ser usada para implementação, foi baseada em um dos módulos do sistema que faz o gerenciamento de recursos com a plataforma Windows. Este módulo, por usar uma tecnologia criada pela Microsoft, o WMI, necessita de uma integração maior com o sistema operacional. Esta integração só é fornecida através da utilização dos recursos fornecidos pela própria Microsoft. Por este motivo foi escolhida arquitetura Windows DNA para fazer a implementação de grande parte do sistema.

Windows DNA é uma estratégia para desenvolvimento de aplicações distribuídas utilizando a plataforma e a tecnologia para componentes [GAR98]. Sua estrutura para desenvolvimento de aplicações é baseada em componentes, utilizando arquitetura de três camadas, como é mostrado na figura 5.1:

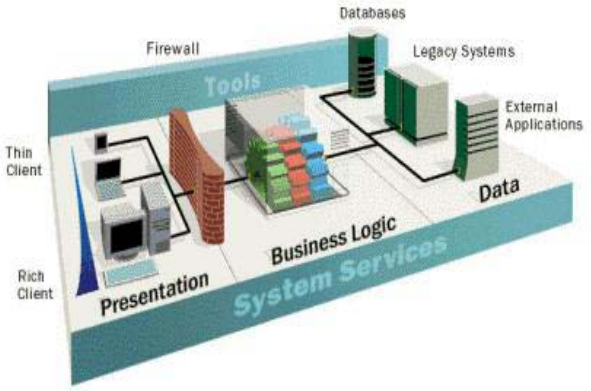


Figura 5.1 - Arquitetura Windows DNA [MIC99]

- **Camada de Apresentação:** é responsável por manter uma apresentação amigável ao usuário. No caso de aplicações WEB, normalmente é usado HTML, DHTML, e scripts de cliente, como por exemplo JavaScript.
- **Camada lógica de Negócio:** é a camada que contém a lógica do sistema. Nessa camada são utilizados os componentes, que contêm a regra de negócio do sistema, que podem ser implementados em Visual Basic, Visual C++ entre outros. Esses componentes são ativados pelas páginas ASP que juntamente com a camada de apresentação, fazem a interface com o usuário
- **Camada de Dados:** é a camada de armazenamento de dados, no caso de tecnologias Microsoft, é usado o SQL Server como servidor de banco de dados.

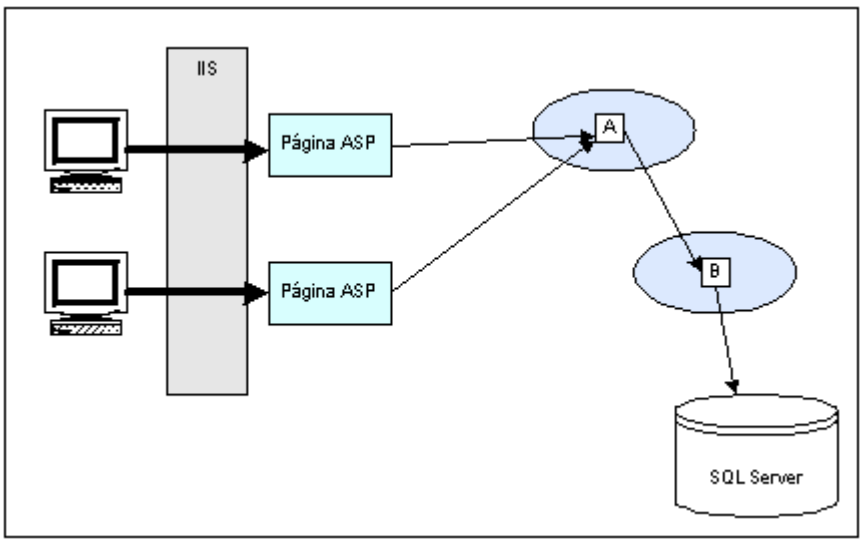


Figura 5.2 - Estrutura de implementação Windows DNA [MIC99]

A figura 5.2, apresenta a estrutura de implementação baseada em Windows DNA. No caso, duas aplicações requisitam um mesmo serviço. Essa requisição passa pelo servidor WEB e requisita a uma página ASP. Cada página cria uma instância de um componente A que por sua vez instancia um outro componente B que faz o acesso a base de dados.

Esta estrutura permite muitos benefícios a um sistema, como:

- Reusabilidade;
- Segurança de código;
- Escalabilidade;
- Interoperabilidade com sistemas e base de dados existentes;
- Redução da complexidade;
- *Time to Market*[4]*;*
- Os componentes podem ser implementados em várias linguagens;

Os módulos implementados seguindo esta arquitetura foram os Módulos notificador, gerador de relatórios, gerência COM, as interfaces WEB e WAP. As linguagens de programação utilizadas foram Visual Basic 6.0 para a camada de negócios e ASP para as interfaces. *Visual Basic 6.0* é uma linguagem bastante simples, permitindo um aumento na velocidade de desenvolvimento, não deixando de ser eficiente e alcançando a performance desejada para o sistema [MIL01]. O ASP é baseado do VBScript, que por sua vez tem uma boa integração com os componentes em VB (Visual Basic).

Os módulos SNMP e CORBA por sua vez, não puderam ser desenvolvidos dentro da arquitetura Windows DNA. Os pré-requisitos destes módulos, são o gerenciamento de recursos independentes de plataformas. Por esse motivo eles foram desenvolvidos em Java, pois essa era a linguagem que fornecia as classes de gerenciamento de falhas necessárias para o gerenciador de alarmes. Com estes módulos fora da arquitetura Windows DNA, os recursos das demais plataformas puderam ser gerenciados.

---

[4] *Time to Market:* é o tempo que leva para uma aplicação entrar no mercado sem estar desatualizada.

Neste capítulo serão descritos como foram implementados cada um dos módulos e de onde surgiu a necessidade de cada um.

5.1.1 Módulo Gerenciador COM

Existem muitos problemas que podem ocorrer em uma rede onde existam vários PCs. Estes problemas geralmente exigem uma detecção rápida, para que o administrador da rede não gaste muito tempo para descobrir e corrigir o erro. Um exemplo é o superaquecimento do processador, que pode ser causado pelo *cooler* ter deixado de funcionar. Esse tipo de problema pode fazer com que o computador reinicie várias vezes. Conseqüentemente, causa problemas ao usuário do computador e ao administrador da rede. Ao usuário do computador, por atrasar seu trabalho e ao administrador da rede por ter que pesquisar qual o problema que está ocorrendo dentre várias opções. Se o problema fosse detectado antes de sucessivos reinícios de sistema, haveria redução de gasto de tempo com a solução e aumentaria a produtividade dos usuários.

Estes problemas puderam ser observados no ambiente de rede *Windows* da Paradigma[5]. Nesta empresa a prevenção de problemas comuns a PCs como aquecimento excessivo do processador, superutilização de CPU entre outros pequenos problemas ligados a hardware, poderiam ser monitorados automaticamente de forma a auxiliar a administração da rede e apresentar soluções rápidas e eficientes. Tal atitude facilita também o trabalho dos desenvolvedores, que não podem perder muito tempo com esse tipo de problema. A partir desta observação, concluiu-se que havia a necessidade de monitorar alguns dispositivo como:

- Processador
- Disco
- Temperatura

---

[5] Paradigma: é uma empresa de softwares , *partner Microsoft*, que atua na área de aplicações de comércio eletrônico para a Internet.

- Memória

Para atribuir neste módulo as funcionalidades necessárias para a resolução destes problemas, foram criados dois componentes responsáveis pela gerenciamento desses PCs. No primeiro deles, são implementados toda conexão com o WMI, onde através deste, o segundo componente tem as informações necessárias para a tomada de decisões. Esse comportamento pode ser observado na figura 5.3, que mostra o sistema gerenciador fazendo uma chamada ao componente captador de falhas que por sua fez, requisita a informação ao componente onde são implementadas as funções do WMI.

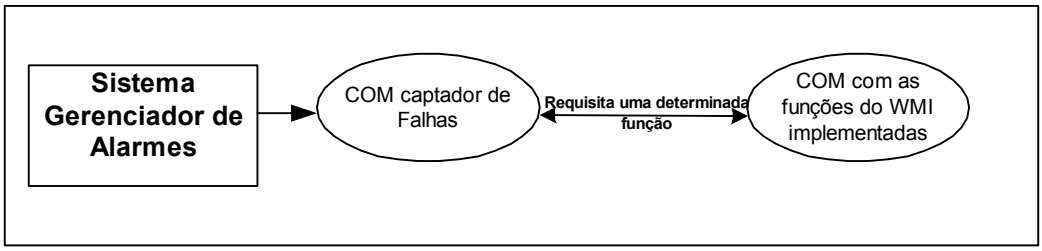


Figura 5.3 - Chamada a funções WMI pelo componente captador de Falhas

O WMI (*Windows Management Instrumentation*), como já citado na seção 2.7.2, trabalha com uma estrutura baseada em *Providers*, onde cada *Provider* tem uma responsabilidade na gerência. Alguns dos *Providers* que o WMI implementa são:

- SNMP *Provider:* funcionam com *gateway* para sistemas e dispositivos que usam SNMP para gerenciamento.
- Win32 *Provider*: fornece informações sobre o sistema operacional, equipamentos periféricos, *file systems* e segurança.
- *Registry Provider*: permite que as chaves do Registro possam ser lidas, criadas e reescritas .
- Existem outros *providers,* e também podem ser desenvolvidos para uma atividade específica.

Neste trabalho, é abordado apenas o Win32 *Provider* . Através deste *Provider*, é possível obter informações de todas as classes necessárias para o monitoramento de ambientes Windows. As classes utilizadas do Win32 *Provider* para monitoração das variáveis apresentadas são as seguintes:

- **Win32_processor** – é a classe do WMI responsável pelo gerenciamento do processador, como endereço, velocidade, capacidade entre muitas outras. Neste trabalho é utilizada a propriedade *LoadPercentage.*
- **Win32_LogicalDisk** – responsável pelo gerenciamento dos discos do computadores, tanto lógicos como físicos. As propriedades utilizadas no trabalho serão *Size*, *FreeSpace* e *FileSystem.*
- **Win32_TemperatureProbe** – representa as propriedades do sensor de temperatura da máquina. Utilizando a propriedade *NominalReading.*
- **Win32_PhysicalMemory** – representa o dispositivo de memória física do computador. Para monitora-la será utilizada a propriedade *Capacity.*

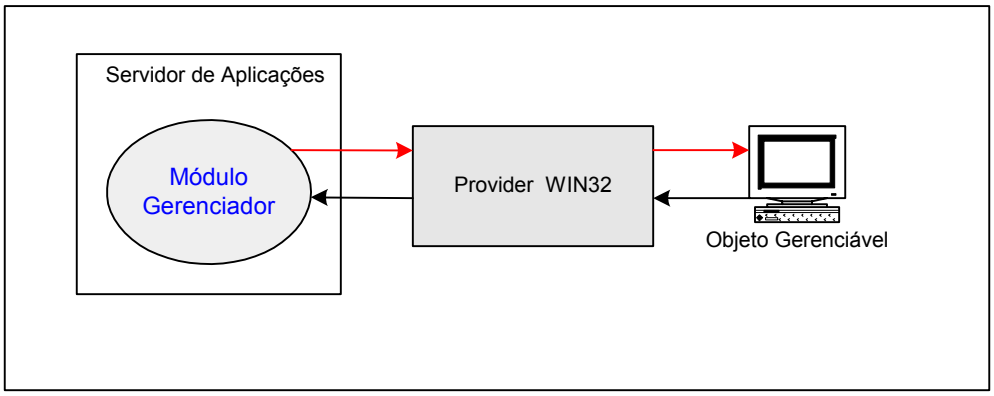


Figura 5.4 - Atuação do Módulo Gerenciador.

A figura 5.4 ilustra a atuação do módulo Gerenciador, que através do *Provider* WIN32 resgata as informações do objeto gerenciável e retorna novamente ao módulo. Se houve alguma falha, a mesma é armazenada no banco de dados e o módulo de notificação, que será visto neste capítulo, é acionado.

Este módulo foi definido como uma DLL, e configurada para a cada 5 minutos rastrear cada um dos objetos gerenciáveis e seus respectivos dispositivos configurados pelo sistema, em busca das possíveis falhas que podem estar ocorrendo. A figura 5.5 ilustra o gerente, rastreando os objetos gerenciáveis OBJ1, OBJ2, OBJ3 e OBJ4 e seus respectivos dispositivos, que podem ser CPU, memória, espaço em disco e temperatura.

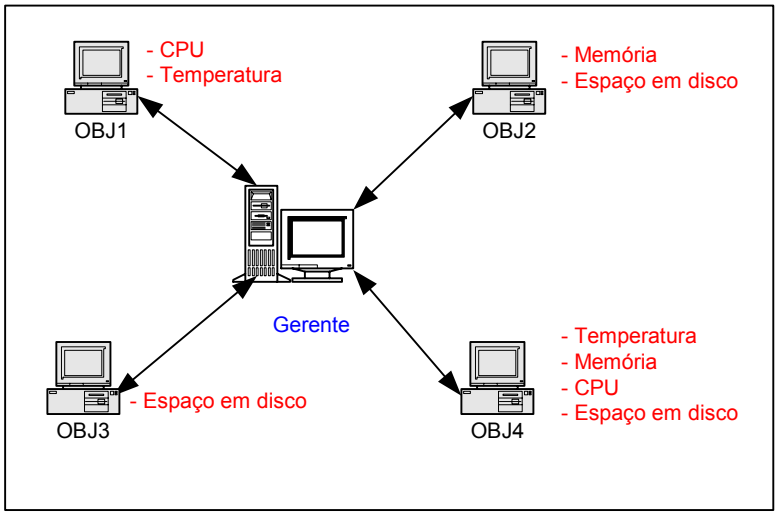


Figura 5.5 - Gerente gerenciando os objetos e seus respectivos dispositivos

Tanto os objetos gerenciáveis quanto o intervalo de tempo para o rastreamento podem ser configuráveis. Os objetos que serão gerenciados no sistema devem ser incluídos via interface WEB, que será detalhada ainda neste capítulo. Nessa configuração são definidos quais os dispositivos que necessitam de gerenciamento.

Juntamente com a definição dos dispositivos de cada objeto, são definidos pelo usuário, o seu valor crítico. A partir deste valor o módulo gera ou não um alerta. Por exemplo:

- Objeto: OBJ1
- Dispositivo: Temperatura
- Valor crítico: 60º C

Com esta configuração, o alarme só irá ser disparado quando a temperatura for superior a 60º C. Desta forma, pode-se definir diferentes limiares de acordo com a necessidade.

5.1.2 Módulo Notificador

Assim que uma falha ocorre numa rede, automaticamente um responsável deve ser avisado para resolver o problema. O ideal é que houvesse alguma garantia que alguém realmente recebesse esse alarme.

Essa funcionalidade é atribuída ao módulo notificador do sistema. Assim que ocorre alarme, este é automaticamente acionado e tem a tarefa de notificar um responsável de acordo com a severidade do problema. Se o problema for crítico, o módulo notificador mantém-se notificando até que um responsável leia o alarme e o receba.

A escolha dos meios de notificação foi baseada nos meios de comunicação que são mais usados nos dias de hoje, como mostra a figura 5.6, que através do sistema gerente, notifica o usuário das seguintes formas:

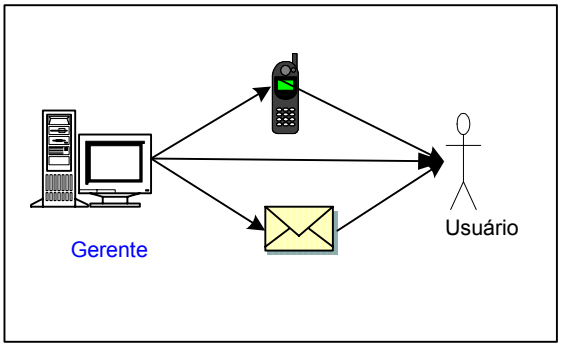


Figura 5.6 - Notificação via e-mail,direta via interface e SMS.

- **via SMS**: grande parte dos celulares fabricados ultimamente já vem com suporte de recepção de mensagens curtas, e é muito comum a necessidade de um administrador de rede precisar sempre ter a mão um celular, próprio ou da companhia pelo qual ele trabalha. Por essas razões, a utilização deste tipo de notificação torna-se bastante eficiente;

- **via *e-mail:*** o e-mail é um meio bastante difundido e com grande facilidade de acesso. Por essas razões, torna-se fundamental a notificação por este meio. Servindo também de documentação da ocorrência do alarme;
- **via interface WEB**: é uma alternativa para aqueles que de uma forma ou outra não tiveram acesso a *e-mail* ou celulares, mas podem entrar via interface WEB e verificar os alarmes ocorridos;

Este módulo também é um componente COM, implementado em *Visual Basic* 6.0. Desta forma, ele pode ser reutilizado de várias formas. Este componente é acionado imediatamente pelos módulos gerentes quando ocorre uma falha, e um alarme é gerado. Dependendo da severidade do alarme, é enviado um tipo diferente de notificação:

1. **Baixa**: apenas um *e-mail* é enviado e a possibilidade de visualização via WEB;
2. **Média**: *e-mail* e WEB;
3. **Alta**: é enviado um *e-mail,* uma mensagem SMS e visualização via WEB;
4. **Crítica**: é enviado *e-mail* e as mensagens SMS são enviadas a cada 5 minutos até que um responsável leia a mensagem, e a marque o alarme como recebido;

No caso da mensagem crítica, é feito um agendamento que faz a monitoração de tempo em tempo para verificar se a mensagem já foi lida por alguém. Esse *status* é verificado quando um usuário entra no sistema, via WAP ou via WEB, e sinaliza o recebimento do alarme. No caso, enquanto não for feita essa sinalização o sistema estará reenviando a notificação e um contador é acionado para futura verificação de quanto tempo levou para um usuário receber o alarme.

Existe uma limitação no uso do SMS e depende da operadora a liberação do serviço. Existem *softwares* que são capazes de fazer o envio de SMS por várias operadoras, mas isso geralmente é feito conectando-se com o *site* da operadora em questão e submetendo os dados que são esperados pela página de envio, esse é o funcionamento do "*SMS Send"* que é um programa disponibilizado em [SMS00]. Esse

programa tem arquivos de configuração para cada operadora existente. Ele mostra um modelo e um documento que ensina como codificar este *script*. Neste trabalho, as operadoras que iriam ser utilizadas seriam Global Telecom e Tim Sul. No caso da Global, o usuário ao se cadastrar no sistema, deve adicionar no formulário de cadastro, o seu usuário da Global. Já com a Tim esse processo foi inviabilizado pela política de segurança implementado na página da TIM, onde para poder enviar uma mensagem SMS, é necessário entrar com o número do celular. Em seguida a Tim envia para esse celular uma senha, que com esta o usuário poderá mandar até 5 mensagens. Com esse sistema fica inviabilizado a geração do arquivo necessário para o aplicativo "SMS Send". Esse problema poderia ser solucionado fazendo um acordo com a operadora, que poderia liberar o acesso para aplicação, mas isso acarretaria em um custo, que devido a natureza educacional do trabalho optou-se por não utilizar esta operadora.

5.1.3 Interface WEB

Nos dias de hoje, é importante que a visualização da rede não seja feita apenas através de uma console de um único computador. É preciso levar em consideração a possibilidade de se ter acesso as informações de gerência por diversos pontos. Para viabilizar essas características, a interface escolhida para este projeto, é a WEB.

É via Interface WEB que são feitas as principais configurações do sistema e acesso as informações de gerência como:

- Criação de usuários e tipos de usuários;
- Criação de tipos de alarmes;
- Disparo de alarmes;
- Visualização de alarmes ocorridos;
- Configuração dos objetos e seus respectivos dispositivos a serem gerenciados;
- Visualização de relatórios de conduta.

A interface WEB é a responsável por toda configuração e interação amigável com o usuário. Esse módulo é implementado em ASP (*Active Server Pages)* e executa em um servidor Windows 2000 com IIS (*Internet Information Server*) 4.0 ou superior. A escolha da linguagem ASP, foi feita por ela permitir uma maior integração com os componentes já existentes no sistema, que foram criados para atender os processos e as interfaces do gerente de alarmes. Toda movimentação e configuração do sistema pode ser visualizado através desta interface de qualquer lugar com acesso à *Internet.*

Todos os usuários que terão acesso ao sistema deverão ser cadastrados via interface WEB. O cadastro é composto pelas principais características do usuário e entre elas é feita uma associação do usuário com um tipo de usuário.

Através destes tipos, podem ser criadas diversas maneiras de acesso às páginas, aumentando a segurança do sistema. Esta segurança é garantida com a utilização de variáveis de sessão, uma vez que o usuário entre no sistema, seu *login* e senha são armazenados na sessão do *browser,* mantendo a identificação do usuário. Todas as páginas tem em seu escopo um método que verifica as permissões do usuário identificado na sessão. Essa funcionalidade é assegurada mantendo em base de dados a informação das páginas que cada tipo de usuário terá acesso. Desta forma, torna-se impossível o acesso a qualquer página sem estar logado ou o acesso direto a uma página que não se tem permissão.

A manutenção das permissões em base de dados torna possível a criação de diversas configurações para tipos de usuários, por exemplo: pode existir um tipo de usuário destinado apenas a cadastrar outros usuários e outro tipo é responsável apenas por visualizar os alarmes ocorridos.

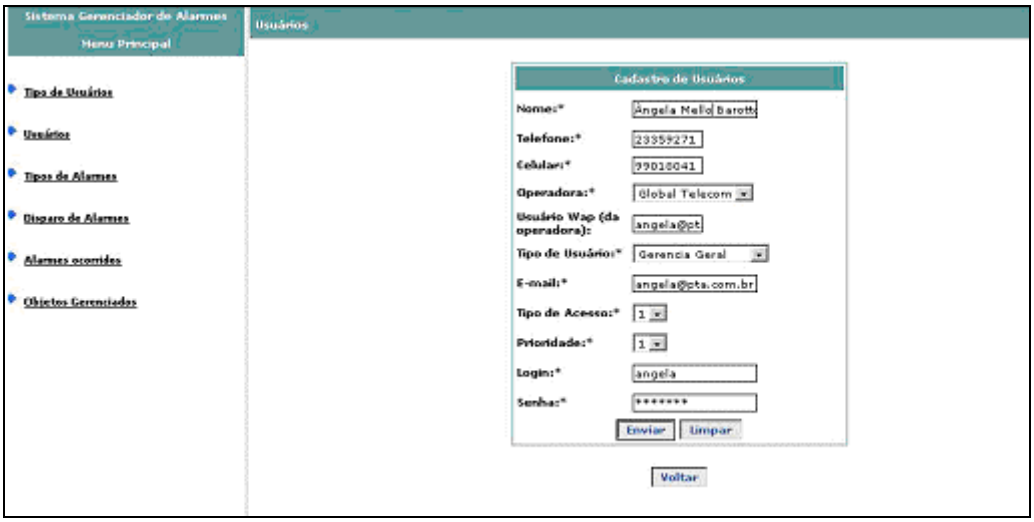


Figura 5.7 - Cadastro de usuários.

A figura 5.7 mostra o formulário de cadastramento do usuário que foi desenvolvido. É um formulário simples onde são obrigatórios campos como Nome, telefone, celular, operadora de celular (caso seja a Global, é obrigatório o usuário SMS), Login, Senha, *e-mail* e o tipo de usuário.

Com o devido acesso, os usuários podem operar o sistema, como fazer a manutenção dos tipos de alarmes, que podem facilmente ser criados pela WEB. Como o sistema permite que usuários, possam disparar alarmes via WEB, pode ser necessária a criação de novos tipos de alarme para atender diferentes necessidades. Por exemplo, uma pessoa que não é responsável pela rede, percebe uma falha no sistema. Se o tipo do alarme não estiver cadastrado, deve ser feito o cadastro e em seguida o disparo do alarme para que as pessoas responsáveis sejam notificadas de forma que o sistema mantenha um histórico do problema ocorrido.

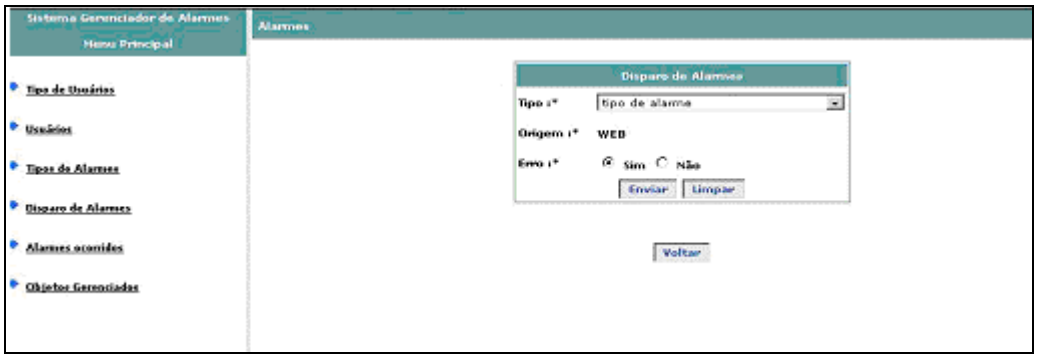


Figura 5.8 - Disparo de alarmes via WEB.

A funcionalidade de disparo manual de alarmes, é mostrada na figura 5.8, onde através de um cadastro simples é enviado um alarme. Para enviar um alarme específico, é preciso cadastrar o tipo deste alarme com antecedência ao envio do mesmo.

Todo alarme ocorrido na rede pode ser visualizado pela interface WEB. O portal permite uma consulta por filtragem de todos os tipos de alarme e pelo andamento da solução, como pode ser visualizado na figura 5.9, que mostra o formulário de consulta com o resultado mostrado na tabela logo abaixo.

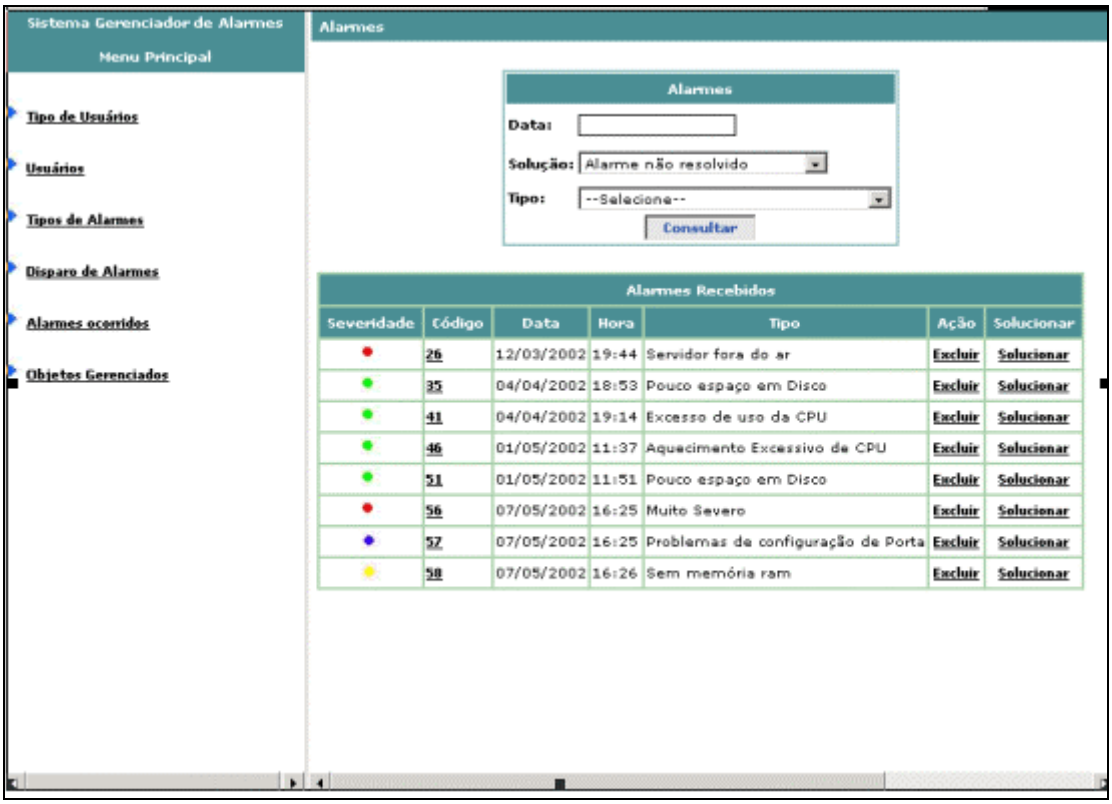


Figura 5.9 - Visualização de alarmes.

Assim que visualizados, o usuário poderá receber o alarme tomando parte na solução do problema e responsabilizando-se pela solução da falha. Toda seqüência de movimentações de recepção, prorrogação e solução de alarmes é devidamente documentado para futura implementação de relatórios.

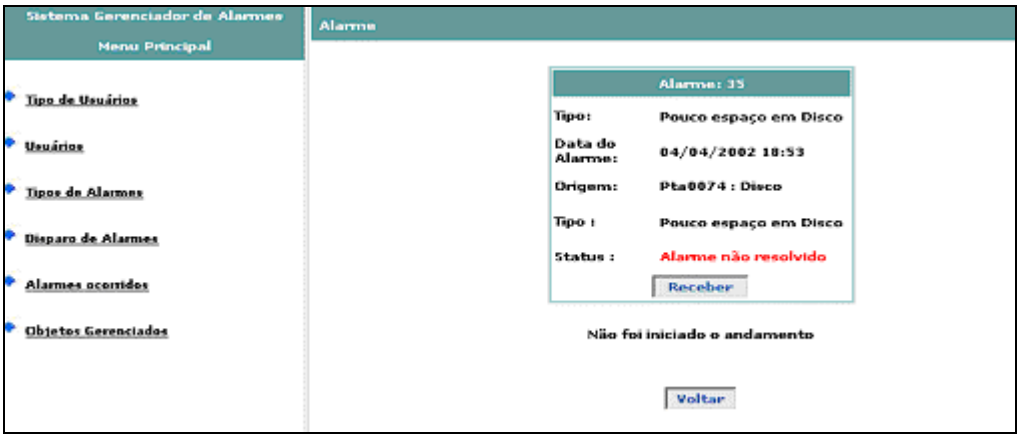


Figura 5.10 - Detalhes do alarme.

A figura 5.10 mostra a tela em que o usuário entra para verificar os detalhes de cada alarme e se desejar ele pode ou não receber o alarme. Caso ele clique em Receber, é guardada a informação do usuário que recebeu o alarme e a hora, tornando-o responsável pela resolução do problema.

Todos os objetos e seus respectivos dispositivos que serão gerenciados pelo sistema deverão ser configurados pela interface WEB. O primeiro passo é indicar o nome de rede do objeto, e para cada um, configurar quais dispositivos deverão ser configurados. O primeiro passo, é entrar com as informações do objeto, como é o caso mostrado na figura 5.11, onde o objeto gerenciado é a máquina de nome PTA0074. Logo na seqüência, inclui-se para esse objeto, os dispositivos que se deseja que sejam monitorados, como pode ser visto na figura 5.12.

Figura 5.11 - Criação do Objeto Gerenciável.

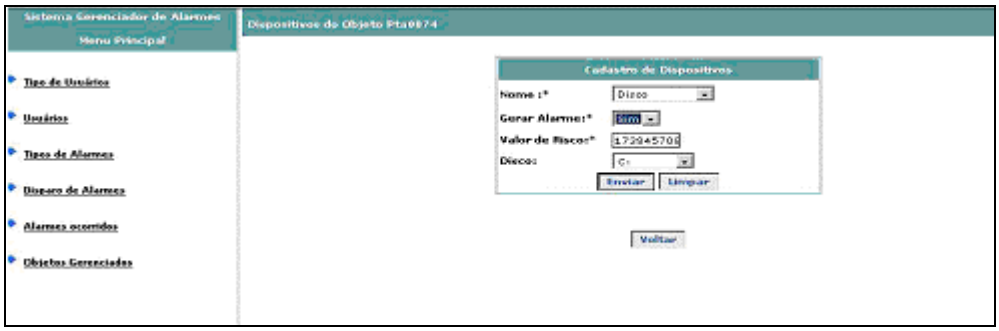


Figura 5.12 - Configuração dos Dispositivos.

5.1.4 Interface WAP

Cada vez mais as pessoas utilizam dispositivos móveis para comunicação, como celulares, *palms* entre outros [STU002]. Os novos celulares já vêm com a possibilidade de navegar pela Internet através do protocolo WAP. Essa facilidade permite que em qualquer lugar que se esteja com um celular WAP e este tendo cobertura do serviço de telefonia da operadora, é possível acessar facilmente os recursos da Internet.

Utilizando desta tecnologia, verificou-se a necessidade da criação da Interface WAP, que também faz a comunicação com o usuário. Com isso, uma pessoa responsável por uma rede, pode ficar sabendo de alguma falha ocorrida, receber esta falha em forma de alarme, mesmo sem ter um computador ligado a rede para verificar o ocorrido.

A interface WAP do sistema foi desenvolvida para agilizar e facilitar o conhecimento de falhas ocorridas no ambiente em que foi instalado o sistema Gerenciador de alarmes.

Para a implementação das páginas WAP, a linguagem usada é o WML, que é baseada no XML (eX*tensible Markup Language*).

O WML permite a integração com algumas linguagens conhecidas para a criação das páginas, ASP e PHP são exemplo dessas linguagens. O ASP foi a melhor escolha para que possa ter a possibilidade de uma fácil integração como os componentes COM já existentes. O desenvolvimento foi feito com WML 1.1 e ASP, usando o Interdev 6.0 da Microsoft para editar o código das páginas ASP, *o EasyPad WapTor* 2.3, para editar as páginas WML e *o SimApp* 2.0 (*Nokia Simulator Application*) para fazer a simulação da aplicação.

Através da interface WAP, um usuário que possuir um telefone que suporte este protocolo, pode instantaneamente à ocorrência da falha, receber o Alarme e tomar as devidas providências para que a falha seja resolvida.

Da mesma forma que na interface WEB, a interface WAP traz uma alternativa rápida e eficiente para acessar o sistema. Na interface WAP, são apresentados todos os alarmes não solucionados, prorrogados e aguardando solução. Após feita a entrada no sistema, também com validação de usuário e senha, é possível alterar o *status* da falha

para receber, prorrogar ou resolver a falha assim como tomar conhecimento dos problemas que estão ocorrendo na rede. Todas essas operações ficam gravadas em histórico para futura geração de relatório.

A seqüência de imagens de 5.13 até 5.16 mostra a seqüência de navegação pela aplicação criada.

Figura 5.13 - Log In no sistema.

Figura 5.14 - Escolha da visualização do tipo de alarme.

Figura 5.15 - Listagem dos alarmes.

Figura 5.16 - Recepção do alarme.

A programação para o protocolo WAP é bastante complicada pois não existem muitos recursos, e para todo código escrito é necessária uma preocupação com alguns fatores essenciais que poderão interferir na performance da página e na apresentação amigável da mesma [LEE01]:

- Taxa de erro elevada;
- Latência elevada;
- Tamanho das telas;
- Dificuldade de digitação.

Dificuldades como estas podem desestimular um usuário a utilizar a aplicação WAP, por isso neste trabalho optou-se por manter poucas funcionalidades na Interface WAP. Para não inviabilizar a utilização do protocolo WAP, foram necessários alguns cuidados especiais:

- **Tamanho da página**: uma página não pode ter muito código ou processamento muito pesado para não prejudicar a eficiência na apresentação das informações;

- **Número de Caracteres**: todo caractere que for impresso na página deve ter um tamanho calculado. O visor de dispositivos *wireless* são bastante restritos quanto a tamanho. Suas interfaces não são nada amigáveis, por isso é necessário ter cautela ao construir uma página WML.
- **Base Reduzida**: ao acessar a base de dados a resposta deve ser rápida, para isso as tabelas não podem ser muito grandes. Para isso, buscou-se uma alternativa para se manter uma tabela pequena mas sem perder informações de histórico. Isso foi feito mantendo os alarmes atuais em uma tabela e a medita que os problemas forem sendo corrigidos, os registros são transferidos para uma tabela auxiliar.
- **Pouca utilização do teclado:** um dos fatores que mais desestimulam a utilização de um sistema baseado em WAP é a necessidade de digitação de textos. Para a implementação deste sistema, houve uma preocupação de diminuir ao máximo o uso do teclado, fazendo-se necessário apenas ao entrar com o *login* e senha, todas as outras funcionalidades são feitas por meio de seleções, assim, o usuário só utiliza o teclado para escolher a opção desejada.

5.1.5 Módulo de Gerência CORBA

Poderia haver alguma situação de já existir algum tipo de aplicação que tenha objetivos parecidos com o gerenciador de Alarmes. Neste caso seria interessante fazer a união das funcionalidades do gerenciador de alarmes com outra aplicação. Para ter esta possibilidade foi criado o módulo de gerência CORBA. Este módulo foi desenvolvido como uma opção de integração do Sistema Gerenciador de Alarmes com outros sistemas. Através do desenvolvimento de um ORB *client* que se comunica com o ORB *server* desenvolvido, qualquer aplicação poderá se integrar com o Sistema Gerenciador de Alarmes, como pode ser visto na figura 5.17. A criação do ORB *Client* deve seguir a interface criada para o ORB *Server* poder interpretar. Nesta interface devem ser enviadas informações como:

- **Tipo**: é o tipo do alarme, que é cadastrado previamente no sistema através da interface WEB;
- **Descrição**: é uma descrição do alarme, que pode ser mandada de forma opcional;
- **Severidade**: indica se o alarme é crítico ou uma simples notificação;
- **Origem**: é a origem do dado, neste caso, pode ser colocada uma *string* fixa como "Gerenciador CORBA";
- **Objeto gerenciado**: deve trazer o nome do objeto gerenciado que gerou o alarme;
- **Dispositivo**: deve trazer qual o dispositivo que está com problemas;
- **IP**: deve trazer o IP do objeto que gerou o alarme.

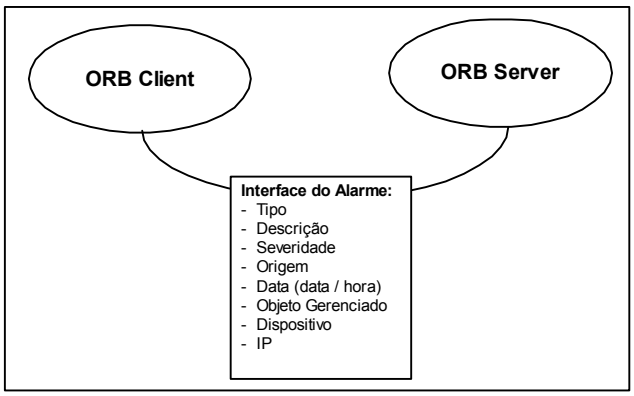


Figura 5.17 - Comunicação do ORB *Client* com o ORB *Server* através da interface criada.

Este módulo foi desenvolvido utilizando a linguagem Java na sua versão JDK 1.3.1. A construção dos ORB's foi feita com o Visibroker 4.5 da Borland [VIS02].

A figura 5.18 ilustra o funcionamento deste módulo. A base representa o módulo CORBA do Sistema Gerenciador de Alarmes. Neste é implementado um ORB *Server* que tem a funcionalidade do recebimento do alarme. Ao receber este alarme, este será avaliado e armazenado em base de dados. Assim que entrar na base, este alarme será tratado pelos demais módulos do sistema. Na parte superior da figura, são representadas as aplicações que podem se comunicar com o servidor.

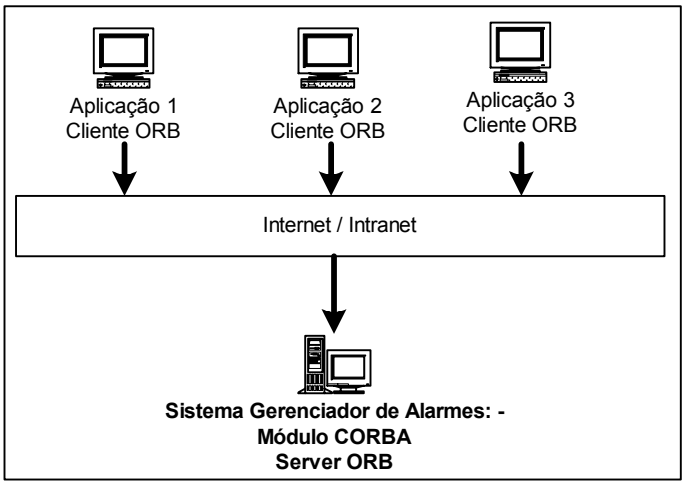


Figura 5.18 - Funcionamento do módulo CORBA.

5.1.6 Módulo de Gerência SNMP

Como o módulo de gerência COM tem um enfoque para redes Windows, a aplicação necessitava de uma opção para o gerenciamento das outras plataformas. Tal funcionalidade é função do Módulo SNMP. O módulo de gerência SNMP é responsável pela coleta e tratamento de *traps* ocorridos na rede.

Traps são mensagens críticas de falhas ocorridas em um dispositivo. Estas são enviadas pelo agente SNMP instalado nos nós de uma rede. Essas mensagens são transmitidas através de PDU's, e delas é possível ter as seguintes informações:

Propriedade	**Descrição**
Versão	Versão da *trap.*
Comunidade	Define a forma de acesso. A comunidade é uma forma de autenticação.
Endereço do Agente	Endereço do agente que gerou a *trap.*
Entidade	Nome do agente que gerou a *trap*
Data	*Time Stamp* da geração da *trap* no formato data.
IP do Agente gerador do trap	Enderço IP do agente gerador da *trap* no formato IP.

Tempo	*Time Stamp* da notificação recebida
Nó	Nome do agente gerador do *trap.*
Mensagem	Descrição da notificação.
Severidade	Serveridade da *trap*
Definição da notificação	Nome da notificação.
Varbinds	É o campo de dados da *trap* SNMPv2. Cada *binding variable* associa uma instância em particular de um objeto com seu valor atual.
TrapOID	Identificador único da *trap.*

O módulo SNMP foi construído utilizando a linguagem Java com a versão JDK1.3.1. A escolha desta linguagem foi feita baseada na existência da AdventNet API SNMPv2c [ADV02]. Esta API fornece as classes Java necessárias para implementar uma aplicação de gerência SNMP.

A função deste módulo é a captação de *traps* SNMP geradas pelos agentes instalados nos nós da rede onde está rodando o Sistema gerenciador de Alarmes. O funcionamento do módulo SNMP pode ser visto na figura 5.19, que mostra no centro o servidor que executa a aplicação. Os computadores ao redor devem possuir um agente SNMP instalado. Esses agentes ao encontrar uma situação de falha, enviarão ao servidor uma notificação, chamada *trap* para o servidor.

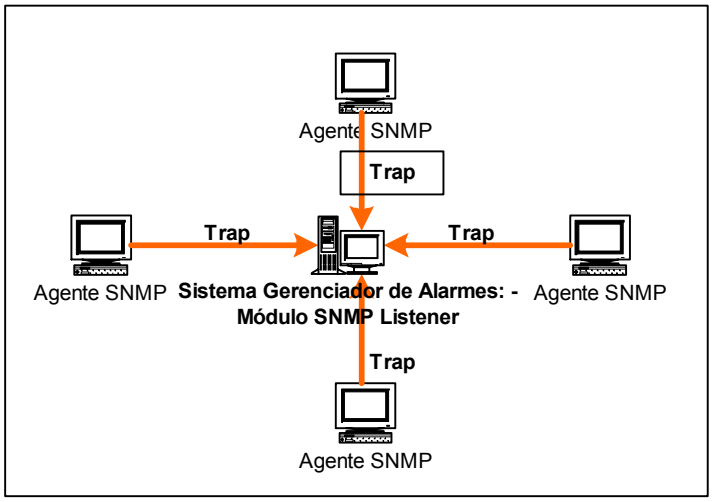


Figura 5.19 - Funcionamento do SNMP *Listener*.

A captação das *traps* é feita pela implementação de um *listener* que é configurado para "ouvir" uma determinada porta do servidor onde é executada a aplicação. Os agentes devem ser configurados para enviar as *traps* para a mesma porta que o Sistema Gerenciador está "ouvindo".

Ao receber uma notificação, o Sistema Gerenciador de Alarmes extrairá todas as informações necessárias, como severidade, descrição, objeto que gerou a *trap* entre outros. Ao extrair as mensagens é feita uma avaliação da severidade da *trap*. De acordo com a severidade, isto é, se o alarme é uma notificação simples ou crítica, a *trap* é armazenada na base de dados. Estando na base de dados, a *trap* já é considerada um alarme e esta é tratada com os outros módulos do sistema, primeiramente o notificador e em seguida o gerador de relatório.

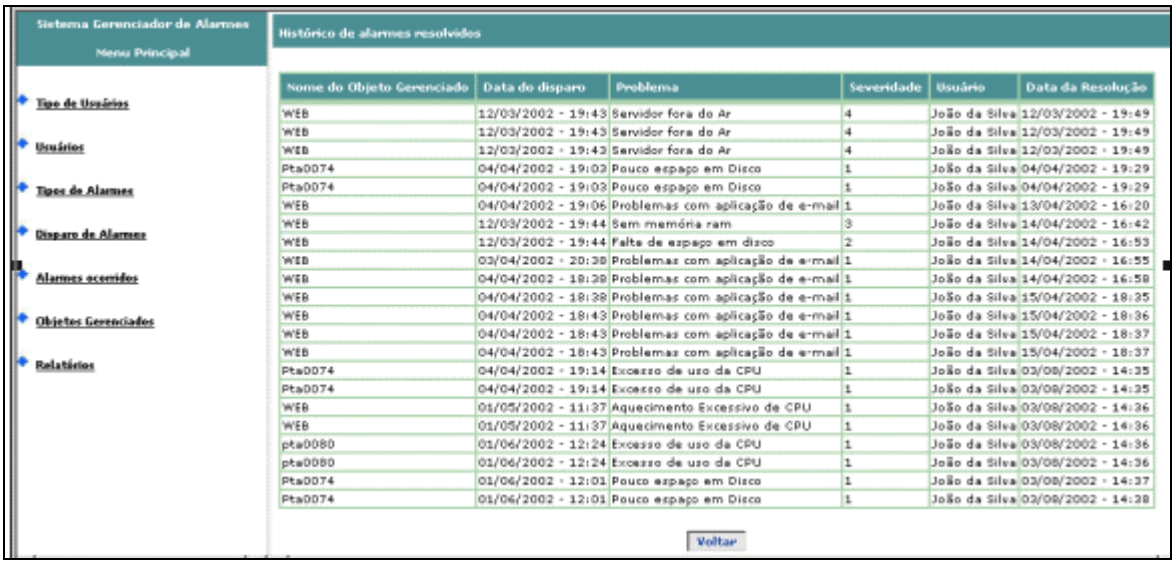
5.1.7 Gerador de Relatórios

É muito importante se manter um histórico de problemas ocorridos na rede e suas soluções. Para que isso seja possível, todas as operações efetuadas no sistema são armazenadas na base de dados com data de ocorrência.

Com esse utilitário, é possível manter relatórios dos alarmes ocorridos e dos objetos gerenciados que mais geraram problemas. Pode ser útil também para avaliar a velocidade que os problemas foram resolvidos e quais os usuários que foram mais e menos eficientes para solucionar os problemas.

Para a versão atual do sistema, foram implementados quatro relatórios básicos:

1. Objetos Gerenciáveis que mais geraram problemas: mostra a contabilização de alarmes gerados para cada objeto configurado no sistema;
2. Usuários que mais resolveram problemas: relatório que contabiliza os usuários que mais solucionaram problemas da rede;
3. Dispositivos que mais geraram problemas: é um detalhamento do primeiro, onde são mostrados para cada um dos objetos, os dispositivos que mais geraram problemas;
4. Histórico de alarmes resolvidos: relatório geral dos procedimentos realizados pelo sistema.



Nome do Objeto Gerenciado	Data do disparo	Problema	Severidade	Usuário	Data da Resolução
WEB	12/03/2002 - 19:43	Servidor fora do Ar	4	João da Silva	12/03/2002 - 19:49
WEB	12/03/2002 - 19:43	Servidor fora do Ar	4	João da Silva	12/03/2002 - 19:49
WEB	12/03/2002 - 19:43	Servidor fora do Ar	4	João da Silva	12/03/2002 - 19:49
Pta0074	04/04/2002 - 19:02	Pouco espaço em Disco	1	João da Silva	04/04/2002 - 19:29
Pta0074	04/04/2002 - 19:03	Pouco espaço em Disco	1	João da Silva	04/04/2002 - 19:29
WEB	04/04/2002 - 19:06	Problemas com aplicação de e-mail	1	João da Silva	13/04/2002 - 16:20
WEB	12/03/2002 - 19:44	Sem memória ram	3	João da Silva	14/04/2002 - 16:42
WEB	12/03/2002 - 19:44	Falta de espaço em disco	2	João da Silva	14/04/2002 - 16:53
WEB	03/04/2002 - 20:38	Problemas com aplicação de e-mail	1	João da Silva	14/04/2002 - 16:55
WEB	04/04/2002 - 18:38	Problemas com aplicação de e-mail	1	João da Silva	14/04/2002 - 16:58
WEB	04/04/2002 - 18:38	Problemas com aplicação de e-mail	1	João da Silva	15/04/2002 - 18:35
WEB	04/04/2002 - 18:43	Problemas com aplicação de e-mail	1	João da Silva	15/04/2002 - 18:36
WEB	04/04/2002 - 18:43	Problemas com aplicação de e-mail	1	João da Silva	15/04/2002 - 18:37
WEB	04/04/2002 - 18:43	Problemas com aplicação de e-mail	1	João da Silva	15/04/2002 - 18:37
Pta0074	04/04/2002 - 19:14	Excesso de uso da CPU	1	João da Silva	03/08/2002 - 14:35
Pta0074	04/04/2002 - 19:14	Excesso de uso da CPU	1	João da Silva	03/08/2002 - 14:35
WEB	01/05/2002 - 11:37	Aquecimento Excessivo de CPU	1	João da Silva	03/08/2002 - 14:36
WEB	01/05/2002 - 11:37	Aquecimento Excessivo de CPU	1	João da Silva	03/08/2002 - 14:36
pta0080	01/06/2002 - 12:24	Excesso de uso da CPU	1	João da Silva	03/08/2002 - 14:36
pta0080	01/06/2002 - 12:24	Excesso de uso da CPU	1	João da Silva	03/08/2002 - 14:36
Pta0074	01/06/2002 - 12:01	Pouco espaço em Disco	1	João da Silva	03/08/2002 - 14:37
Pta0074	01/06/2002 - 12:01	Pouco espaço em Disco	1	João da Silva	03/08/2002 - 14:38

Figura 5.20 - Página do relatório de histórico.

A figura 5.20 mostra um relatório gerado pelo sistema do histórico geral dos alarmes resolvidos. Esse relatório é mantido na base de dados, possibilitando futuras consultas.

5.2 Ambiente de Validação

Para fazer a validação da aplicação implementada, foi criado um ambiente de redes utilizando uma parte dos computadores localizados na Paradigma. Este ambiente é composto de:

- **1 servidor**
  - Processador: K7; 800 Mhz
  - Memória: 128 Mbs
  - SO: Windows 2000 *Server*
- 2 computadores
  - Processador: K6; 700 Mhz
  - Memória: 128 Mbs
  - SO: Windows 2000 *Professional*

- 1 computador
  - Processador: TunderBird; 850 Mhz
  - Memória: 128 Mbs
  - SO: Windows 2000 *Server*
- Simulador WAP: Nokia SimulationApplication 2.0, para simular um celular

O primeiro passo foi fazer a verificação do ambiente. Para permitir que o Módulo COM funcionasse de maneira adequada, o ambiente de rede deveria estar bem configurado. Isto significa que há necessidade de realizar às seguintes verificações:

- Os equipamentos devem estar em rede e se comunicando entre si;
- O serviço WMI deve estar iniciado, que já é o padrão do Windows 2000, e o usuário do serviço deve ser um administrador do computador. O ideal é que a senha deste usuário nunca expire, para evitar problemas futuros.
- Instalar como um adicional nestes computadores o agente SNMP, que já presente no Windows 2000, mas não é um padrão de instalação.

Feitas essas verificações, a aplicação pode ser instalada. A instalação foi feita da seguinte forma:

- As páginas ASP e WML são publicadas no IIS 5.0, da máquina servidora Windows 2000, isto é, na máquina pta0080.
- Os componentes COM, ficam num diretório qualquer do servidor e são registrado no servidor de Aplicações COM+[6], também da pta0080[7] .
- O SNMP *trapListener* deve estar executando, como um serviço.
- O ORB *Server* do sistema gerenciador também deve estar executando.
- A base de dados foi criada no próprio servidor [8], utilizando o SGBD SQLServer 2000.
- O arquivo de configuração da aplicação deve estar configurado para acessar o servidor de base correto. Para isso deve ser indicado o nome do servidor, a base e o usuário para acessá-la, como pode ser visto na figura 5.21. Este arquivo pode ser encontrado no diretório onde se encontra o componente **conexao.dll.**

```
[Conexao]
Server=Pta0080
Base=gerenciador
Usuario=sa
Senha=
```

Figura 5.21 - Configuração do arquivo gerencia.ini.

Com a aplicação instalada, o próximo passo foi fazer a configuração dos objetos gerenciáveis, indicando quais os dispositivos que deveriam ser gerenciados e quais os valores limites para cada dispositivo. Essa configuração pode ser vista na figura 5.22.

---

[6] Se o servidor for o Windows NT, a instalação é feita no MTS (Microsoft Transaction Server)

[7] Os componentes poderiam ser colocados em um ambiente distribuído, através da criação de um pacote proxy para ser instalado em outros servidores. Este processo é viabilizado pelo protocolo DCOM.

[8] Optou-se por criar a base no próprio servidor pela facilidade de manutenção. Mas poderia ser instalado em qualquer máquina da rede que tivesse o SQL Server instalado. Para isso, basta alterar o arquivo de configuração da aplicação, indicando o nome do servidor, base e senhas para acessar esta base.

Os valores limites escolhidos para os testes estão abaixo dos valores que realmente representariam uma falha grave no sistema. Este fato é justificado pelas máquinas escolhidas não apresentarem nenhum problema com estas variáveis no tempo estimado para os testes. A solução foi a configuração dos limiares a baixo do normal, para simular uma situação de erro.

OBJ1	OBJ1	OBJ1	OBJ1
Disco: 1738457088 **Temperatura:** 30 ºC **CPU:** 12% **Memória:** 12%	**Disco:** 1347000000 **Temperatura:** 30 ºC **CPU:** 30% **Memória:** 100%	**Disco:** 15000000000 **Temperatura:** 30 ºC **CPU:** 20% **Memória:** 50%	**Disco:** 4000000000 **Temperatura:** 80 ºC **CPU:** 80% **Memória:** 35%

Figura 5.22 - Configuração dos valores limites dos dispositivos.

De forma a permitir maior diversidade nos testes relacionados aos módulo SNMP e CORBA, foram implementadas algumas ferramentas de teste. No caso do módulo SNMP, foi implementado um utilitário que envia *traps* para uma determinada porta, que deve ser a mesma porta que o módulo SNMP recupera as *traps*. Desta maneira é possível gerar qualquer tipo de *trap* para testar o sistema. Através deste utilitário o sistema não fica na dependência de um recurso vir a gerar uma falha, o que pode não ocorrer na fase de testes. Já para testar o módulo CORBA, foi criado um ORB *client*, que enviam para o ORB *server* alguns alarmes criados para efeito de teste.

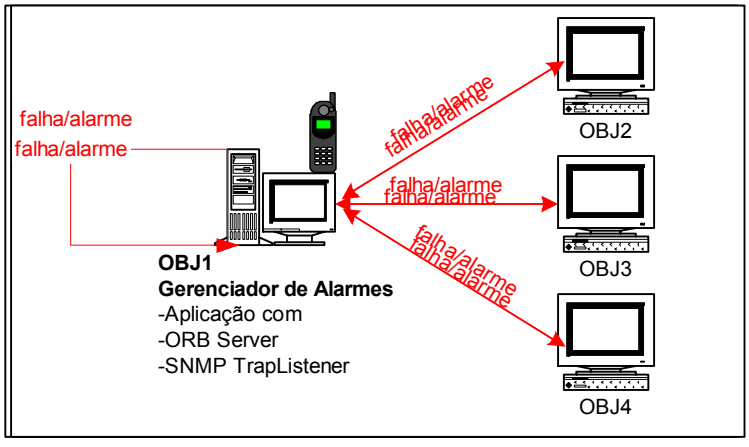


Figura 5.23 - Ambiente de Validação.

A figura 5.23 representa a maneira como estava disposto o ambiente testado. No servidor, representado pelo OBJ1, estava instalada toda a aplicação. Este servidor era configurado para a cada 5 minutos monitorar os equipamentos OBJ1,OBJ2, OBJ3 e OBJ4 e verificar se havia ocorrido alguma falha. Este monitoramento era feito pelo acionamento repetido do Módulo COM.

As falhas provenientes da ação do módulo SNMP, foram recuperadas através do aplicativo gerador de *traps*. Vários tipos de *traps* foram geradas e tratadas de acordo com a severidade da mesma. Caso a *trap* fosse uma simples notificação, essa não entraria no sistema.

Já as falhas provenientes do Módulo CORBA, foram recuperadas através do ORB *Client* criado para este propósito. Vários alarmes foram criados a partir deste ORB. Só os alarmes definidos como prioridade média, alta e críticos entraram no sistema.

Assim que os alarmes entravam no sistema, automaticamente o módulo notificador era acionado. De acordo com a severidade dos alarmes, foram sendo chamados os diferentes tipos de notificação. No caso de um alarme crítico, foi criada uma repetição de notificação, que só era desfeita assim que o sistema recebesse a informação que o alarme foi recebido. Mas surgiu um problema na notificação e houve a necessidade do tratamento da seguinte situação: quando uma falha era detectada, e depois de 5 minutos a mesma não fosse resolvida, os responsáveis estariam recebendo duas vezes a notificação do mesmo alarme. Para resolver esta situação a aplicação teria que fazer a verificação de que o alarme já estava cadastrado, e foi padronizado que só voltaria a ser cadastrada no sistema depois de 12 horas. Resolvendo este problema a aplicação teve uma boa atuação.

5.3 Resultados

A aplicação se mostrou bastante útil no que se propôs a fazer, mas algumas considerações devem ser feitas.

Quanto ao módulo Gerenciador COM foi possível alcançar resultados bastante vantajosos no ambiente criado, pois sua proposta é a geração de alarmes para problemas bem específicos. Desta forma, não há a possibilidade da criação de um número excessivo de alarmes. Só serão gerados alarmes se o limiar dos valores limite de cada dispositivo for atingido.

A recuperação da maioria das informações foi feita de maneira bastante eficiente. O único problema surgiu com a recuperação dos valores de utilização de memória. Dois dos equipamentos do ambiente proposto, não retornaram valores. Este problema ocorreu pelo não reconhecimento dessa memória. Simplesmente o WMI não retornou esta informação.

O módulo SNMP foi uma escolha importantíssima para fazer parte da aplicação, uma solução completa para o gerenciamento de falhas. O protocolo SNMP tem uma característica muito vantajosa de geração de *trap* pelo agente instalado nos equipamentos da rede. Uma *trap* somente é gerada na detecção de uma falha. Desta forma não é necessário que a aplicação faça o *pooling*, que é a verificação de equipamento em equipamento em busca de falhas. Este é um diferencial em relação ao WMI, que não possui essa funcionalidade. Por esta razão, é necessário configurar o Módulo COM a monitorar a rede de tempos em tempos para a detecção de falhas. Esta característica pode ser um problema no caso da rede estar congestionada, mas de maneira nenhuma inviabiliza o funcionamento da aplicação.

Já o módulo CORBA foi uma idéia interessante. Ele funciona de forma parecida com o Módulo SNMP, mas os alarmes serão gerados por um ORB *Client* que poderá ser implementado de qualquer forma para gerar o alarme, contanto que haja a comunicação com o ORB *Server* implementado. A comunicação entre os ORB´s é garantida através da correta configuração da interface que o ORB *Server* entende. Isso permite que qualquer aplicação que detecte uma falha, se comunique com o gerenciador criado. Como não é usado o *pooling,* se não houver os *clients* enviando alarmes, nenhum problema ligado a tráfego na rede será causado.

Depois de realizados todos os testes, foi observado que neste protótipo criado poderiam ser feitos alguns melhoramentos. Dentre eles podem ser citados:

- Além do monitoramento e geração de alarmes, a aplicação poderia permitir ao usuário tomar algumas atitudes. O WMI disponibiliza várias formas de fazer

esse tipo de implementação. Mas esse melhoramento sairia do escopo do trabalho.

- Configuração de filtros para *traps* geradas pelo módulo SNMP através da interface WEB. Esse tipo de melhoramento é muito útil, mas necessitaria de mais conhecimento do usuário que estivesse configurando a aplicação. Também como uma boa opção seria simplesmente colocar estes filtros fixos no código.
- Geração de ORB's *Client* padrões. Isso seria uma opção para a comercialização da aplicação.

5.4 Conclusão

O ambiente a ser gerenciado é formado de equipamentos com várias plataformas, sendo a plataforma Windows dominante. Para obter uma gerência efetiva de um ambiente com estas características, a arquitetura de desenvolvimento precisaria oferecer uma boa integração com as redes Windows e também a possibilidade de integração com outras tecnologias necessárias para a gerência das demais plataformas. Isto se tornou possível com a utilização do conceito de componentes. Alguns componentes foram construídos para o gerenciamento de recursos com a plataforma Windows e outros com a função de disponibilizar ao usuário as informações de forma a unificar as informações de gerência captadas por diferentes tecnologias. Para melhor organizar estes componentes, foram criados os módulos do sistema. Em cada um dos módulos gerentes foram utilizadas tecnologias diferentes como o WMI, SNMP e CORBA. Com essas diferentes tecnologias o ambiente pode ser gerenciado de maneira completa e eficiente. As informações captadas por estes módulos foram unificadas através da base de dados e componentes COM de acesso a mesma. Estes componentes interagem com o usuário através das interfaces WEB e WAP do sistema.

Apesar do bom funcionamento da aplicação, algumas particularidades puderam ser observadas ao comparar módulos gerentes utilizados para recuperar informações

relacionadas a falhas no sistema. A escolha do WMI foi fundamental para se ter o resultado esperado do ambiente. O fato de se poder implementar especificamente quais os alarmes a serem gerados, foi importante para a diminuição do número de alarmes gerados. Este fator é essencial para a viabilização da aplicação. Ao contrário do SNMP, que gera *traps* que podem ser simples notificações ou problemas críticos. Para este módulo, foi necessária a criação de filtros, para que o gerenciador recebesse somente *traps* realmente importantes. Já no módulo CORBA não existe este problema, pois os alarmes são gerados pelo ORB *Client* por isso é a responsabilidade deste gerar apenas alarmes importantes. Mas por garantia, existe um filtro pela severidade dos alarmes gerados pelo módulo CORBA.

Outra consideração é o fato do WMI só permitir que sejam recuperadas informações do recurso através do *pooling*. Este fato pode vir a ser um ponto negativo quando a rede está congestionada, mas isso não inviabiliza o bom funcionamento da aplicação. Esta já é uma vantagem dos módulos SNMP e CORBA. No módulo SNMP são recuperadas as *traps* enviadas pelos agentes, sem a necessidade do *pooling*. Assim, uma *trap* é gerada somente quando realmente existe o problema. Da mesma forma funciona o Módulo COM, que tem os ORB *Client* em cada equipamento, assim que for detectada uma falha, o ORB *Client* envia o alarme para o ORB *Server*, sem utilização excessiva de tráfego na rede.

A criação destes três módulos gerentes proporcionou um gerenciamento bastante completo do ambiente com relação a falhas do sistema. Com a extensão desses módulos é possível a construção de uma ferramenta ainda mais completa, para que além do gerenciamento de falhas, se tome decisões de gerência.

6 Conclusão

À medida que as redes cresceram e tornaram-se integradas às organizações, elas passaram a fazer parte do cotidiano dos usuários como uma ferramenta que oferece recursos e serviços que permitem a interação e o aumento de produtividade. A conseqüência natural dessa situação é o crescimento significativo da importância gerencial de redes de computadores para mantê-las funcionando de forma adequada.

Com base nos padrões, como SNMP, CMIP, WBEM, criados para executar a gerência destas redes, foram desenvolvidas pelo mercado, as ferramentas destinadas ao gerenciamento de redes, que atualmente alcançaram um padrão de abrangência considerado satisfatório além de uma boa gama de funcionalidades, mas que ainda exigem um investimento financeiro um pouco alto além de requererem um treinamento de pessoas especializadas. Sua aplicabilidade fica restrita a grandes corporações que contam com disponibilidade de recursos para fazerem frente a investimento dessa grandeza.

Na outra ponta encontram-se instituições que não se dispõem a arcar com os custos ou mesmo que não encontrem funcionalidades consideradas fundamentais para seus ambientes. Isso acarreta um grande número de trabalhos existentes na área de gerência de redes, focados em ambientes específicos.

Desta forma, este trabalho se concretizou com base nas necessidades de um ambiente heterogêneo, mas que tem em usa essência computadores com a plataforma Windows, cujo "modos operandi" baseia-se no desenvolvimento de soluções de comércio eletrônico público e privado onde o "*time to market*" é o principal diferencial em relação a concorrência.

Ambientes com essas características não podem ser susceptíveis a falhas constantes sob risco de comprometimento total de sua proposta de mercado. Por estas razões, esta dissertação propôs uma aplicação eficiente e inovadora para gerência de falhas em redes de computadores assim caracterizados.

O objetivo básico desta aplicação foi prover uma infra-estrutura eficiente para a detecção e notificação de falhas, assim como o acompanhamento das eventuais soluções para o problema.

Para se obter êxito nesse propósito, foi necessária a utilização de várias tecnologias, como a programação WEB, programação para dispositivos WAP, COM além dos padrões SNMP, WBEM e CORBA. Mas para permitir que fossem utilizadas todas essas tecnologias, a aplicação foi desenvolvida seguindo os conceitos de componentes, onde são separadas as funcionalidades do sistema em componentes. Esta opção foi fundamental para a criação da aplicação, pois seguindo estes conceitos foi possível criar uma arquitetura em três camadas, isto é, interface com o usuário, regra de negócios e camada de dados.

As funcionalidades foram implementadas na camada de regras de negócio. A união das informações de gerência recuperadas pela utilização dessas tecnologias na camada de negócios, puderam ser apresentadas de forma transparente ao usuário através das camadas de interface e camada de dados.

Na camada de interface do usuário foram usadas a WEB e WAP. Esta opção de interface foi essencial para ter uma interação bastante amigável com o usuário e permitir que a informação chegasse rapidamente ao usuário e conseqüente resolução do problema.

A única preocupação desta camada da aplicação, foi para a construção da interface WAP. Pelas características *wireless*, como alta taxa de erros, pouca largura de banda entre outros resultou em uma série de cuidados com a implementação. Mas codificando páginas pequenas, que não sejam demoradas para carregar, não tenham acessos exagerados ao banco, foi possível obter uma interface enxuta e crucial para obter uma maior rapidez no recebimento da informação e resolução do problema

Foi observado que nem sempre a interface é uma preocupação presente em alguns sistemas. Normalmente estes sistemas são desenvolvidos para a resolução dos problemas sem se preocupar com usuários que não sejam dotados de conhecimento suficientes para a utilização do sistema. Este diferencial foi alcançado com sucesso na aplicação desenvolvida por este trabalho.

Na camada de negócios do sistema, foram implementadas todas as funcionalidades de gerência e dos processos do sistema. Estas funcionalidades foram divididas em componentes que tem responsabilidades como configuração do sistema, criação de usuários, implementação da segurança das páginas da interface WEB, as gerências e o sistema de notificação. A junção destes componentes para um objetivo

único formaram os módulos, sendo eles o Módulo de Gerência COM, Módulo de Gerencia SNMP, Módulo de Gerência CORBA, Módulo Notificador e Gerador de Relatórios.

Nos módulos de gerência foram utilizadas as tecnologias WMI, SNMP e CORBA.

O módulo COM gerencia os principais problemas que podem ocorrer em uma rede Windows, sendo estes problemas de *software* ou *hardware*. Sua atuação dentro de recursos com a plataforma Microsoft se mostrou extremamente útil. Os testes mostraram que ao utilizar uma tecnologia integrada a ao sistema operacional Windows, a eficiência é bem maior, pois ao executar um componente implementado para interagir com o serviço WMI, este tem direitos de administrador sob a máquina gerenciada, podendo resgatar qualquer informação necessária para o reconhecimento de falhas.

Outro fato importante é a facilidade de extensão da aplicação através do WMI. O WMI permite que se faça o gerenciamento total de uma máquina windows, inclusive tarefas como listagem dos serviços que estão rodando, parar e reiniciar estes serviços, entre muitas outras. Com a implementação feita com os recursos da Microsoft, no caso Visual Basic 6.0, foi relativamente simples a construção deste módulo. Uma grande desvantagem encontrada, é o fato de o WMI não ter nenhum tipo de integração com outros sistemas operacionais, por este motivo foi necessária a implementação de outros módulos que viessem a suprir esta deficiência, sendo necessária a construção dos módulos SNMP e CORBA.

O módulo SNMP foi construído com o objetivo de captação de *traps* ocorridos em qualquer equipamento, independente de plataforma, que tenha o agente SNMP instalado. A grande vantagem deste módulo é pela proliferação do protocolo SNMP, que por ser o mais usado, costuma ser incluído em grande parte dos equipamentos.

Com a utilização deste módulo, a aplicação pôde ser completa. Com a implementação em Java, foi possível a fácil manipulação das *traps* recebidas e conseqüentemente a interação com os outros módulos, que têm a dependência de informações básicas das falhas. A execução deste módulo se mostrou bastante eficiente principalmente pelo filtro implementado, que não permite a geração de um número muito elevado de alarmes. Como este módulo foi implementado para ouvir uma porta,

processar a falha e em seguida passar o processamento para os outros módulos do sistema.

Não ouve a necessidade da criação de uma terceira interface alem da WEB e WAP. Este fato fez com que o módulo SNMP não exigisse muito da configuração de um equipamento, como por exemplo memória RAM, o que é uma grande vantagens sob grandes aplicações feitas em Java.

Já o Módulo CORBA, foi criado para que seja utilizado como uma forma de integração do Gerenciador de Alarmes com outras aplicações. Com esta funcionalidade qualquer tipo de alarme criado por qualquer aplicação, pode ser facilmente gerenciado pelo gerenciador de alarmes. Este módulo se mostrou bastante eficiente e simples de implementar com a utilização do [VIS02].

A união das informações recuperadas por esses módulos foi feita pela camada de dados, isto é pela base de dados, e pela camada de interface. Desta forma ficou transparente ao usuário a utilização dessa variedade de tecnologias.

A junção dessas tecnologias em uma só aplicação tornou-a bastante eficiente podendo ser utilizada em qualquer instituição que tenha um servidor Windows, pois possibilitará a gerência de qualquer equipamento.

Com a execução da aplicação foram comparados os módulos e algumas considerações puderam ser feitas como a escolha do WMI, que foi fundamental para se ter o resultado esperado do ambiente.

O fato de se poder implementar especificamente quais os alarmes a serem gerados, foi importante para a diminuição do número de alarmes gerados. Este fator foi essencial para a viabilização da aplicação. Ao contrário do SNMP, que gera *traps* que podem ser simples notificações ou problemas críticos. Para este módulo, foi necessária a criação de filtros, para que o gerenciador recebesse somente *traps* realmente importantes. Já no módulo CORBA não existe este problema, pois os alarmes são gerados pelo ORB *Client* por isso é a responsabilidade deste gerar apenas alarmes importantes. Mas por garantia, existe um filtro pela severidade dos alarmes gerados pelo módulo CORBA.

Outra consideração é o fato do WMI só permitir que sejam recuperadas informações do recurso através do *pooling*. Este fato pode vir a ser um ponto negativo quando a rede está congestionada, mas isso não inviabiliza o bom funcionamento da

aplicação. Esta já é uma vantagem dos módulos SNMP e CORBA. No módulo SNMP são recuperadas as *traps* enviadas pelos agentes, sem a necessidade do *pooling*. Assim, uma *trap* é gerada somente quando realmente existe o problema. Da mesma forma funciona o Módulo COM, que tem os ORB *Client* em cada equipamento, assim que for detectada uma falha, o ORB *Client* envia o alarme para o ORB *Server*, sem utilização excessiva de tráfego na rede.

A solução apresentada nesta dissertação difere do enfoque de outros trabalhos realizados na gerência de falhas. Geralmente os trabalhos como [KET02], e [STE01] são focados na correlação de alarmes, já nesta dissertação, são buscadas falhas específicas.

Ambos enfoques são importantes e têm vantagens e desvantagens, por exemplo, no Gerenciador de Alarmes, se houver a necessidade da verificação de algum dispositivo não configurado, haverá a necessidade de alterar a implementação. Por outro lado os erros que mais causam problemas de inviabilização do funcionamento rede, serão os que estão implementados.

Isto significa que o número de alarmes é menor, podendo ser monitoradas as pessoas que os resolveram através dos relatórios dos históricos. Já na correlação de alarmes, o numero excessivo de registro tornaria impossível tal acompanhamento.

Pela implementação relativamente simples viabilizada pelas linguagens Visual Basic, e Java, a aplicação ficou enxuta, modularizada e conseqüentemente com baixo custo, o que foi crucial para o ambiente proposto, proporcionando uma fácil monitoração da rede, evitando que falhas comuns tragam problemas para os usuários da rede.

6.1 Trabalhos Futuros

Como sugestões para trabalhos futuros a serem realizados a partir do tema deste trabalho, podem ser citados:

1. Ampliação do sistema gerenciador de alarmes para um gerenciador de redes, utilizando as vantagens oferecidas pelo WMI unidas pelo SNMP e CORBA. Essa ampliação pode ser no sentido de permitir tomadas de decisões a partir da WEB e dispositivos WAP. Podendo até ser criado um algoritmo para tomar essas decisões de forma automática para alguns casos.
2. Fazer a união dessas tecnologias diretamente na arquitetura WBEM, através de mapeamentos do modelo CIM com alguma outra tecnologia, por exemplo CORBA onde as informações que foram armazenadas em base de dados pudessem ser acessados através da criação desta aplicação, sem perder as funcionalidades já implementadas do WMI.
3. Criar a aplicação utilizando a extensão do modelo CIM como no item 2 e em seguida fazer a verificação da performance das duas aplicações com objetivo de concluir qual seria a melhor maneira, de se utilizar tais tecnologias.

4. Ampliar a aplicação para servir de auxilio para a construção de SLA (*Service Level Agreement*), onde ao definir as variáveis, poderão ser gerados relatórios específicos para auxiliar neste tipo de contrato.

Referências Bibliográficas

[ABD02] ABDUL-FATAH, Istabrak.; MAJUMDAR, Shikharesh. Performance of CORBA-based client-server architectures. **IEEE Transactions on Parallel and Distributed Systems,**
[s.l], v. 13, n. 2 Fevereiro. 2002, 17 p., p. 111 –127.

[ADV02] ADVENTNET SNMPv2cAPI. Disponível em: http://www.adventnet.com/products/snmp/help/index.html acessado em 25 de julho de 2002.

[BEN00] BENECH, D.; JOCTEUR-MONROZIER, François; RIVIERE, Anne-Isabelle. Supervision of the CORBA environment with SUMO: a WBEM/CIM-based management framework. International Symposium on Distributed Objects and Applications, 2000, Antwerp, Bélgica. **Proceedings…**[s.l]: IEEE Press. 10 p., p. 241 – 250.

[BER94] BERNERS-LEE, T ; MASINTER, L; MCCAHILL, M. Uniform Resource Locators (URL). Request for comments: 1738, Internet Engineering Task Force, dezembro, 1994 .

[BOO00] BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. UML Guia do Usuário. ed. Campus, 2000.

[BOU02] BOUTABA, Raouf.; POLYRAKIS, A Andreas. Projecting advanced enterprise network and service management to active networks. **IEEE Network.** [s.l.]. v. 16, n. 1, p. 28 –33,
Jan.-Fev. 2002 .

[BRO99] BROWN, Steve. Visual Basic6: Bíblia do Programador.1.ed. São Paulo: Berkley, 1999. 609 p.

[CAN01] CANNATARO, Mario; PASCUZZI, Domenico. A Component-Based Architecture for the Development and Deployment of WAP-Compliant Transactional Services. 34th Hawaii International Conference on System Sciences, 2001, Hawaii. **Proceedings…**[s. n.] : IEEE Press, 2001.

[DIA00] DIAS, Adilson de Souza. WAP- Wireless Application Protocol: A Internet Sem Fios .1. ed. Ciência Moderna Ltda., 2000.

[DMT02] DMTF. WBEM. Disponível em <http://www.dmtf.org/standards/standard_wbem.php >. Acesso em 12 maio de 2002-05-17.

[FES99] FESTOR, O.; FESTOR, P.; YOUSSEF,, Ben N.; ANDREY, Laurent. Integration of WBEM-based management agents in the OSI framework IFIP/IEEE International Symposium on Distributed Management for the Networked Millennium, 1999
Proceedings …[s.l]: IEEE Press. 16 p., p 49 – 64.

[FIE97] FIELDING, R.; GETTYS, J.; MOGUL J. ; FRYSTYK H.; BERNERS-LEE T. . Hypertext Transfer Protocol - HTTP/1.1. Request for comments: 2068. Internet Engineering Task Force, Janeiro de 1997.

[FRE96] FREED, N.; KLENSIN, J.; POSTEL, J. Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures. Request for comments:2048. Internet Engineering Task Force, Novembro de 1996.

[GAR98] Voth, Gary.R.; Kindel, Charles; Fujioka, Jon. Distributed application development for three-tier architectures: Microsoft on Windows DNA **IEEE Internet Computing**, [s.l], v. 2 n. 2, Março – Abril 1998 , p. 5, p. 41 –45.

[GRI00] GRIGORAS, Dan; MIHAILA, Stefan. A framework for component-based distributed applications design. The CODE: Component Oriented

Distributed Environment. Conference on Parallel Computing in Electrical Engineering, 2000. **Proceedings…**[s.l.]: IEEE Press. 5p., p. 8 –12 .

[GUI01] GUIAGOUSSOU, M.H.; BOUTABA, R.; KADOCH, M. A Java API for advanced faults management. IEEE/IFIP International Symposium on Integrated Network Management, 2001, Seattle **Proceedings…** [s.l.]: IEEE Press, 2001 16 p., p. 483 –498.

[HAG02] HAGLEITNER Markus. ; MUECK Thomas. Hawaii International Conference on System Sciences, 32, 2002 Big Island. **Proceedings…** [s.n]: IEEE Press, 2002. 10 p. Disponível em : http://computer.org/Proceedings/hicss/1435/volume3/14350088abs.htm.

[HON01] HONG, Liu; DONG, Bai; WEI, Ding . The integration of SNMP and web in embedded devices. International Conferences on Info-tech and Info-net, 2001, Beijing. **Proceedings**… [s.l.]: IEEE Press, 2001, 5 p., p. 83 –87.

[HUL01] HULL, M.E.C.; NICHOLL, P.N.; BI, Y. Approaches to component technologies for software reuse of legacy systems. **Computing & Control Engineering Journal,** [s.l], v. 12 n. 6, Dez. 2001, p. 7, p. 281 -287.

[IBM01] IBM Tivoli. User´s Guide. Disponível em < http://www.tivoli.com/support/public/Prodman/public_manuals/td/netview/SC31-8888-00/en_US/PDF/dwml1mst.pdf> Acesso em 10 maio de 2002.

[KET02] KETTSCHAU, H.-J.; BRUCK, S.; SHEFCZIK, P. LUCAS - an expert system for intelligent fault management and alarm correlation. Network Operations and Management Symposium, 2002. **Proceedings…**[s.l]: IEEE Press. 3p., p. 903 –905.

[KRI01] KRINTZ, Chandra; WOLSKI, Rich. NwsAlarm: a tool for accurately detecting resource performance degradation. Symposium on Cluster Computing and the Grid, 2001. **Proceedings…**[s.l]:IEEE Press, 2001.

10p., p. 404 –413.

[LEE01] LEE, Sungwon; SONG, Nah-Oak Experimental WAP (wireless application protocol) traffic modeling on CDMA based mobile wireless network. Vehicular Technology Conference, 53, 2001. **Proceedings…** [s.l.]: IEEE Press., 2001 5 p., p. 2206 -2210 v.4.

[LIM01] LIMA, Abiel Roche .Proposta e Validação de um Mecanismo para Garantir QoS em Redes Sem Fio de Topologia *ad hoc*.2001 Dissertação de mestrado em Ciência da Computação, Universidade Federal de Santa Catarina, Florianópolis.

[MAC01] MACHADO, Christiano.M.;SAFE, Geórgia.P.; NOGUEIRA, José Marcos.S.; LOUREIRO, Antonio A.F. On the impact of using Web interfaces in a distributed management platform. EEE/IFIP International Symposium on Integrated Network Management, 1, Seattle, 2001. **Proceedings…**[s.l]:IEEE Press. 2001 p. 4, p. 317 – 320.

[MAL01] MALOWIDZKI, Marek. The management of the mobile network with COM+ and SNMP. Military Communications Conference,2001, Washington . **Proceedings…**[s.l.]: IEEE Press, 2001. 5 p., p. 1456 –1460.

[MET00] METTER, Marcin ; COLOMB, Rorbert. WAP Enabling Existing HTML Applications. First Australasian User Interface Conference,2000, Camberra, AU. **Proceedings…** [s. n.] : IEEE Press, 2000. 9 . Disponível em http://computer.org/Proceedings/auic/0515/05150049abs.htm.

[MIC00] MICROSOFT,Corporation. Windows Management Instrumentation: Background and Overview. Disponível em: <http://www.msdn.microsoft.com/library/ . > Acesso em: 18 abril 2002.

[MIC96] MICROSOFT CORPORATION. DCOM: Tecnical Overview. Disponível

em http://www.msdn.microsoft.com/library/. Acessado em 20 setembro de 2001.

[MIC99] MICROSOFT, Corporation. Introducing Windows DNA: Framework for a New Generation of Computing Solutions. Disponível em: <http://www.msdn.microsoft.com/library/ > .Acesso em: 25 abril 2002.

[MIL01] MILANOVIC, Nadza.; MORNAR, Vedran. A software infrastructure for distributed computing based on DCOM. International Conference on Information Technology Interfaces, 2001, Pula, Croacia . **Proceedings…**[s.l] :IEEE Press, 2001, 6 p., p. 63 – 68.

[MIT00] MITZEL,D.. IAB Wireless Internetworking Workshop. Request for Comments: 3002. Internet Engineering Task Force, dezembro de 2000.

[MUE00] MUELLER, John Paul. COM+ Developer´s Guide. 1. ed. São Paulo: Osborne/McGraw-Hill, 2000. 487 p.

[NOK01] NOKIA, Networks. WAP over GPRS: Realizing the potential of mobile services. Disponível em: <http://www.nokia.com > Acesso em 5 outubro 2001.

[NTO02] NTOP. NTOP – Network TOP. Disponível em < http://www.ntop.org/> Acesso em 11 maio de 2002.

[OIN01] OINAS, Arto. Defining Goal-Driven Fault Management Metrics in a Real World Environment: A Case-Study from Nokia**.** 5thConference on Software Maintenance and Reengineering, 2001, Lisboa, PT . **Proceedings…** [s. n.] : IEEE Press, 2001. Disponível em http://computer.org/Proceedings/csmr/0546/05460101abs.htm?SMSESSION=NO .

[OJA00] OJANEN, Eetu**;** VEIJALAINEN, Jarí. Compressibility of WML and WMLScript Byte Code: Initial Results. 10th International Workshop on Research Issues in Data Engineering, 2000 San Diego, CA. **Proceedings…** [s. n.] : IEEE Press, 2000. 8 Disponível em : <http://computer.org/Proceedings/ride/0531/05310055abs.htm>.

[OMG02] OMG. CORBA/IIOP Specification. Disponível em: < http://cgi.omg.org/docs/formal/02-05-08.pdf >. Acesso em 19 maio de 2002.

[ROS91] ROSE, M.. A Convention for Definig Traps for us with the SNMP. Request for Comments: 1215. Internet Engineering Task Force, março de 1991.

[RUG00] RUGGABER, Rainer; SEITZ, Jochen. Using CORBA applications in nomadic environments. 3rd IEEE Workshop on Mobile Computing Systems and Applications,2000 Monterey, CA **Proceedings…** [s. n.] : IEEE Press, 2000 . Disponível em <http://computer.org/Proceedings/wmcsa/0816/08160161abs.htm> .

[RUG99] RUGGABER, Rainer; SCHILLER, Jochen; SEITZ, Jochen. Using WAP as the Enabling Technology for CORBA in Mobile and Wireless Environments. Seventh IEEE Workshop on Future Trends of Distributed Computing Systems,1999, Cape Town, South Africa. . **Proceedings…** [s. n.] : IEEE Press,1999 Disponível em <http://computer.org/Proceedings/ftdcs/0468/04680069abs.htm >.

[SER00] SERHOUCHNI, Ahmed; CHERKAOUI, Omar; SAINT HILLAIRE, Ylian; MILI, Hafedh; OBAID, Abdelatif. The modularity of SNMPv3. IEEE Symposium on Computers and Communications, 3, 1998, Atenas. **Proceedings…**[s.l]: IEEE Press. 5p., p. 120 –124.

[SIL00] SILVA, Ricardo Pereira e, Suporte ao desenvolvimento e uso de *Frameworks* e componentes.2000 Dissertação de Doutorado em Ciências da Computação, Universidade Federal do Rio Grande do Sul, Porto Alegre.

[SMS00] SMS Send. SMS Send Home Page. Disponível em: < http://zekiller.skytech.org/smssend_en.php > Acesso em 20 abril 2002.

[STE01] STERRITT, R. Discovering rules for fault management. IEEE/ACM International Symposium on Cluster Computing and the Grid, 2001 **Proceedings…**[s.l]: IEEE Press, 2001. 10 p., p. 404 – 413.

[STU02] STUCKMANN, Peter; HOYMANN, Christian. Performance evaluation of WAP-based applications over GPRS. IEEE International Conference on Communications,2002, New York. **Proceedings…** [s.l.]: IEEE Press, 2002, p. 5, p. 3356 –3360.

[THA00] THANH, Do Van. Security issues in Móbile eCommerce. International Workshop on Database and Expert Systems Applications, 11, 2000, Greenwich. **Proceedings…** [s. n.] : IEEE Press, 2000 14p. Disponível em: http://computer.org/proceedings/dexa/0680/06800412abs.htm.

[THO98] THOMPSON, J.Patrick Web-based enterprise management architecture . **IEEE Communications Magazine.** [s.l.], n.3, p. 80-86, março 1998.

[VIS02] VISIBROKER Disponível em: http://info.borland.com/techpubs/visibroker/ acessado em 30 de julho de 2002.

[VOU01] VOUGIOUKAS, Stavros; ROUMELIOTIS, Manos. A system for basic-level network fault management based on the GSM short message service (SMS). International Conference on Trends in Communications, 2001, Bratislava. **Proceedings…**[s.l.]:IEEE Press, 2001. 5 p., p. 218 –222, v. 1.

[WAN01] WANG, Szu-Chi; SU, Wei-Cheng; KUO, Sy-Yen. Failure detection mechanism for distributed object computing using CORBA. Pacific Rim International Symposium on Dependable Computing, 2001, Seoul. **Proceedings…** [s.l.]: IEEE Press, 2001, 8 p., p 273 - 280 .

[WAP01] WAP Forum. Wireles Application Protocol Architeture Specification. Disponível em
< http://www.wapforum.org> Acesso em 20 setembro 2001.

[WAP01a] WAP Forum. Wireless Application Protocol WAP 161 WMLScriptCrypto Disponível em
< http://www.wapforum.org> Acesso em 20 setembro 2001.

[WAP01b] WAP Forum. Wireless Transport Layer Security. Disponível em <http://www.wapforum.org> Acesso em 20 setembro 2001.

[WHA01] WHATSUP, IPSwitch. User´s Guide. Disponível em <http://www.ipswitch.com/support/whatsup/guide/v700/index.html > . Acesso em 10 maio de 2002.

[WUN02] WUNNAVA, Subbarao.V.; HAMBISSA, Yonas. Information management using the centralized and distributed schemes. SoutheastCon, 2002 **Proceedings…**[s.l]: IEEE Press, 2002, 5 p., p. 10 – 14.