ULBRA CENTRO UNIVERSITÁRIO LUTERANO DE PALMAS

COMUNIDADE EVANGÉLICA LUTERANA "SÃO PAULO"
Recredenciado pela Portaria Ministerial nº 3.607 - D.O.U. nº 202 de 20/10/2005

JOSE NETO LUZ CARNEIRO

IMPLEMENTAÇÃO DE UMA SOLUÇÃO DE APOIO À AUDITORIA EM REDES WINDOWS

Palmas
2008

ULBRA CENTRO UNIVERSITÁRIO LUTERANO DE PALMAS

COMUNIDADE EVANGÉLICA LUTERANA "SÃO PAULO"
Recredenciado pela Portaria Ministerial nº 3.607 - D.O.U. nº 202 de 20/10/2005

JOSÉ NETO LUZ CARNEIRO

IMPLEMENTAÇÃO DE UMA SOLUÇÃO DE APOIO À AUDITORIA EM REDES WINDOWS

"Monografia apresentada como requisito parcial da disciplina Trabalho de Conclusão de Curso em Sistemas de Informação I (TCC I) do Curso de Sistemas de Informação, sob a orientação do Prof. Fernando Luiz de Oliveira."

Palmas

2008

JOSÉ NETO LUZ CARNEIRO

IMPLEMENTAÇÃO DE UMA SOLUÇÃO DE APOIO À AUDITORIA EM REDES WINDOWS

"Monografia apresentada como requisito parcial da disciplina Trabalho de Conclusão de Curso em Sistemas de Informação I (TCC I) do Curso de Sistemas de Informação, sob a orientação do Prof. Fernando Luiz de Oliveira."

Aprovada em julho de 2008

BANCA EXAMINADORA

Prof. M.Sc. Fernando Luiz Oliveira
Centro Universitário Luterano de Palmas

Prof. M.Sc. Ricardo Costa Soares de Jesus
Centro Universitário Luterano de Palmas

Profª. M.Sc. Madianita Bogo Marioti
Centro Universitário Luterano de Palmas

Palmas
2008

AGRADECIMENTOS

As primícias das gratidões dedico a Deus que me deu forças, clareza de entendimento e perseverança para vencer a metade desta última etapa. O meu Deus é um ser vivo que sempre esteve presente em todos os momentos da minha vida, e, neste contexto, não se ausentou de forma alguma na realização deste trabalho.

Sou muito grato à minha linda família, minha esposa Eliete e os meus pimpolhos Victor, Heloisa e Samuel, os quais estiveram lado a lado sempre me apoiando. Apesar de sempre perguntarem quando papai terminará a Faculdade para sobrar mais tempo para nós, souberam entender as minhas constantes ausências para dedicar aos estudos. Vocês, família amiga, tens sido a razão de não desistir. Amo muito vocês.

Aos meus pais, irmãos e sogros meus sinceros agradecimentos. Vocês acompanharam de perto esta caminhada que está chegando próximo do fim e sempre estiveram orando por mim. Sou grato a Deus pela vida de vocês.

Agradeço também ao Professor Fernando pelo incentivo, paciência, dedicação autêntica nas minúcias corretivas e pelas sugestões de melhorias ao trabalho. A sua atuação como orientador, com certeza, dá segurança aos seus orientandos durante a realização dos trabalhos.

Ao professor Ricardo Max também agradeço pelos incentivos iniciais e pela brilhante idéia de sugerir o assunto a ser trabalhado, o qual muito enriqueceu meus conhecimentos profissionais.

Enfim, também agradeço a todos que me ajudaram de forma direta ou indireta na realização deste trabalho. O meu muito obrigado a todos.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

OSI	Open Systems Interconnection
ISO	Internacional Organization for Standardization
MIB	Management Information Base
CMIP	Common Management Information Protocol
SNMP	Simple Network Management Protocol
SMI	Structure of Management information
GDMO	Guidelines for the Definition of Managed Objects
SGMP	Simple Gateway Management Protocol
RMON	Remote Network Monitoring
ICMP	Internet Control Message Protocol
SMAE	System Management Application Entity
SMASE	System Management Application Service Element
ACSE	Association Control Service Element
ASE	Application Service Elements
PDU	Protocol Data Units
CMIS	Common Management Information Service
CMISE	Common Management Information Service Element
WMI	Windows Management Instrumentation
WBEM	Web-Based Enterprise Management
CIM	Modelo de Informações Comuns
CIMON	Gerente de objetos do Modelo de Informações Comuns
API	Application Programming Interface

RESUMO

A crescente expansão das redes de computadores e a heterogeneidade de padrões e equipamentos tornaram complexo o controle o gerenciamento das redes. Em conseqüência dessa complexidade os organismos internacionais resolveram definir padrões de gerenciamento a serem seguidos pelos fabricantes de hardware e softwares de redes, resultando nos modelos de gerenciamento de rede OSI e no modelo Internet (TCP/IP). Dentre as áreas de gerenciamento definidos se encontra o gerenciamento de segurança, o qual incorpora o processo de auditoria da rede. Fazer auditoria do que acontece em uma rede em termos de acessos indevidos a equipamentos, arquivos, roubos de informações estratégicas das instituições, etc., é um fator também bastante complexo e de domínio de poucos que se dedicam nessa árdua tarefa. O presente trabalho objetiva minimizar essa complexidade para realização de auditoria em redes windows, trazendo um embasamento teórico como suporte aos administradores de redes que tenham interesse em criar soluções de gerenciamento de segurança através de auditoria, utilizando da tecnologia de gerenciamento do Windows, disponibilizada pela Microsoft, denominada de *Windows Management Instrumentation* (WMI).

PALAVRA CHAVE: Gerenciamento, rede, auditoria, OSI, Internet, script, WMI.

1. INTRODUÇÃO

No princípio, as redes de computadores desempenhavam, de forma elementar, as tarefas de integrar os recursos tecnológicos, compartilhar informações e equipamentos de uma empresa, tendo o foco principal reduzir custos e aumentar a produtividade. Porém, a expansão das redes agregou novos valores e a cada dia surgem novas tecnologias de redes para atender a crescente demanda dos usuários. Neste contexto, as redes se expandiram das simples conexões locais para redes de longo alcance (WAN), formando posteriormente a rede mundial de computadores denominada internet. Para concretizar essa realidade diversos fabricantes de equipamentos de redes, com tecnologias diversas e padrões de funcionamentos diferentes surgiram e foram inseridas no mercado, causando uma enorme complexidade para manter o controle e o gerenciamento dessas redes.

Na prática quando se implanta uma rede os investimentos são voltados para montar a estrutura física e lógica da rede, não integrando no primeiro momento os investimentos com o gerenciamento. O surgimento de problemas de conexões, baixo desempenho, roubos de informações, acessos indevidos e falta de controle de tráfego da rede deixam desesperados os administradores da rede e consequentemente faz emergir a necessidade de gerenciamento da rede de forma que integre todas as tecnologias adotadas. Surgiram, então, esforços isolados para encontrar soluções de gerenciamento, porém, sem sucesso porque não havia padrão a ser seguido pelos fabricantes de hardwares e softwares de rede de forma que possibilitasse integração na comunicação entre os diversos componentes da rede.

Diante dessa problemática foram definidos dois padrões de gerenciamento de redes, o modelo OSI (*Open Systems Interconnection*) e o modelo Internet (TCP/IP). Ambos os modelos apresentam similaridades nas funções básicas de interação entre os elementos da rede para fins de obter informações e fazer o gerenciamento. Os dois

modelos adotam os componentes de gerenciamento conceitualmente denominados de gerente, agentes, base de informações de gerenciamento e o protocolo de transporte das operações e informações de gerenciamento. Apesar de terem complexidades diferentes, estes dois modelos atingem as tarefas básicas de gerência de redes: obter informações da rede e tratá-las, fazer diagnóstico e apresentar soluções (EMBRATEL, 1994, p.514-515).

O gerenciamento de redes para ser eficiente engloba diversas áreas funcionais, tais como o gerenciamento de segurança, na qual está inserido o processo de auditoria de rede, o foco do presente trabalho. Através do processo de auditoria torna possível monitorar o acesso à rede, detectar intrusos, acesso às informações sigilosas e de acesso restritos, enfim, traçar o perfil da rede e dos usuários.

A auditoria se faz com base nas informações registradas em arquivos de *logs* gerados nos diversos objetos da rede. A análise desses *logs* além de ser uma tarefa tediosa e complexa, é impraticável quando se tem que acessar máquina a máquina para obtê-los. O trabalho proposto tentará minimizar esse problema, com a implementação de uma solução que objetiva automatizar a coleta dos *logs* de auditoria de uma rede e armazená-los em uma base de dados centralizada. Desta forma, pretende-se facilitar a obtenção das informações para auditoria através de consultas e relatórios gerenciais originadas em uma única base de dados. No entanto, o presente trabalho traz apenas a primeira parte para a solução proposta que é o embasamento teórico que norteia e deixa claro a possibilidade de concretizar a parte prática da implementação da solução.

2. REVISÃO DE LITERATURA

Esta seção objetiva apresentar o estudo teórico sobre os principais conceitos e tecnologias necessárias para o desenvolvimento do projeto proposto. Como o assunto auditoria está incluso no tema gerenciamento de redes e para melhor entendimento e ampliação do conhecimento sobre o trabalho serão abordados conceitos sobre gerenciamento de redes, modelos de gerenciamento, segurança de redes, infra-estrutura de redes windows bem como auditoria em rede windows. Por último será abordada a parte teórica da tecnologia *Windows Management Instrumentation* – WMI, a qual será fundamental para o objetivo final do projeto.

2.1. Gerenciamento de Redes

A existência de equipamentos oriundos de tecnologias e fornecedores diferentes em uma rede de computadores, assim como, a diversidades de protocolos de comunicação e padrões de gerenciamentos próprios desses equipamentos poderá deixar o processo de gerenciamento da rede bastante complexo.

Por outro lado, a infra-estrutura lógica de uma rede poderá coexistir com vários domínios gerenciais e organizacionais com políticas de gerenciamento e de segurança conforme a necessidade inerente a cada um. Esta situação dificulta ainda mais a adoção de políticas de segurança e padrões de gerenciamento uniformes para a rede, bem como unir conhecimentos técnicos capaz de dominar essa diversidade.

Portanto, para ser possível obter um bom nível de gerenciamento da rede torna-se necessária a adoção de recursos tecnológicos apropriados e recursos humanos bem capacitados e dedicados à gerência da rede.

A principal função do gerenciamento de redes é manter o ambiente em pleno funcionamento com bom desempenho, com as informações e recursos compartilhados

sempre disponíveis e com segurança no controle de acesso e uso dos dados que são transportados na rede. Para tanto, as tarefas básicas de gerência em redes precisam ser bem implementadas e executadas para que seja possível descobrir, prever e reagir a problemas. As tarefas básicas de gerência são (EMBRATEL, 1994, p.514-515):

- Obter informações da rede e tratá-las;
- Fazer diagnóstico com base nas informações tratadas; e
- Encaminhar soluções para os problemas.

Assim, fazer a gerência de pequenas redes, contendo poucos equipamentos, não é tão complicado, mas gerenciar uma rede composta de várias sub-redes que abrangem diversos departamentos e pavimentos, ou ainda uma rede remota (WANs/MANs) com uma cobertura geográfica extensa, a qual incorpora vários pontos e equipamentos de interconexão, é uma tarefa indispensável e bastante complexa. Por isto é exigido uma dedicação e conhecimento técnico, além de boas ferramentas de gerenciamento de redes para que a mesma possa ter uma gerência aceitável e controlável. Para tanto, para melhor aproveitar os recursos e vantagens que as redes propiciam é necessário investir em ferramentas de gerenciamento para facilitar a coordenação, controle e monitoração do comportamento dos elementos da rede de forma integrada (EMBRATEL, 1994, p.508).

Diante dessas dificuldades e para resolver principalmente os problemas quanto à heterogeneidade de soluções de redes e interoperabilidade entre domínios diferentes, a qual é fundamental para prover sistemas que suportem as cinco áreas funcionais de gerenciamento: gerenciamento de falhas, de configuração, de desempenho, de segurança e de contabilização; os institutos de normalização e fabricantes de equipamentos de rede de computadores têm se juntado e definido padrões de gerenciamento de redes (EMBRATEL, 1994, p.508). Os padrões mais conhecidos para gerência de redes são: o modelo OSI (*Open Systems Interconnection*) de gerenciamento de redes, definido pela ISO (*International Organization for Standardization*) e o modelo de Gerenciamento Internet (TCP/IP).

2.1.1. Componentes de gerenciamento

Tanto o modelo OSI como o modelo Internet apresentam semelhanças quanto ao gerenciamento, os quais incluem os conceitos de agentes, gerente da rede, uma base de informações de gerenciamento (MIB - *Management Information Base*) e um protocolo de

aplicação com a finalidade de transportar operações e informações de gerenciamento entre os agentes e gerentes. A figura 1, apresentada a seguir, demonstra esta relação.

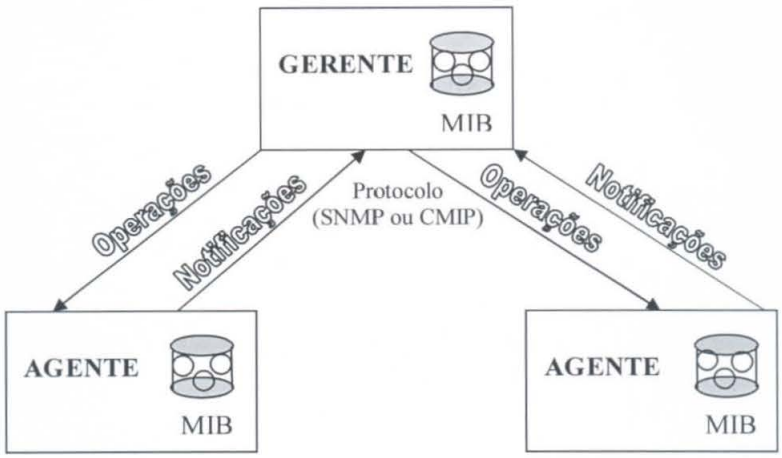

Figura 1: Relação Gerente-Agente, modificada de (FERNANDEZ, 2007, On-Line).

Conforme demonstra a figura 1, o gerente possui sua base de informações gerenciais que contempla dados de todos os objetos gerenciados da rede e possui a função de enviar solicitações e operações a serem executadas pelos agentes. Os agentes, além de atender as solicitações e executar as operações enviadas pelos gerentes, possuem a função de enviar notificações sobre ocorrências na rede ou nos objetos, assim como possuem uma base de informações sobre os respectivos objetos.

2.1.1.1. Gerentes e Agentes

O sistema gerente possui a responsabilidade de coordenar as atividades a serem executadas, através do envio de solicitações aos processos agentes. Os agentes possuem a responsabilidade de coletar e armazenar informações dos respectivos objetos gerenciados e notificar o sistema gerente sobre qualquer alteração ocorrida no estado dos objetos, além de ser responsável pela execução das operações sobre os objetos gerenciados e responder às solicitações dos gerentes. A comunicação entre gerente e agentes se dá conforme figura 2. A disponibilização das informações para os sistemas gerentes ocorre de duas formas (FREITAS, 2001, p. 6):

- *Request-response* ou *Polling*: quando o sistema gerente solicita a um agente o envio de alguma informação armazenada na MIB agente, caracterizando uma interação do tipo pergunta-resposta entre o gerente e agente.
- *Event-reporting* ou relato de ventos: quanto o agente é o autor da ação de enviar informações ou notificações ao gerente, se posicionando o gerente na condição de ouvinte. Esses eventos ocorrem quando o agente envia informações sobre o seu estado atual ou quando envia notificações a respeito de uma ocorrência relevante, por exemplo, quando acontece algo anormal na rede.

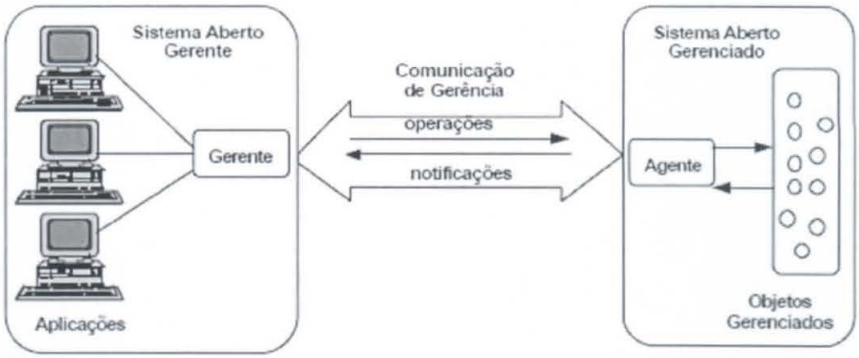

Figura 2: Comunicação entre gerente e agente (RAIMIR, 1998, p. 21).

O modelo OSI de gerenciamento de redes utiliza o protocolo CMIP (*Common Management Information Protocol*) para definir as regras de comunicação entre os sistemas gerente e agente, enquanto o modelo de Gerenciamento Internet utiliza o protocolo SNMP (*Simple Network Management Protocol*).

2.1.1.2. Base de Informações de Gerenciamento - MIB

A base de informações de gerenciamento é o repositório dos objetos gerenciados e pode-se classificá-la como um componente fundamental em um sistema de gerenciamento. Isto porque quando há uma solicitação do sistema gerente sobre um objeto gerenciado, o agente com base nas informações armazenadas em sua MIB, responde à solicitação do gerente.

Não faz parte do conceito da MIB nenhuma definição de estrutura interna da base de informação, nem a forma como será armazenada. O que tem que ser visualizado na

definição de uma MIB é o local onde estarão os objetos gerenciados, e a possibilidade dos sistemas identificarem de forma correta os respectivos objetos. Para tanto, devem ser seguidas as regras de definição e identificação das variáveis na MIB, definidas pela Estrutura de Informação de Gerenciamento – SMI (*Structure of Management Information).* Esta estrutura determina que os nomes e tipos de variáveis da MIB devem ser definidos e referenciados de acordo com a linguagem formal padronizada pela ISO, a saber: a ASN.1 (*Abstract Syntax Notation one).* O padrão ASN.1 define regras tanto para a linguagem humana quanto para a linguagem codificada, utilizada nos protocolos de comunicação (RAIMIR, 1998, p.23).

A MIB é usada tanto no agente quanto no gerente para armazenar e trocar informações de gerenciamento, conforme figura 1. A MIB associada ao agente é denominada MIB agente, e a associada ao gerente é denominada MIB gerente. Uma MIB gerente consiste de informações de todos os componentes que ela gerencia, enquanto a MIB agente necessita conhecer somente sua informação local.

Conforme definição da SMI (*Structure Management Information*) a MIB, em termos gerais, é caracterizada tanto pela organização dos itens e pela forma de identificá-los, como também pelos tipos de operações realizadas sobre os mesmos. Portanto, a diferença existentes entre as MIBs (MIBs da ISO e da Internet) se concentram exatamente nas hierarquias utilizadas para representar os objetos que serão gerenciados. No modelo OSI são definidas as hierarquias de herança, nomeação e de registro. No entanto, em ambos os modelos de gerenciamentos referenciados a hierarquia de registros é utilizada para identificar de forma genérica os objetos, cuja hierarquia foi especificada de acordo com as normas definidas pelo padrão ASN.1 usada na atribuição de identificadores de objetos (BRISA, 1993, p. 144).

2.1.1.2.1. Hierarquias de Gerenciamento

A SMI (*Structere of Management Information*), com base na abordagem de orientação a objetos, introduz os conceitos de hierarquia de herança, hierarquia de nomeação e de registros usados na caracterização e identificação dos objetos gerenciados. Além disso, é responsável pela definição do conjunto de operações que pode ser realizado sobre os objetos da MIB e o comportamento dos mesmos em relação à execução das operações (BRISA, 1993, p. 139).

Convém salientar que os três tipos de hierarquias são definidos para o modelo OSI, visto que no modelo Internet não é definida nenhuma hierarquia de nomeação para identificar instâncias de objeto, nem hierarquia de herança (classes). Neste modelo é definido somente o conceito de instância de objeto desvinculado do conceito de atributos, como também somente tipos de objetos e não classes de objetos (BRISA, 1993, p. 139). As hierarquias serão melhores descritas nos itens abaixo:

- **Hierarquia de Herança:** a hierarquia de herança é responsável por definir o conceito de classes de objetos e está relacionada às propriedades associadas aos objetos, as quais são descritas através de seus atributos, comportamento, pacotes condicionais, operações e notificações (BRISA, 1993, p. 139). Uma classe de objetos hierarquizados contém objetos que possuem propriedades similares.

  Dentro do conceito de classes são definidas a superclasse e subclasses, sendo que estas herdam, de forma irrestrita, todas as propriedades da superclasse conhecida também como classe pai, conforme exemplifica a figura 3. No entanto, as peculiaridades exclusivas das subclasses são inseridas como propriedades adicionais.

  Todos os objetos gerenciados pertencentes a uma classe devem possuir o mesmo comportamento, o qual pode ser definido pelos atributos. O comportamento dos objetos reflete a semântica dos atributos, as operações e notificações, como também a resposta às operações de gerenciamento, as circunstâncias em que as notificações devem ser emitidas, as dependências entre valores dos atributos particulares e os efeitos dos relacionamentos entre objetos gerenciados (BRISA, 1993, p. 141).

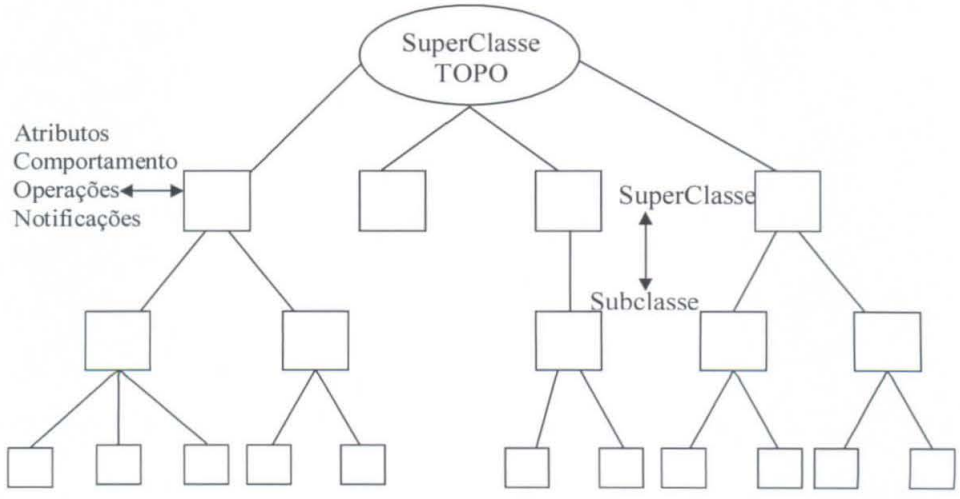

Figura 3: Hierarquia de classes de objetos gerenciados (BRISA, 1993, p.141).

- **Hierarquia de Nomeação:** esta hierarquia descreve os relacionamentos entre as instâncias de objetos com seus respectivos nomes. Esses relacionamentos aplicados aos objetos são do tipo "estar contido em". Assim o objeto que contém outros objetos, podendo ser da mesma classe ou de outras, é definido como superior e o que está contido como subordinado. Contudo, um objeto gerenciado pode estar contido somente dentro de um objeto gerenciado superior.

  Este relacionamento descrito pela hierarquia de nomeação, também conhecido como *containment*, pode ser usado para modelar hierarquias de partes do mundo real ou hierarquias organizacionais, como: diretórios, arquivos, registros e campos. Deste modo para cada classe de objeto são definidas uma ou mais regras para identificação da classe superior e do atributo característico, cujas regras são nomeadas de "*name bindings*". Um *name binding* é definido para uma classe de objetos, sendo que o mesmo fica disponível para ser utilizado em todas as classes derivadas da original (BRISA, 1993, p. 143).
- **Hierarquia de Registro:** esta hierarquia é utilizada para identificar de maneira universal os objetos (tanto no modelo OSI como no modelo Internet) e é especificada conforme as regras estabelecida pela notação ASN.1 (*Abstract Syntax Notation.One)* para árvore de registros usada na atribuição de identificadores a objetos, sendo que cada nó desta árvore está associada a uma autoridade de registro que determina como são atribuídos os seus números. A figura 4 exemplifica o modelo de hierarquia de registro do modelo OSI e a figura 5 o modelo Internet (BRISA, 1993, p. 144).

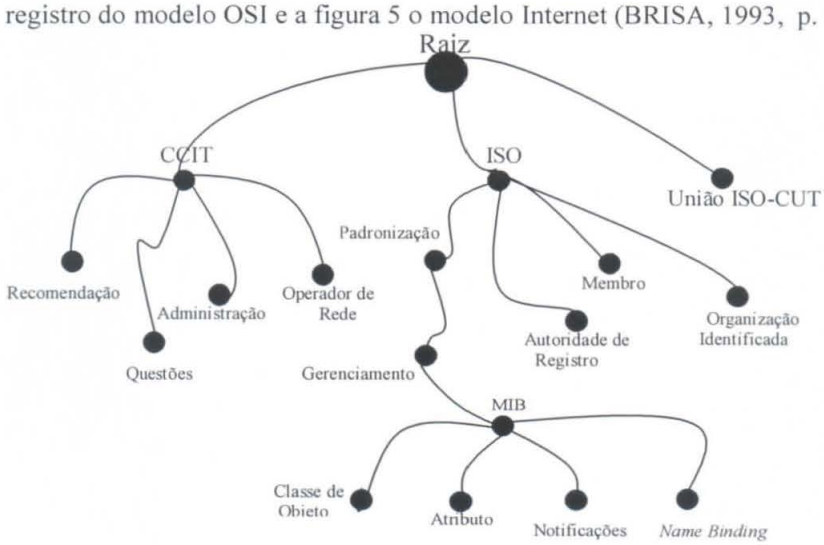

Figura 4: Ex. de Hierarquia de Registro–Modelo OSI (BRISA, 1993, p. 145).

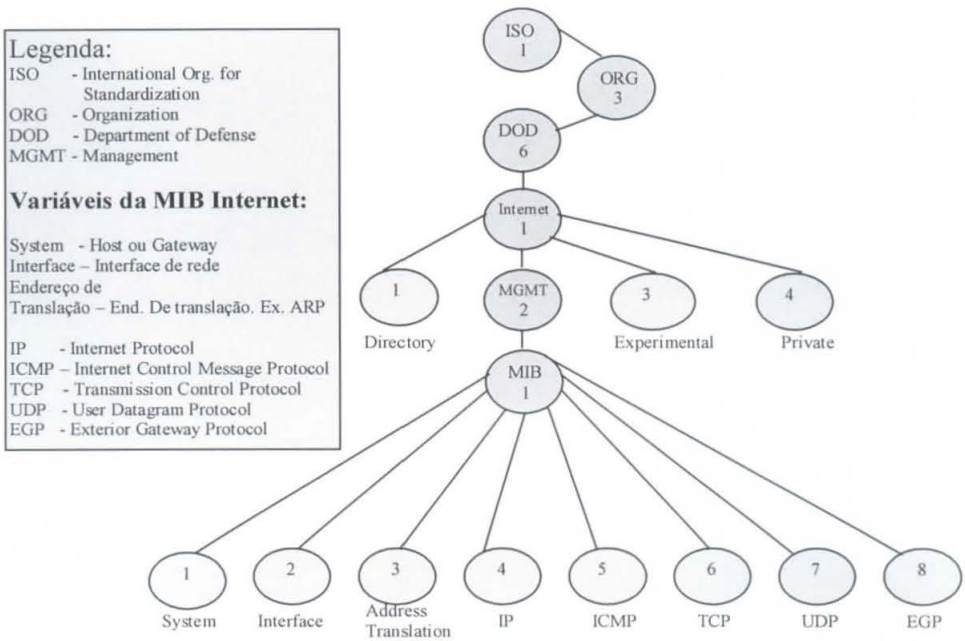

Figura 5: Ex. de Hierarquia de Registro – Modelo Internet (BRISA, 1993, p.164)

Analisando as figuras 4 e 5 convém destacar a hierarquia de registro de cada modelo quanto ao gerenciamento, especificadamente a diferença da base de informações de gerenciamento do modelo OSI em relação ao Internet, enfatizando que na figura 4 é notório a aplicação do conceito de orientação a objetos definido pelo modelo OSI. Outro fator importante é que a hierarquia de registros do modelo internet inicia-se no nó da autoridade de registro ISO, enquanto que o modelo OSI engloba nó de autoridades de mesmo nível da autoridade ISO, hierarquiamente subordinados a um nó raiz, apresentando maior estrutura quanto às autoridades de registros.

Como as MIBs, tanto do modelo OSI como do modelo Internet, são modeladas através do conceito de orientação a objetos, os recursos que serão gerenciados são denominados de "objetos gerenciados".

2.1.1.3. Objeto Gerenciado

Objeto gerenciado é uma representação de um recurso de rede, seja de ordem física ou lógica. Como o gerenciamento de redes trabalha com uma base de informações construída com base nos conceitos de orientação a objetos, estes constituem a base do gerenciamento

de redes. Nesta proporção, caso haja algum elemento de rede não gerenciado, o mesmo não é contemplado no sistema de gerenciamento. A definição de objeto gerenciado apresenta os seguintes aspectos (BRISA, 1993, p. 18):

- a localização do objeto dentro da estrutura de rede que está sendo gerenciada;
- os atributos do objeto, os quais representam a natureza do objeto;
- as operações em que são submetidos;
- as notificações que podem emitir ao gerente; e
- as relações com outros objetos.

Convém reforçar que para definição de objeto deve-se ater aos padrões definidos pela linguagem ASN.1, a qual descreve os princípios necessários para especificação de objetos – GDMO (*Guidelines for the Definition of Managed Objects*).

As informações estáticas e dinâmicas identificam a interação entre os sistemas de gerenciamento e os objetos gerenciados. As primeiras não causam alteração no conteúdo dos objetos, enquanto que as informações dinâmicas têm a função de manter a coerência das informações de gerenciamento entre os recursos reais e seus respectivos objetos gerenciados. Recursos reais pode ser software ou hardware, ou ambos, e podem gerar eventos que quando transmitidos podem causar modificação nos objetos gerenciados, por exemplo, atualização dos atributos dos objetos. A coerência é mantida pela atualização do objeto através da ação enviada antes dessa ação ser aplicada sobre o recurso real (RAIMIR, 1998, p.19).

2.2. Modelos de gerenciamento de redes

Como mencionado anteriormente, existem dois modelos a serem seguidos para implantação de gerenciamento de redes: o modelo de referência OSI e o modelo de gerenciamento TCP/IP. Como também já visto, esses dois modelos são similares quanto aos tipos de componentes adotados na arquitetura de gerenciamento, conhecidos por gerente, agente, objetos gerenciados e uma base de informações de gerenciamento – MIB. Cada modelo também possui um protocolo de aplicação exclusivo, o qual é responsável pelo transporte de operações e informações de gerenciamento entre os agentes e o gerente. Apesar da similaridade quanto aos tipos de componentes, existem diferenças substanciais quanto à forma e abrangência de comunicação entre os mesmos, inclusive quanto aos protocolos, visto que o protocolo CMIP é bem mais completo e complexo do que o

protocolo SNMP. Popularmente esses dois modelos são conhecidos como "Modelo de Gerenciamento OSI" e "Modelo de Gerenciamento Internet".

2.2.1. Diferenças entre os Modelos OSI e Internet

O modelo de Gerenciamento OSI usa como base de transporte o protocolo CMIP (*Common Management Information Service Over TCP)*, enquanto o modelo de Gerenciamento Internet utiliza o protocolo SNMP (*Simple Network Management Protocol)*. O protocolo SNMP não é orientado a conexão, o qual utiliza dos serviços prestados pelo UDP (*User Datagram Protocol*). Enquanto o protocolo CMIP é orientado a conexão e é executado sobre a pilha de protocolos OSI de gerenciamento, o que o torna mais confiável do que o modelo Internet (EMBRATEL, 1994, p. 548).

O protocolo CMIP apresenta uma melhor qualidade das informações, maior nível de confiabilidade e de segurança, porém trouxe conseqüências que impossibilitaram a sua adoção em maior escala, tais como:

- requer mais memória e processamento devido número de recursos de sistemas;
- o grande número de variáveis dificulta a compreensão sobre todas elas por parte do programador, o que a torna mais complicada a programação.

Já o protocolo SNMP foi o primeiro protocolo de gerenciamento público, como o próprio nome sugere é simples e de fácil implementação e possibilita o gerenciamento de ambientes diversos. Apesar das deficiências das primeiras versões, a simplicidade fez com que fosse adotado em um maior número de equipamentos e sistemas de gerenciamento de redes, tornando na prática um padrão de uso abrangente em gerência de redes.

No Gerenciamento Internet, o elemento agente possui a função básica de responder às operações emitidas pelo gerente, o envio de notificação ocorre somente em algumas situações de erros. Enquanto que no Gerenciamento OSI o elemento agente tanto responde às solicitações do gerente como emite notificações diversas de gerenciamento, cujas operações tendem a ser bem mais complexas do que as executadas pelo modelo Internet.

Apesar das duas arquiteturas adotarem os conceitos de orientação a objetos para descrever e especificar as informações das MIBs, no modelo Internet apenas são definidos os tipos de objetos a serem armazenados, enquanto o modelo OSI vai mais além ao especificar algumas classes de objetos a serem utilizadas nos sistemas de gerenciamento. O modelo OSI especifica um conjunto mais completo de operações de gerenciamento,

além de ser possível a sincronização das operações realizadas sobre vários objetos (EMBRATEL, 1994, p. 547).

2.2.2. Modelo Gerenciamento Internet

O ambiente de Gerenciamento Internet tem sido mais utilizado do que o modelo OSI e de certa forma dominado o mercado de sistema de gerenciamento de redes. O motivo de sua expansão tem sido sua simplicidade na implementação e por consumir poucos recursos de rede e de processamento, o que tem facilitado a sua inclusão desde os mais simples equipamentos até nos mais sofisticados.

O gerenciamento Internet baseia-se no protocolo SNMP (*Simple Network Management Protocol*), que teve origem no protocolo para monitoração de *gateways* IP, o *Simple Gateway Management Protocol* – SGMP. Nessa evolução, a partir do original SGMP para o SNMP, foram inclusas as definições de uma Estrutura de Informação e Gerenciamento – SMI (*Structere of Management Information*) e de uma Base de Informação de Gerenciamento – MIB (*Management Information Base*) (EMBRATEL, 1994, p. 532).

Apesar da abrangência na utilização do Gerenciamento Internet ser atribuída à sua leveza e facilidade de implementação, estas características tornaram-se insuficientes à medida que as redes tomaram maiores dimensões e complexidades, principalmente quanto à carência de funcionalidades, eficiência e segurança. A segunda versão do protocolo SNMP, denominado SNMPv2, surgida em 1993, além de suportar as operações *getbulkrequest* e *informrequest* para transferência de grandes blocos de informação e possibilitando o gerenciamento distribuído, teve o objetivo de amenizar as deficiências da primeira versão do protocolo, no entanto, o quesito segurança somente foi implementado na versão 3 do protocolo SNMP, SNMPv3 (RFC 2571). Esta última versão não aborda uma especificação completa de um protocolo de gerenciamento de redes, mas apenas traz uma documentação com definições das características de segurança, até então carente, e um *framework* para ser utilizado com as funcionalidades já existentes nas versões anteriores (FREITAS, 2001, p. 34). Outro ponto é que foram definidas as versões 1 e 2 do RMON (*Remote Network Monitoring*) com o objetivo de diminuir o tráfego no gerenciamento e monitoração no cenário onde o sistema de gerenciamento e as redes gerenciadas são interconectadas através de redes de longa distância.

O modelo de Gerenciamento Internet apresenta um esquema centralizado, sendo que o gerente é configurado em uma estação de trabalho e os agentes ou *proxy agents* consistem nos outros elementos da rede. O *proxy agent* faz o papel de um procurador para os equipamentos que não implementam o protocolo SNMP. Os componentes de gerenciamento do modelo Internet são os mesmos já referenciados: Gerente e agentes, objetos gerenciados associados a uma MIB e o protocolo de gerenciamento, conforme figura 6 (EMBRATEL, 1994, p. 534).

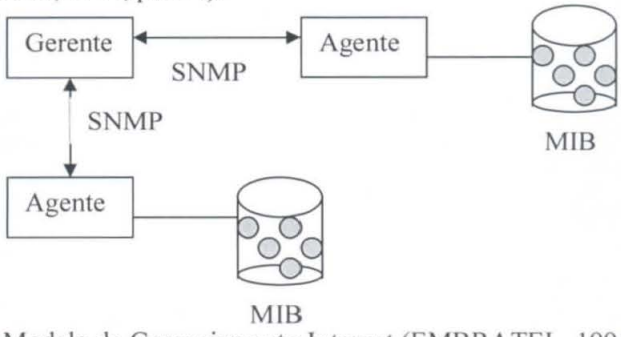

Figura 6: Modelo de Gerenciamento Internet (EMBRATEL, 1994, p. 535).

Analisando a figura 6 nota-se a relação entre os componentes que compõem o modelo de gerenciamento internet, destacando o protocolo SNMP que faz a comunicação entre o componente gerente e os agentes.

No modelo Internet não é definido o conceito de classes e atributos de objetos a serem empregados pelos sistemas de gerenciamentos, conforme o modelo OSI, apenas são definidos tipos de objetos a serem armazenados na MIB (EMBRATEL, 1994, p.536), cuja definição contém cinco campos:

- nome textual com o respectivo identificador;
- definição da semântica associada ao tipo de objeto;
- a sintaxe ASN.1;
- o tipo de acesso (*Read-only*, *read-write*, *write-only* ou não acessível); e
- o *status* (obrigatório, opcional ou obsoleto).

Os tipos de objetos definidos são:

- simples: tipos primitivos da ASN.1: *Integer, Octet String, Null* e *Object Identifier*;
- contexto de Aplicação: utilizando tipos especiais definidos pela SMI: *IpAddress, NetworkAddres, Counter, Gauge, TimeTicks* e *Opaque*; e
- construídos: utilizando para definição tipos construtores da ASN.1: *<list>* e *<table>*.

A funcionalidade do sistema de Gerenciamento Internet é alcançada através das seguintes operações (EMBRATEL, 1994, p.543-544):

- *get-Request*: usada para recuperar uma informação de gerenciamento;
- *get-Next-Request*: utilizada quando não está definido ou não tem conhecimento das variáveis a serem recuperadas;
- *get-Response*: corresponde a operação *Get-Request, Get-Next-Request* ou *Set-Request;*
- *set-Request*: utilizada para atribuição de valores às variáveis que contêm informações de gerenciamento;
- *trap*: utilizada para relatar ocorrências de eventos extraordinários;
- *get-Bulk-Request*: corrige uma deficiência do SNMPv1 na recuperação de grandes blocos de informação de gerenciamento;
- *inform-Request*: possibilita um gerente enviar informações de para outro gerente.

As mensagens do SNMP são constituídas de um cabeçalho de autenticação seguida de uma das seguintes unidades de dados de protocolo – PDU (*Protocol Data Unit*): *get-request, get-next-request, set-request, get-response* e *trap*. O mecanismo de *request-response* é implementado utilizando as quatros primeiras PDUs. O monitoramento da rede é feito utilizando as PDUs *get-request e get-next-request*, enquanto que para o controle da rede utiliza-se a PDU *set-request*, a qual permite modificar, criar e remover novas instâncias de informações de gerenciamento. A PDU *get-response* é transmitida pelo agente em resposta às PDUs *get-request, get-next-request, set-request* (EMBRATEL, 1994, p.542).

Como o protocolo SNMP não é orientado à conexão, em caso de redes locais o mesmo utiliza a pilha de protocolos abaixo, a qual está também exemplificada na figura 7:

PDU do SNMP
Cabeçalho do SNMP
Camada de Transporte: UDP
Camada de Rede: IP
Camada de Enlace: LLC e MAC
Camada Física

Figura 7: Pilha de protocolos utilizada pelo SNMP (EMBRATEL, 1994, p.545).

- Camada de enlace: Protocolos LLC (*Logical Link Control*) e MAC (*Medium access Control*);
- Camada de Rede: Protocolos IP e ICMP (*Internet Control Message Protocol*):
- Camada de transporte: Protocolo UDP

2.2.3. Modelo OSI de Gerenciamento de Rede

Em 1989, a ISO (*International Organization for Standardization*) apresentou a Arquitetura de Gerenciamento OSI, com o objetivo de cobrir a necessidade de uma arquitetura que fosse capaz de gerenciar os mais diversos elementos gerenciáveis existentes em uma rede, mantendo as características de integração, simplicidade, segurança e flexibilidade. A arquitetura de Gerenciamento OSI define os seguintes itens (EMBRATEL, 1994, p. 508-509):

- estrutura de gerenciamento;
- os componentes de gerenciamento;
- estrutura da informação de gerenciamento;
- os serviços e o protocolo para a troca de informações de gerenciamento.

A estrutura do modelo de Gerenciamento OSI se subdivide da seguinte forma, (EMBRATEL, 1994, p. 509):

- gerenciamento de Sistemas: exige funções de apoio em todas as sete camadas da arquitetura de redes OSI.
- gerenciamento de Camada: utiliza protocolos de gerenciamento de propósito especial e restringe-se ao gerenciamento de recursos relacionados com as atividades de comunicação de uma única camada.
- operação de Camada: é uma forma de gerenciamento restrita a uma única instância de comunicação em uma camada.

Os componentes de gerenciamento do modelo OSI são os já descritos anteriormente, os quais incluem: os conceitos de gerente e agente, objetos gerenciados (entidades lógicas), a base de informação de gerenciamento – MIB e o protocolo de transporte das informações de gerenciamento. Logo, para desenvolver aplicações de gerenciamento é necessário utilizar os processos distribuídos formado pelos gerentes, responsáveis pela gerência dos objetos da rede, e pelos agentes, responsáveis por realizar as operações e pelo envio de notificações aos gerentes. A MIB é constituída pelo conjunto

de objetos gerenciados e respectivos atributos, operações que podem ser enviadas pelo gerente para ser executada pelos agentes e as notificações que os agentes podem enviar ao sistema gerente.

Ao se tratar de gerenciamento de sistemas, duas ou mais entidades de aplicação podem associar-se para prover uma instância de aplicação de gerenciamento. A entidade de aplicação de gerenciamento de sistemas é denominada como SMAE (*System Management Application Entity*), a qual é composta pelo: SMASE (*System Management Application Service Element*), ACSE (*Association Control Service Element*) e por outros ASE (*Application Service Elements*), conforme mostra a figura 8 (EMBRATEL, 1994, p.510).

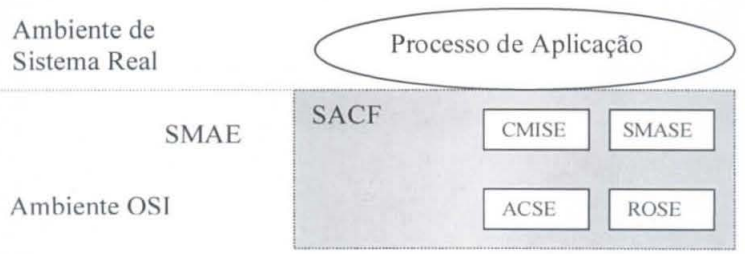

Figura 8: Entidade de aplicação de Gerenciamento de Sistemas (EMBRATEL, 1994, p.510).

O serviço geral de Gerenciamento OSI é denominado de CMIS (*Common Management Information Service*), sendo que a aplicação que utiliza este serviço é denominada de MIS-User (*Management Information Service - User*). As interações entre os MIS-Users são realizadas através de trocas de informações de gerenciamento.

O protocolo utilizado pelo modelo OSI para transporte de informações de gerenciamento é o CMIP (*Common Management Information Protocol)*, o qual implementa as primitivas oferecidas pelo CMIS. Para estabelecimento da comunicação este protocolo utiliza os elementos de serviço ACSE e ROSE (*Remote Operations Service Element*) (EMBRATEL, 1994, p. 511).

Através da MIB, o modelo de Gerenciamento OSI proporciona uma interface com cada uma das sete camadas de arquitetura de rede OSI, oferecendo as operações necessárias para executar o gerenciamento da rede em todas as camadas. Uma LME (*Layer Management Entity*) é uma entidade de gerenciamento de camada que concentra a

funcionalidade da camada sob sua responsabilidade. Sendo que a integração e a função de interface com o gerente são realizadas pela SMAE, conforme demonstra a figura 9, cuja figura apresenta ainda a comunicação entre duas entidades de aplicação de gerenciamento de sistemas.

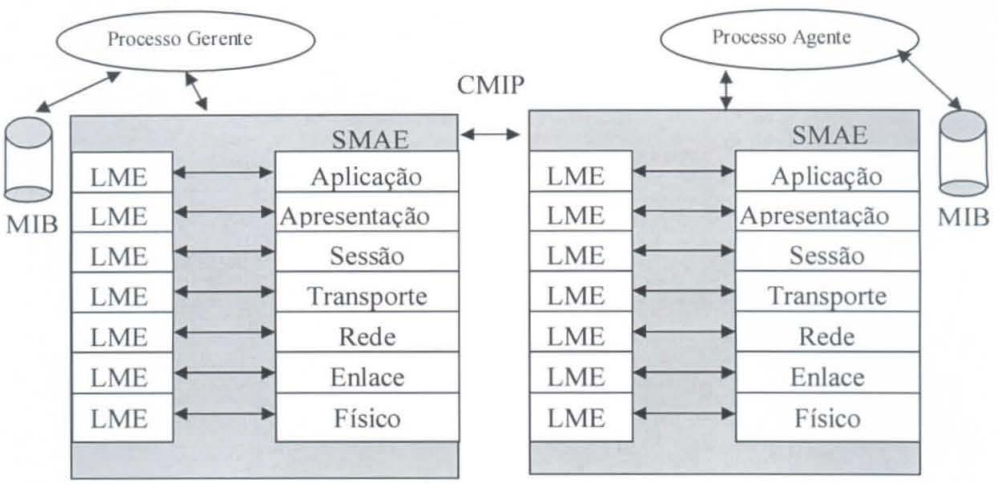

Figura 9: Comunicação entre entidades de aplicação de ger. de sistemas (EMBRATEL, 1994, p.511).

2.2.4. Serviços do Gerenciamento OSI

A camada de aplicação do modelo de Gerenciamento OSI fornece os serviços e o protocolo utilizado para implementação da aplicação de gerenciamento, dos quais incluem: gerenciamento de desempenho, do nível de falhas, de segurança, de configuração e de contabilidade. Na aplicação de gerenciamento de sistemas estão envolvidos os seguintes elementos de serviço de aplicação (BRISA, 1993, p. 169):

- o SMASE (*System Management Aplication Service Element*): este serviço define a sintaxe abstrata e semântica das informações transferidas para o gerenciamento OSI. Este elemento está relacionado com as áreas funcionais de gerenciamento.
- o CMISE (*Common Management Information Service Element*) implementa os serviços definidos pelo CMIS (Serviço geral de gerenciamento OSI) executando o protocolo CMIP. CMISE é o elemento de serviço que tem a função de prover meio para a troca de informações de gerenciamento comum. Cuja troca é realizada através das CMIPDU (*Common Management Information Protocol Data Units*), o qual

constitui as unidades de dados do Protocolo CMIP. O elemento CMISE é tido como comum por ser utilizado nas cinco áreas funcionais de gerenciamento.

O serviço proporcionado pelo CMISE às aplicações de gerenciamento para fins de troca de informações é definido conforme Tabela 1 (BRISA, 1993, p. 161):

Tabela 1: Primitivas de definição que define o serviço de troca de informações.

Primitivas	**Descrição**
M-CREATE-request	Solicita a criação de uma instância de um objeto gerenciado
M-CREATE-response/confirmação	Confirma a criação da instância do objeto gerenciado
M-DELETE-request/indications	Solicita remoção de uma instância do objeto
M-DELETE-response/confirmation	Confirma a remoção da instância
M-GET-request/indication	Solicita a leitura de valores de atributos
M-GET-response/confirmação	Retorna o valor do atributo solicitado
M-CANCEL-GET-request/indication	Solicita cancelamento do M-GET
M-CANCEL-GET-response/confirmation	Confirma o cancelamento de uma operação M-GET
M-SET-request/indication M-SET-response/confirmation	Solicita a atribuição de valor a um atributo Confirma a atribuição de valor ao objeto
M-EVENT-REPORT	Emite relatório de eventos.
M-EVENT-REPORT:	Confirma a recepção de algum tipo de relatório de redes
M-ACTION-request/indication M-ACTION-response/confirmation	Solicita execução de uma ação sobre objeto gerenciado Confirma a execução da ação sobre o objeto.

No CMISE existe a unidade funcional *Kernel* e um conjunto de unidades funcionais adicionais, como: de seleção de objetos múltiplos, de filtro, de respostas múltiplas, de serviço estendido e *Cancel Get*. Na unidade funcional *kernel* estão inclusos todos os serviços listados na tabela 1, exceto o *M-Cancel-Get* por estar incluído na unidade *Cancel Get*.

A figura 10 apresenta o detalhamento da camada de aplicação e o relacionamento entre os elementos de serviços de aplicação de gerenciamento de sistemas. Convém destacar que as operações e notificações de gerenciamento de sistemas estão dentro da SMI (Estrutura de Informação de Gerenciamento). Essas operações de gerenciamento são definidas em dois tipos (BRISA, 1993, p. 172):

- operações orientadas a atributos, as quais são aplicadas sobre os atributos, como: *Get*, *Replace*, *Set-With-Default*, *Add member* e *Remove member*.
- operações que se aplicam sobre os objetos gerenciados, como: *Create*, *Delete* e *Action*. A semântica destas operações faz parte da definição das classes dos objetos gerenciados.

A figura 10 apresenta o relacionamento entre os conceitos apresentados anteriormente, bem como deixa mais nítido o que de fato existe em uma MIB, ou seja, o conjunto de informações de gerenciamento pertencente a um sistema que é transferido através dos protocolos de gerenciamento.

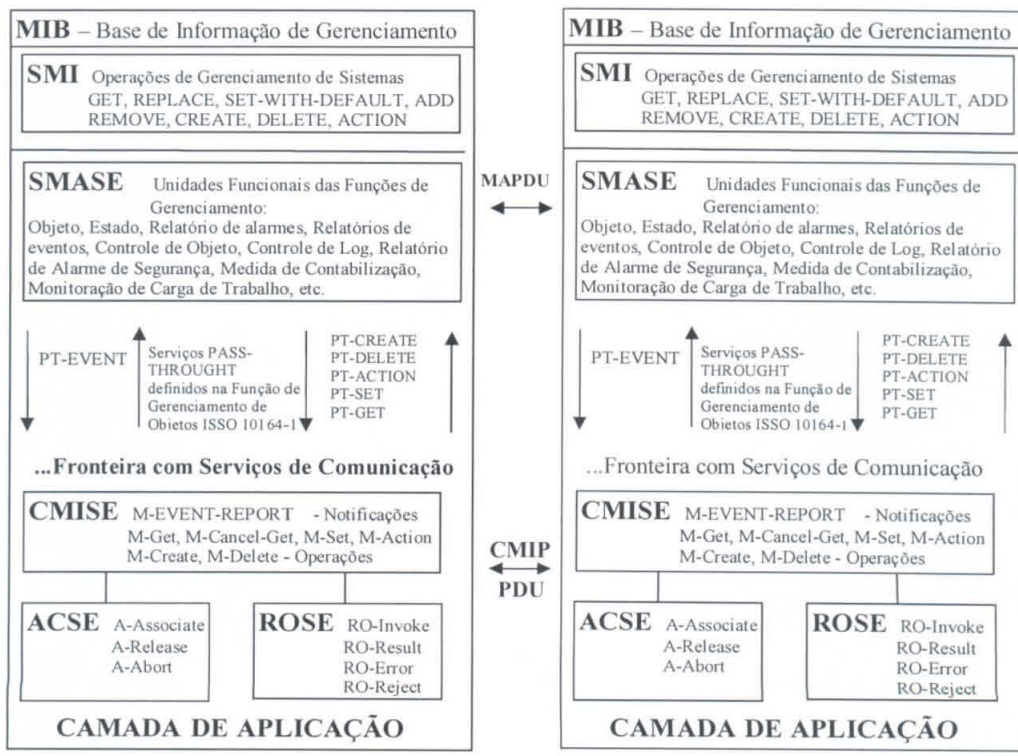

Figura 10: Detalhamento da Camada de Aplicação (BRISA, 1993, p. 173).

Explicitando as informações da figura 10 destaca-se que a SMI especifica a sintaxe da informação que é transferida para fins de gerenciamento; o SMASE traz as unidades funcionais das funções de gerenciamento identificadas pelas respectivas aplicações, as quais especificam as informações de gerenciamento a serem trocadas entre os processos de gerenciamento através das MAPDUS; e a interação entre SMASE e o CMISE comprovando que o serviço de comunicação utilizado pelo SMASE é o CMISE, o qual define o serviço e procedimentos para transferência das CMIPDUS (BRISA, 1993, p. 172-175).

2.2.5. Áreas Funcionais de Gerenciamento de Redes

Áreas funcionais de gerência são as diferentes partes que podem ocorrer num problema de gerenciamento de redes, como: configuração da rede, falhas de componentes, o nível de desempenho apresentado no uso da rede, o nível de segurança relativo aos acessos e às informações e a contabilização da utilização da rede.

Essas atividades de gerenciamento de redes foram classificadas pela ISO (*International Organization for Standardization*) em cinco áreas funcionais de aplicação (gerenciamento de configuração, de falhas, de desempenho, de contabilização e de segurança), conforme demonstra a figura 11, as quais foram adotadas na maioria dos sistemas de gerenciamento de redes.

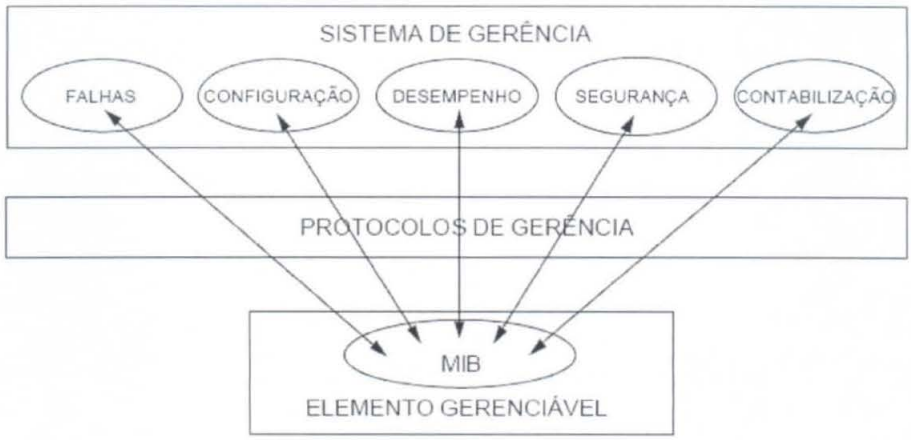

Figura 11: Áreas funcionais do gerenciamento de redes (RAIMIR, 1998, p. 24).

As informações relevantes para o funcionamento dessas áreas funcionais estão nas MIBs, as quais são constituídas pelos objetos gerenciados, seus atributos, as notificações passíveis de serem enviadas ao sistema gerente e as operações que podem ser executadas. O padrão de representação dessas informações de gerenciamento, as ferramentas de coletas das mesmas e o controle dos objetos gerenciados constituem a estrutura de dados que foi definida com a linguagem ASN.1 (BRISA, 1993, p. 178).

2.2.5.1. Funções de Gerenciamento

Como as informações geradas em uma área funcional podem ser úteis como suportes para decisões em outras áreas, a ISO definiu algumas funções de gerenciamento para garantir os requisitos intrínsecos às áreas funcionais. Por exemplo, funções de gerenciamento de objeto, estado, atributos para representação de relacionamentos, relatórios de alarmes, relatórios de eventos, controle de *logs* e controle de acesso, os quais serão melhores descritos a seguir (EMBRATEL, 1994, p. 515):

- Função de Gerenciamento de Objeto – OMF (*Object Management Function*), que tem o objetivo de gerenciar a criação e a remoção de um objeto gerenciado, as alterações nos respectivos atributos e gerar relatórios dessas ações. Essa função descreve a forma de utilização do serviço no mapeamento de uma operação de gerenciamento sobre o serviço correspondente do CMISE, conforme demonstrado na Tabela 2 (EMBRATEL, 1994, p. 516);

Tabela 2: Mapeamento das Operações de Gerenciamento

OPERAÇÃO DE GERENCIAMENTO	PASS-THROUGH	CMISE
CREATE	PT-CREATE	M-CREATE
DELETE	PT-DELETE	M-DELETE
ACTION	PT-ACTION	M-ACTION
REPLACE	PT-SET	M-SET
ADD MEMBER	PT-SET	M-SET
REMOVE MEMBER	PT-SET	M-SET
SET-WITH-DEFAULT	PT-SET	M-SET
GET	PT-GET	M-GET
NOTIFICATION	PT-EVENT	M-EVENT-REPORT

- Função de Gerenciamento de Estado – STMF (*State Management Function)*, responsável por representar as condições reais e instantâneas sobre a disponibilidade e operacionalização de um recurso. Cada classe de objetos gerenciados possui o próprio conjunto de atributos de estado. Essa função deve ser padronizada visto ser comum a uma grande quantidade de recursos gerenciados. Três fatores, conforme figura 12, afetam o estado de gerenciamento de um objeto em relação à disponibilidade dos recursos associados:
  - operacionalização: ver se um dado recurso está ou não em operação;
  - utilização: ver se um recurso está ou não em uso e se pode aceitar ou não outros usuários;
  - administração: atribui permissão ou proibição quanto ao uso de um recurso.

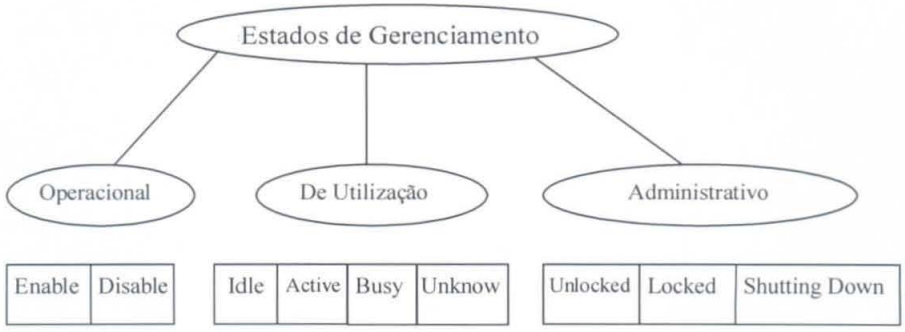

Figura 12: Estados de gerenciamento (EMBRATEL, 1994, p.517).

- Atributos para Representação de Relacionamento – ARR (*Attributes for Representing Relationship*), que ocorre quando uma operação de um objeto gerenciado afeta outro objeto, e pode ser do tipo (EMBRATEL, 1994, p. 517-518):
  - direto: ocorre quando partes das informações de gerenciamento de um objeto identifica de forma expressa outro objeto gerenciado;
  - indireto: ocorre quando há dois ou mais relacionamentos diretos concatenados;
  - simétrico: ocorre quando os conjuntos de regras que governam as interações entre dois objetos são idênticos;
  - assimétrico: ocorre quando os papéis de dois objetos são complementares.
- Função de relatório de Alarme – ARF (*Alarm Report Function*), que permite monitorar a taxa de erros do sistema e a evolução do nível de severidade dos

alarmes, com o objetivo de manter as condições operacionais dos serviços do sistema gerenciado. Nesta função são definidas cinco classes de alarme e seis níveis de severidade (EMBRATEL, 1994, p. 519):

- Classes de Alarme;
  - alarme de comunicação: relacionado a procedimentos e processos usados na transferência de informações;
  - alarme de qualidade de serviço: Indica a degradação da qualidade do serviço;
  - alarme de processamento: indica falha de processamento ou de software;
  - alarme de equipamento: indica falha de equipamentos;
  - alarme ambiental: indica as condições do ambiente em que se encontra em funcionamento o objeto gerenciado;
- Níveis de Severidades:
  - *cleared*: remoção de alarmes;
  - indeterminado: sem possibilidade de definir como foram afetadas as condições de funcionamento;
  - crítico: exigência de ação corretiva imediata;
  - maior: condições de funcionamento afetadas;
  - menor: necessário prevenir ocorrências de falhas;
  - alerta: suspeita de ocorrência de falhas.

- Função de Gerenciamento de Relatório de Evento - ERMF (*Event Report Management Function*), sendo que esta função tem como objetivo controlar o repasse de relatórios de eventos. Os relatórios de eventos potenciais são obtidos a partir das notificações geradas pelos objetos gerenciados e processadas através do componente de processamento e detecção de eventos. Estes relatórios são distribuídos aos discriminadores de repasse de eventos existentes no sistema, conforme demonstra a figura 13.

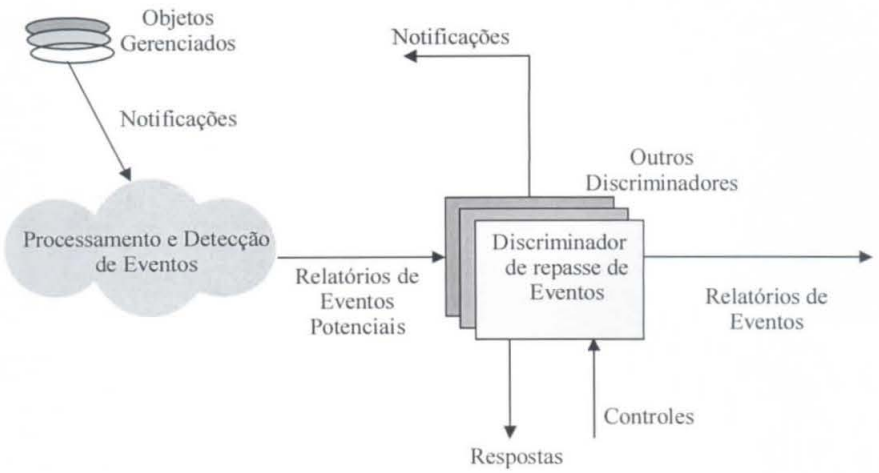

Figura 13: Gerenciamento de relatórios de falhas (EMBRATEL, 1994, p.521)

- Função de controle de *Log* – LCF (*Log Control Function*): o *log* nada mais é do que um repositório de registros, os quais contêm informações sobre eventos ocorridos ou por operações executadas pelos objetos gerenciados que precisam ser preservados. O controle da preservação desses registros é o objetivo da função de controle de *log* (EMBRATEL, 1994, p. 521);
- Função de registro para Auditoria de Segurança – SATF (*Security Audit Trail Function*): através desta função são gravados todos os eventos relativos à segurança em registros de *log* de auditoria de segurança, seguindo o modelo da função de registro de *log*. A partir da análise dos relatórios e alarmes de segurança poderá ter controle da manutenção da política de segurança especificada para o sistema, com a possibilidade de detectar desvios quanto à mesma, ou ainda pontos vulneráveis de segurança (EMBRATEL, 1994, p. 523).

A coleção de registros de auditoria de segurança pode ser local ou remota. No caso local é necessário ter a possibilidade do sistema gerente recuperar os registros, enquanto que no caso remoto os registros são enviados ao sistema gerente à medida que os eventos de segurança ocorrem. No entanto, para que seja possível o envio dos eventos ao sistema gerente é preciso ter no sistema local um Discriminador de repasse de eventos, conforme demonstra a figura 13. Cuja função é filtrar os

relatórios de eventos a serem enviados além de determinar o endereço dos sistemas destinatários dos eventos (BRISA, 1993, p. 98);

Segundo a BRISA (1993, p. 99) existem dois tipos de registros de auditoria de segurança:

- Relatório de serviço: indica se o registro está associado a um evento ligado à provisão, negação ou recuperação de um serviço;
- Relatórios estatísticos: indica se o registro contém informações estatísticas.

- Função de controle de acesso (OAAC): a função de controle de acesso permite ao administrador prevenir-se contra acessos não autorizados aos recursos gerenciados. Através dessa função podem-se fazer avaliações dos pedidos de acessos a objetos gerenciados para que as decisões de permissão ou negação do acesso sejam feito de acordo com a política de segurança, considerando fazer parte desta a política de controle de acesso.

Os objetos de controle de acesso, no gerenciamento OSI, são utilizados nas funções de controle de acesso da associação de aplicação e das operações de gerenciamento. Essa associação de gerenciamento é definida na estrutura da camada de aplicação, de sistemas abertos diferentes, com o objetivo de troca de informações necessárias à interoperação entre tais aplicações, conforme a figura 14 (BRISA, 1993, p. 102).

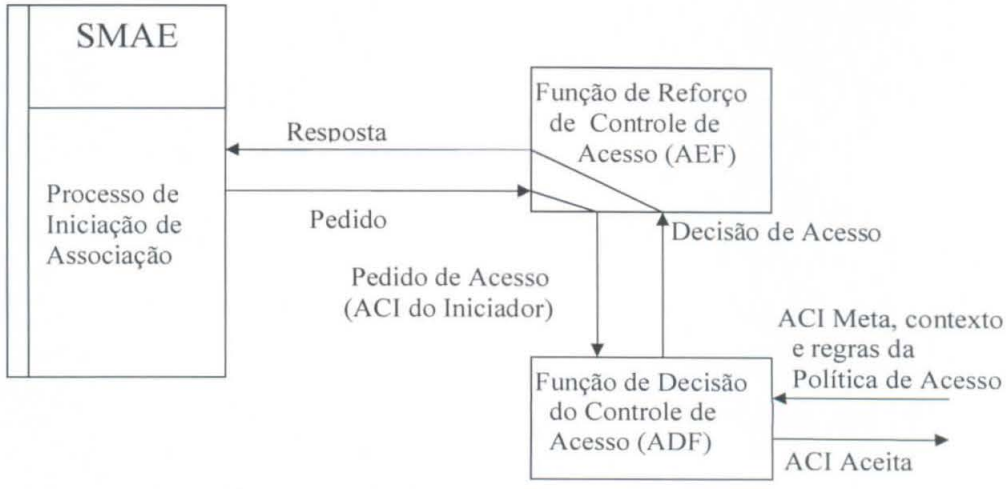

Figura 14: Modelo de controle de Acesso (BRISA, 1993, p. 103).

2.2.5.2. Gerenciamento de configuração

Para o bom gerenciamento de uma rede é extremamente importante manter a documentação atualizada com os registros técnicos relativos à instalação, inicialização, modificação e demais registros de parâmetros de configurações dos agentes e gerentes. Inclui-se ainda nesse gerenciamento de configuração o projeto da rede com localização e especificação de todos os pontos e elementos pertencentes à mesma.

2.2.5.3. Gerenciamento de Falhas

O gerenciamento de falhas tem como principal objetivo auxiliar o administrador da rede quanto ao número, tipos, horas das ocorrências e as localizações dos erros apresentados na rede, com o objetivo de que o retorno da rede ao estado operacional ocorra o mais breve possível. Logo, quando ocorre uma falha deve haver uma concentração das atividades para localizar, isolar a falha do restante da rede, recuperar os componentes e serviços em falha, identificar a causa do problema e registrar a solução para fins de uso futuro (BRISA, 1993, p. 18).

No gerenciamento de falhas, o administrador da rede, com base nas informações estatísticas, poderá propor projetos que contemplem soluções de tolerância à falhas, com o uso de elementos redundantes, sobressalentes, rotas de comunicação alternativas e etc (FREITAS, 2001, p.9).

2.2.5.4. Gerenciamento de Desempenho

Quando ocorre uma queda no desempenho da rede, os usuários reclamam quase que incansavelmente à medida que vêem as suas atividades serem prejudicadas. Portanto, os componentes de uma rede precisam ser monitorados de forma contínua e os dados estatísticos sobre o desempenho da rede devem ser registrados para subsidiar o planejamento e o controle de qualidade dos serviços disponíveis na rede. A partir destes dados, o administrador passa a possuir elementos que sustentarão ações de pró-atividades no sentido de antecipar-se ao surgimento de problemas.

Para gerenciar o desempenho da rede devem ser seguidos os passos: coletar dados do uso corrente dos dispositivos da rede, analisar os dados relevantes, estabelecer limiares

e simulação da rede. Quanto aos serviços, é importante ter registros do tempo de resposta, taxa de rejeição e disponibilidade (FREITAS, 2001, p.9).

2.2.5.5. Gerenciamento de Contabilização

A função de contabilização é medir a utilização dos recursos da rede para fins de quantificação dos custos, tarifação, cotas de utilização, etc. Este gerenciamento, além de prover um melhor acompanhamento do aumento de uso dos recursos da rede, poderá apresentar indícios de necessidade de novas aquisições de elementos para a mesma. A gerência de contabilização facilita o trabalho do administrador no sentido de possibilitar a identificação de usuários que abusam dos privilégios de acesso, uso ineficiente e até mesmo no planejamento de crescimento da rede com base no aumento da demanda (EMBRATEL, 1994, p.515).

2.2.5.6. Gerenciamento de Segurança

O controle de acesso à rede e às informações deve garantir que somente pessoas ou aplicações autorizadas tenham permissão aos recursos da rede. O objetivo é proteger o maior bem para as instituições atualmente: a informação. Dentre as funções do gerenciamento de segurança, destacam-se (BRISA, 1993, p. 87):

- a coleta, armazenamento e análise dos registros de auditoria e *logs* de segurança, bem como ativação e desativação de diretivas de auditoria;
- o registro das tentativas de ataques de forma efetiva;
- a definição das normas, políticas e procedimentos de segurança para a empresa, de modo que proteja a rede contra os acessos indevidos e sirva de subsídios para projetos de implementação de mecanismos de segurança, como: *firewall*, técnicas de criptografias, armazenamento seguros das informações.

2.3. Protocolo CMIP

Como mencionado anteriormente, a ISO (*International Organization for Standardization*) estabeleceu como solução para gerenciamento de redes o Protocolo CMIP (*Common Management Information Protocol*), o qual fornece um formato inteligível comum para a

transferência de informação de gerenciamento entre entidades pares. Neste caso, um dos pares atua como gerente e outro como agente durante a comunicação de gerenciamento para troca de informações sobre os objetos gerenciados e armazenados nas respectivas MIBs (MIB gerente e MIB Agente) (BRISA, 1993, p.321).

O protocolo CMIP utiliza classes de operação que são relacionadas com os serviços do CMISE (Elemento de Serviço de Informação de Gerenciamento Comum), conforme Tabela 4, cujos serviços podem ser confirmados ou não-confirmados. Esses serviços são mapeados em operações aplicadas sobre objetos ou recursos gerenciados da rede. A tabela 3 apresenta as classes de operação e associações definidas em termos gerais (BRISA, 1993, p. 181).

Tabela 3: Classes de operação e associações definidas

Classes	**Descrição**
Classe de Operação 1	Síncrona relatando sucesso ou falha
Classe de Operação 2	Assíncrona relatando sucesso ou falha
Classe de Operação 3	Assíncrona relatando somente falha (erro)
Classe de Operação 4	Assíncrona relatando somente sucesso
Classe de Operação 5	Assíncrona resultado não relatado.
Classe de Associação 1	Somente a entidade de aplicação iniciadora da associação pode invocar operações
Classe de Associação 2	Somente a entidade de aplicação respondedora da associação pode invocar operações
Classe de Associação 3	Ambas as entidades de aplicação iniciadora e respondedora da associação podem invocar operações

Dentre as classes de operações e associações apresentadas, o Protocolo CMIP utiliza a classe de associação 3 e as classes de operação 1, 2 e 5.

Tabela 4: Relacionamento das classes de operação do CMIP com os serviços CMISE.

Serviço	**Tipo**	**Classe de Operação**
M-Event Report	Confirmado/não confirmado	2 ou 1/5
M-Get	Confirmado	2 ou 1
M-Cancel-Get	Confirmado	2 ou 1

M-Set	Confirmado/não confirmado	2 ou 1/5
M-Action	Confirmado/não confirmado	2 ou 1/5
M-Create	Confirmado	2 ou 1
M-Delete	Confirmado	2 ou 1

O protocolo CMIP, de forma semelhante ao protocolo SNMP, engloba vários tipos de PDUs (*Protocol Data Units*), as quais são apresentados na Tabela 5 (EMBRATEL, 1994, p. 512).

Tabela 5: PDUs do protocolo CMIP

PDUs	Descrição
M-Action	Execução de alguma ação sobre um objeto gerenciado
M-Create	Criação de uma instância de um objeto gerenciado
M-Delete	Remoção de uma instância de um objeto gerenciado
M-Get	Leitura de atributos dos objetos gerenciados
M-Set	Modificação de atributos de objetos gerenciados
MEvent-Report	Notificação de um evento associado a um objeto gerenciado

Para estabelecer a comunicação, o CMIP faz uso dos elementos de serviço de Controle de Associação – ACSE e do ROSE (*Remote Operations Service Element*). Por isto, convém destacar os serviços assumidos pelo CMIP em relação ao ACSE e ao ROSE e suas respectivas APDUS (*Application Protocol Data Units*), conforme demonstram as tabelas 6 e 7 (BRISA, 1993, p.182).

Tabela 6: Serviços do CMIP em relação ao ACSE

APDUs	Serviços	Primitivas de Serviço
AARQ	A-Associate	A-Associate-Request e indication
AARE		A-Associate-Response e confirmation
RLRQ	A-Release	A-Release-request e indication
RLRE	A-Release	A-Release-response e Confirmation
ABRT	A-Abort	A-Abort-request e indication

Tabela 7: Serviços do CMIP em relação ao ROSE

APDUs	Serviços	Primitivas de Serviço

ROIV	RO-Invoke	RO-Invoke-request e indication
RORS	RO-Result	RO-Result-request e indication
ROER	RO-Error	RO-Error-request e indication
RORJ	RO-Reject	RO-Reject-U-request e indication

Nas comunicações entre aplicação gerente e agente, onde a aplicação gerente envia operações a serem executadas pelo agente sobre os objetos gerenciados e os agentes enviam notificações de gerenciamento ao gerente, para transporte dessas operações e notificações, são utilizadas as PDUs do ROSE. As operações remotas são identificadas dentro do contexto de aplicação da seguinte forma (BRISA, 1993, p.185):

- *REMOTEoperation {joint-isso-ccitt remote operation(4) apdu(1)}*
- *ROIV apdu(1), RORS apdu(2), ROER apdu (3) e RORJ apdu (4)*

Cada PDU do ROSE possui seus próprios parâmetros que são mapeados com os parâmetros do CMIP, conforme Tabela 8 (BRISA, 1993, p.186).

Tabela 8: Mapeamento dos parâmetros do CMIP nos parâmetros das APDUs do ROSE

APDUs do ROSE	**Parâmetros das APDUs**	**Parâmetros do CMIP**
ROIV (RO-INVOKE)	Invoked-Id	Invoked-Id
	Linked-Id	Linked-Id
	Operation Value	Operation Value + Mode
	Argument	Base Object Class Base Object Instance Manage Object Class Manage Object Instance Attribute List Access Control Action Type Action Information Current Time Action Reply Errors Linked Identifier Scope Filter Synchronization
RORS (RO-RESULT)	Invoked – Id	Invoked-Id

	Operation Value	Operation Value
	Result	Managed Object Class Managed Object Instance Current Time Attribute List
ROER (RO-ERROR)	Invoked-Id	Invoked-Id
	Error Value	Error Value
	Error Parameter	AccessDenied-Error–Local Value(LV) 2 ClassInstanceConflict-ERROR-LV 19 ComplexityLimitation-ERROR–LV 20 DuplicateManagedObjectIntance- ERROR – LV 11 getListError ERROR – LV 7 invalidArgumentValue-ERROR –LV 15 invalidAtributeValue-ERROR-LV 6 invalididFilter-ERROR-LV 4 invalidScope-ERROR-LV 16 invalidObjectInstance-ERROR-LV 17 missingAttributeValue-ERROR-LV 18 noSuchAction-ERROR-LV 9 noSuchArgument--ERROR-LV 14 noSuchAttribute--ERROR-LV 5 noSuchEventType-ERROR-LV 13 noSuchObjectClass-ERROR-LV 0 noSuchObjectInstance-ERROR-LV 1 noSuchReferenceObject-ERROR-LV 12 processingFailure-ERROR-LV 10 setListError-ERROR-LV 8 syncNotSupported-ERROR-LV 3
RORJ(RO-REJECT)	Invoked-Id	Invoked-Id
	Problem	General Problem Invoke Problem Return Result Problem Return Result Problem

Convém explicitar que os parâmetros do CMIP correspondentes ao parâmetro argumento (*Argument*) da APDU ROIV referem-se a todas as operações e notificações que podem ocorrer. Já os parâmetros CMIP mapeados com o *Error Parameter* (RO-ERROR) dizem respeito às notificações ou com quaisquer das operações que estão sendo realizadas em uma determinada associação (BRISA, 1993, p.187).

Dentro da abrangência do protocolo CMIP, ainda existem alguns recursos relevantes que facilitam o gerenciamento e merecem destaque, como os disponibilizados pelo Serviço de Informação de Gerenciamento Comum (CMIS) (EMBRATEL, 1994, p.531):

- *scoping*: permite realizar uma única operação sobre um grupo de instância de objetos selecionado;
- filtro: permite definir um conjunto de testes a ser aplicado a um grupo de instâncias de objetos previamente selecionado através do *Scoping*, proporcionando formar um grupo menor a partir deste, sobre o qual as operações de gerenciamento devem ser aplicadas;
- sincronização: sincroniza várias operações de gerenciamento realizadas sobre instâncias de objetos selecionados por meio dos recursos de *Scoping* e de Filtro;

Analisando a quantidade de recursos providos pelo protocolo CMIP, é notável a diferença em relação ao Protocolo SNMP, do Gerenciamento Internet. Este é o protocolo concebido para o padrão de sistemas de gerenciamento distribuídos de grandes e complexas redes corporativas e redes públicas de telecomunicações.

Como o trabalho proposto aborda a questão da segurança e que para promover o gerenciamento de redes é necessário prover segurança aos dados armazenados nos equipamentos da rede e às informações que trafegam na mesma, convém abordar conceitos relativos à segurança em redes de computadores.

2.4. Segurança de Redes

Segundo Wadlow (2000, p.4), "segurança é a direção que se pode viajar, mas nunca chegar de fato ao destino", ou seja, é o caminho que tem começo, mas não tem fim. "Segurança é um processo que se pode aplicar seguidamente à rede e à empresa que a mantém e, assim, melhorar a segurança dos sistemas. Se não iniciar ou interromper a aplicação do processo, a segurança será cada vez pior, à medida que surgirem novas ameaças e técnicas" (Wadlow, 2000, p.4). O processo de segurança que deve ser aplicado continuamente, se resume em analisar o problema, sintetizar uma solução para o problema com base na análise realizada e avaliar a solução empregada procurando entender e aprender em quais aspectos não correspondeu às expectativas. Nesse contexto o gerente de segurança deve ter habilidades e disponibilidade para aprender o máximo que puder sobre os tipos de ameaças encontradas, saber planejar soluções com base no que aprendeu antes de aplicá-las e em seguida implementá-las sendo fiel ao que foi projetado.

Esse trabalho deve ser constante porque os sistemas e informações a serem protegidos mudam ou sofrem alterações frequentemente, e, ao mesmo tempo surgem

novas vulnerabilidades, novos riscos e tipos de ameaças à segurança. As novas ameaças são originadas da dedicação dos criminosos e terroristas que se dedicam em adquirirem conhecimentos sobre as novas tecnologias e como violarem os sistemas (HORTON, 2003, p.4). Portanto, métodos de segurança devem sempre ser inovados e readequados para manter um bom nível de segurança. É um árduo trabalho, porém, extremamente necessário para uma empresa ou instituição.

Para promover uma efetiva proteção da rede de dados de uma instituição, devem ser seguidas algumas diretrizes que são fundamentais e que fazem parte do ciclo de vida da segurança da informação, conforme apresentado na figura 15. Estas diretrizes abrangem a proteção dos bens da rede (servidores, computadores e etc) com base no gerenciamento dos riscos, conhecimento sobre as ameaças e vulnerabilidades da rede (HORTON, 2003, p.5).

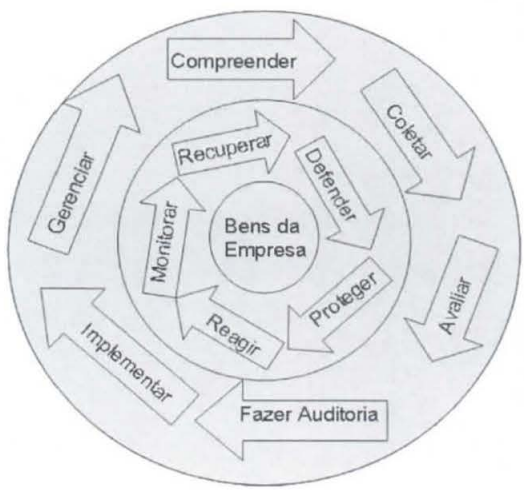

Figura 15: Processo de Segurança baseado em Riscos e Bens (HORTON, 2003, p.5).

A seguir será explanada cada ação a ser executada dentro do ciclo vida para que seja mantida a segurança das informações.

- compreensão e conhecimento dos bens da instituição, dos produtos e serviços, da estrutura organizacional e seus objetivos;
- coleta de informações sobre a infra-estrutura computacional e da rede, o que existe de proteção e o que precisa ser protegido;
- avaliação das estratégias para a rede e da arquitetura computacional, definindo a sua inserção no papel e objetivos da instituição, considerando os mecanismos de proteção existentes que estão sendo aplicados e o que está proposto;

- auditoria dos recursos de rede para obter conhecimento da segurança atual e os pontos críticos que devem ser protegidos para fechar as vulnerabilidades existentes;
- implementação de ações corretivas;
- gerenciamento dos controles e mecanismos de proteção aplicados;
- implementar mecanismos de proteção para os bens da instituição;
- fazer monitoração com base no processo de auditoria e registrar alertas e dados do sistema, avaliando essas informações nos eventos de segurança. A maior parte de tempo deve ser dedicada nesta fase de monitoração;
- reação quanto ao preparo de recursos para iniciar a defesa e a recuperação em tempo hábil, quando houver incidentes; e
- reavaliação das necessidades de segurança.

A execução dessas funções de forma cíclica propicia um ambiente com bom nível de segurança para uma instituição, protegendo o que existe de mais valioso atualmente, a informação. Vale ressaltar que prejuízos incalculáveis são imputados a empresas e instituições que não dão valor e, consequentemente, não investem o necessário no processo de segurança da informação. Os prejuízos são calculados com base nos valores tangíveis e intangíveis, por exemplo, a perca de confiança dos clientes.

A segurança de uma rede, considerando ser o meio de acesso às valiosas informações de uma empresa ou instituição, deve se ater a três objetivos principais: confidencialidade, integridade e disponibilidade. Dentro do objetivo de confidencialidade os dados devem estar disponíveis somente para quem precisa ter acesso a eles, mantendo o controle de acesso e evitando acessos indevidos. O fator integridade é bastante crítico e devem ter medidas severas de proteção para garantir que os dados não sejam alterados indevidamente. Contudo, não adianta garantir a confidencialidade e proteger os dados contra alterações indevidas, se não incluir garantias de que os dados estejam disponíveis quando necessários. A falta de disponibilidade poderá causar grandes prejuízos à empresa ou instituição, principalmente danos financeiros (HORTON, 2003, p.7). Portanto, a quebra de uma ou mais dessas três propriedades constitui ameaça à segurança da informação de qualquer instituição. Existem diversas formas e ferramentas que podem ser aplicadas no processo de segurança de uma rede, como (HORTON, 2003, p.12):

- *firewalls* de rede e de estação;
- IDS (Sistema de detecção de intrusão) de rede e de estação;
- servidor *proxy*;

- VPN de hardware/software;
- implementações de VLANs dos *Switchs*;
- softwares de gerenciamento e verificação de vulnerabilidades;
- softwares de alerta e monitoração de *logs*;
- softwares de Antivírus de servidor e estação;
- servidor central de autenticação;
- servidor de controle de diretivas e acesso;
- máquina virtual;
- criptografia de arquivos e de e-mails; e
- auditoria/avaliação de segurança da rede.

Como apresentado na relação acima, dentre os meios de prover segurança em uma rede, inclui o processo de auditorias. Com a aplicação desse processo em uma rede ocorre a gravação, em arquivos de *logs*, dos registros das ações dos usuários sobre os objetos que estão sendo auditados. Através da análise desses registros é possível traçar o comportamento de uso da rede, dos usuários e dos serviços, o que facilita a administração da rede, até mesmo, pelo simples fato de detecção de desvios de padrões conhecidos e aceitáveis para a rede em análise (BRISA, 1993, p.97).

Na rede Windows, o processo de auditoria é possível através de aplicação de diretivas de segurança de auditoria, as quais fazem parte da estrutura lógica e administrativa da rede construídas a partir da criação do serviço de diretório do Windows, denominado de *Active Directory*. Para melhor entendimento dos serviços providos pelo *Active Directory* faz-se necessário apresentar a infra-estrutura da rede Windows.

2.5. *Infra-estrutura da rede Windows*

Uma rede do Windows normalmente é composta por um ou mais servidores e estações de trabalho. O servidor funciona com o sistema operacional Windows Server (Versão 2000, 2003, 2008 ou versão posterior) e possui a função de controlador do domínio da rede. Ao assumir essa função ocorre a construção da estrutura dos Serviços de Diretórios (*Active Directory*), o qual é a base para a implementação da infra-estrutura da rede.

O *Active Directory* funciona como uma base de dados da rede onde são cadastrados todos os objetos da mesma, como: usuários e grupos de usuários, computadores, impressoras, unidades organizacionais, esquemas e etc. Essa base

contempla todas as informações necessárias para funcionamento da rede baseada no *Windows 2003 Server* (BATTISTI, 2006. Parte 2, p4).

Segundo Battisti (2006. Parte 1, p. 3), o *Active Directory* possui duas estruturas: uma física e outra lógica. A estrutura lógica é formada pelos elementos: domínios, árvores, florestas, relações de confiança, objetos do *Active Directory*, unidades organizacionais e esquemas, enquanto que a estrutura física é responsável por definir o local de armazenamento das informações do *Active Directory* e como será feita a sincronização das informações entre os controladores de domínio *master* e membros de um domínio.

Na definição da estrutura lógica constituem-se os limites de segurança e da gerência administrativa da rede. No topo da hierarquia administrativa temos o conceito de floresta, a qual pode ser formada por uma ou mais árvores de domínios. Dentro de uma floresta a relação de confiança entre as árvores existentes e os respectivos domínios se dá de forma automática. No entanto, se a rede possuir mais de uma floresta, a relação de confiança entre as mesmas não se dá de forma automática, estabelecendo nesse ponto o limite de segurança da rede.

Uma árvore pode ser formada por um ou mais domínios, sendo que o domínio de uma rede representa o limite de gerenciamento do conjunto de objetos do *Active Directory* que faz parte desse domínio. Em um domínio poderá existir um ou vários controladores do domínio, os quais compartilham e gerenciam a mesma base de dados do *Active Diretory*. A idéia de construir uma rede com vários domínios ou subdomínios proporciona uma independência administrativa entre alguns setores de uma empresa ou instituição (DOMÍNIOS, 2004, *On-Line*).

Apesar da independência administrativa estabelecida pela presença de domínios e/ou subdomínios existentes em uma rede, o processo automático da relação de confiança por pertencerem a uma mesma árvore ou floresta propicia a formação do catálogo global da rede através dos vários conjuntos dos objetos dos serviços de diretórios (BATTISTI, 2006. Parte 4, p. 4).

O conceito de esquema do *Active Directory* é definido pela constituição de classes de objetos e respectivos atributos, contendo as informações essenciais para os objetos do serviço de diretório. A classe contém todos os atributos capazes de definirem as informações que deve conter no objeto, contudo, um atributo pode pertencer a várias classes, uma vez que são definidos de forma separada da classe. Para definir os atributos

de uma classe, é realizada uma associação entre a classe e os respectivos atributos (DIRECTORY, 2005, *On-Line*).

Na constituição de um domínio para a rede Windows, o primeiro controlador, denominado de *master*, deverá agregar também a função de servidor de DNS (*Domain Name System*), visto que o *Active Directory* simplesmente não funciona sem ele. O servidor DNS do *Windows Server*, versão 2000 e posteriores, além de possuir a função de resolver nomes dos objetos do *Active Directory*, mais precisamente dos computadores e servidores, fazendo uma associação entre os nomes dos objetos e os respectivos números dos endereços IPs (*Internet Protocol*), agregou as funções de atualizações dinâmicas definidas pela RFC 2136 . Essa atualização dinâmica faz com que as bases de dados de nomes sejam atualizadas de forma automática, por exemplo, ao inserir um objeto computador no *Active Directory* automaticamente ocorre a inserção no DNS também, o que faz permitir ao *Active Directory* manter boa parte de sua informação no DNS (MINASE, 2001, p.863). Desta forma o Servidor de DNS dinâmico do *Windows Server* veio facilitar o trabalho do administrador, eliminando o trabalho despendido no DNS convencional, onde o administrador teria que alimentar o arquivo *hosts,* manualmente, digitando o nome e o respectivo endereço IP para que os nomes fossem resolvidos através do DNS.

Dentro da estrutura do *Active Directory* fazem presentes também as diretivas de segurança, que por sua vez incorporam as diretivas de auditorias que são as ferramentas utilizadas para aplicação de auditoria em uma Rede Windows.

2.6. Auditoria em Redes Windows

No Sistema Windows, os processos de auditorias de segurança são realizados através de diretivas de grupo de auditoria. Essas diretivas, através dos recursos dos serviços de diretório, podem ser aplicadas em todos os computadores e/ou usuários de um domínio, ou em apenas alguns ou ainda em modo local (AUDITORIA, 2004, *On-Line*).

Para possibilitar a ocorrência de auditoria de segurança é necessário que os *logs* de auditoria sejam gravados com os eventos relativos à segurança. Esta gravação pode ocorrer tanto localmente em cada máquina auditada, como também de forma remota, dependendo do tipo de auditoria definida. Portanto, os eventos de auditoria utilizados para detecção de mau uso dos recursos da rede, atentados contra a segurança das informações

ou de problemas que prejudiquem a concretização da política de segurança, como alteração indevida das configurações de auditorias, precisam ser controlados pelos usuários do grupo de gerenciamento de segurança (BRISA, 1993, p.98).

As opções de auditoria, por padrão em qualquer sistema, possuem *status* de não configuradas ou não definidas, exatamente porque exigem definição do que precisa ser auditado de acordo com a política de auditoria da empresa ou instituição, o que é bastante peculiar a cada interesse. Para realizar uma boa análise de auditoria de segurança, dois passos são necessários: primeiramente é preciso definir e aplicar a política de auditoria e, posteriormente, é necessário ter o conhecimento sobre a estrutura dos arquivos de *logs* gerados para auditoria.

As diretivas de auditoria da rede Windows, quando aplicadas, geram arquivos de *logs* dos eventos de várias áreas de segurança, como (AUDITORIA, 2004, *On-Line*):

- acesso a objetos;
- acesso ao serviço de diretório;
- alteração de diretiva;
- controle de processo;
- eventos de *logon*;
- eventos de sistema;
- gerenciamento de conta;
- uso de privilégios; e
- eventos de *logon* de conta.

É notório que ativar todas essas diretivas de auditoria de segurança tornam impraticável a gerência e a análise dos arquivos de *logs* sem uma boa ferramenta de administração e filtragem desses *logs* e grande espaço em disco, visto a quantidade excessiva de *logs* de eventos que são gerados. Todas essas diretivas de auditoria ativadas geram milhares de *logs* por dia para cada máquina, o que poderia prejudicar a performance do equipamento e consequentemente da rede. Portanto, a recomendação para se ter bons resultados oriundos da aplicação de auditoria em uma rede é a definição sensata da política de auditoria de segurança a ser adotada.

Uma boa política de segurança, segundo Wadlow(2000, p.12), precisa atender a alguns propósitos, adequando-os a uma boa política de auditoria, tais como:

- descrever o porquê, como e o que será auditado;

- definir prioridades sobre o que precisa ser auditado, conforme a necessidade e o impacto da auditoria;
- definir métodos de retenção dos registros de eventos;
- definir meios de gerenciamento e acompanhamento sistemático dos registros de eventos;
- impedir que o departamento de segurança tenha um desempenho fútil e sem objetivo perante a administração da instituição;
- permitir um acordo explícito e o convencimento das várias partes da instituição sobre o valor da auditoria.

O maior desafio para o gerenciamento de uma rede Windows através de uma política de auditoria se concentra na forma como coletar as informações dos *logs* que são gravados localmente em cada máquina. Isto porque se conectar, mesmo que de forma remota, em cada estação para fazer análise dos *logs* é uma tarefa impraticável, cujos objetivos dificilmente serão alcançados. Portanto, como solução desse impasse projeta-se a busca de métodos capazes de fazer coletas dessas informações de forma automática e armazená-la em uma base de dados. Contudo, antes de descrever os métodos e os recursos que serão utilizados, convém reforçar a base teórica sobre as normas, tecnologias e recursos necessários para esta finalidade.

2.7. *Windows Management Instrumentation* (WMI)

WMI é uma tecnologia da Microsoft que permite a manipulação de informações úteis para gerenciamento e controle de redes corporativas Windows, construída com base nas definições do modelo OSI de gerenciamento de redes. WMI surgiu em 1998 como um componente no *Service Pack* 4 do Windows NT 4.0, mas, a partir do Windows 2000, passou a fazer parte dos componentes nativos dos Sistemas Operacionais Windows. Para as versões anteriores existe um pacote para *download* no próprio site da *Microsoft* (WMI, 2008, *On-Line*).

Essa instrumentação surgiu com base na iniciativa da WBEM (*Web-Based Enterprise Management*), sendo esta criada pela indústria responsável por buscar o desenvolvimento de tecnologias padrões para o acesso às informações de gerenciamento de ambientes corporativos. Como a Microsoft estava entre as empresas que propuseram a

criação desse padrão, a tecnologia WMI é conhecida como a implementação da Microsoft para o padrão WBEM (WMI, 2008, *On-Line*).

WMI pode ser considerada a tecnologia de gerenciamento primária para os Sistemas Operacionais da Microsoft. Essa tecnologia permite aos administradores de rede e de sistemas uma forma consistente para acessar informações de gerência de sistemas e foi projetado desde o início para trabalhar através de redes. Utilizando as informações de gerência acessadas por meio do WMI é possível fazer consultas, monitoração, configuração de computadores *desktop* e servidores, assim como do próprio Sistema Operacional, além de aplicações de rede e até mesmo dispositivos físicos. Portanto, entender o funcionamento dos recursos do WMI é tão importante e útil quanto entender o funcionamento dos serviços do *Active Directory*.

Utilizando a biblioteca de *Scripting* WMI, administradores de sistemas podem criar *scripts* para trabalhar com WMI e criarem diversos sistemas de gerenciamento e *scripts* de monitoramento, o que facilita sobremaneira o árduo trabalho de gerência de redes, principalmente pela possibilidade de realizar acessos tanto a máquinas locais como remotas.

A tecnologia WMI permite que o gerenciamento de redes Windows através do fornecimento de acesso aos objetos gerenciáveis seja feito de forma simples, utilizando os conceitos de orientação a objetos, incluindo classes, métodos, propriedades e relações entre os recursos gerenciáveis, cujos conceitos são compatíveis com o modelo OSI de gerenciamento de redes. Para cada recurso há uma classe WMI correspondente, de forma que exista uma descrição, mesmo que sucinta, das propriedades do recurso gerenciável e as ações que o WMI pode realizar para gerenciar o recurso (WMI, 2008, *On-Line*).

Um recurso ou objeto gerenciável através da tecnologia WMI pode ser um computador, software, serviços, conta de usuários e grupos, pastas, recursos compartilhados, eventos registrados em *logs*, arquivos de sistemas, segurança, componentes de redes, enfim, uma diversidade de recursos subordinados à gerência através do WMI. A grande vantagem da utilização do WMI se destaca pelo fato de não ser necessário o uso de diversas ferramentas para fazer a gerência dessa diversidade de recursos, dependendo apenas que o administrador saiba como escrever *scripts* que faz uso da biblioteca de *Scripting* WMI.

Utilizando o Windows *Script Host* (WSH), *VBScript* ou *Microsoft JSCript* é possível escrever *scripts* WMI para automação do gerenciamento de redes e sistemas.

WMI possui várias características e aplicações, as quais se destacam para o desenvolvimento do trabalho proposto (WMI, 2008, *On-Line*):

- *scripts* para administrar *logs* de eventos, segurança da rede e de sistemas, sistemas de arquivos, configurações, impressoras, processos e muitos outros componentes do sistema operacional;
- *scripts* para administrar serviços de redes, como: DNS, DHCP e outros; e
- *scripts* para monitoração em tempo real.

Uma das principais vantagens em utilizar a tecnologia WMI é a possibilidade de aplicar os *scripts* para fazer a gerência ou configurações em computadores tanto locais como remotos. Desta forma, torna-se possível obter *logs* de segurança dos computadores de uma rede, instalar software remotamente, obter a lista de usuários que estão logados em cada máquina da rede em tempo real, obter características dos equipamentos da rede e softwares instalados, enfim, recursos gerenciais que se tornaram possíveis graças ao surgimento do WMI. Antes do WMI, essas gerências eram realizadas através de ferramentas de terceiros e dificilmente tinha o acesso remoto, sendo que as ferramentas próprias do Windows eram bastante limitadas.

Executando o simples *script* WMI da tabela 9 no *prompt* de comando utilizando o *CScript* é possível obter o total da memória física em qualquer computador da rede, basta atribuir o nome do computador à variável *strComputer*. Para executar o *script* deve ser utilizada a sintaxe: "*cscript* [nome do *script*]" no *prompt* de comando de uma estação ou servidor Windows, sendo que o *script* abaixo deverá ser salvo com a extensão '.vbs' (WMI, 2008, *On-Line*).

Tabela 9: Exemplo de Script WMI – Memória (WMI, 2008, *On-Line*).

Linha	Sintaxe
1.	`strComputer = "."`
2.	
3.	`Set objSWbemServices = GetObject("winmgmts:\\" & strComputer)`
4.	`Set colSWbemObjectSet = _`
5.	`objSWbemServices.InstancesOf("Win32_LogicalMemoryConfiguration")`
6.	
7.	`For Each objSWbemObject In colSWbemObjectSet`
8.	`Wscript.Echo "Total Physical Memory (kb): " & _`
9.	`objSWbemObject.TotalPhysicalMemory`
10.	
11.	Next

Analisando o *script* da Tabela 9 comprova-se que para cada recurso gerenciável existe uma classe WMI correspondente. Na linha 5 é instanciada a classe "*Win32_LogicalMemoryConfiguration*" a qual contêm as propriedades tanto da memória física como virtual. Para mostrar a memória virtual basta substituir a propriedade do objeto para *TotalVirtualMemory* na linha 9 do *script* exemplo. Ao executar o *script* WMI exemplificado na Tabela 10 é possível obter todos os serviços de um determinado computador ordenados pelo nome, *status* e o modo de inicialização dos mesmos.

Tabela 10: Exemplo de Script WMI – Serviços (WMI, 2008, *On-Line*).

Linha	Sintaxe
1.	`strComputer = "."`
2.	
3.	`Set objSWbemServices = GetObject("winmgmts:\\" & strComputer)`
4.	`Set colSWbemObjectSet =`
	`objSWbemServices.InstancesOf("Win32_Service")`
5.	
6.	`For Each objSWbemObject In colSWbemObjectSet`
7.	`Wscript.Echo "Display Name: " & objSWbemObject.DisplayName &`
	`vbCrLf & _`
8.	`" State: " & objSWbemObject.State & vbCrLf & _`
9.	`" Start Mode: " & objSWbemObject.StartMode`
10.	`Next`

Observa-se que no exemplo da Tabela 10, linha 4, está sendo instanciada a classe de objetos de serviços do Windows "*Win32_Service*", e que as propriedades são exclusivas para cada tipo de objeto. As classes WMI são na prática representatividades virtuais da realidade, enquanto que as propriedades dos objetos são as mesmas dos objetos reais. Outra análise importante sobre WMI é a existência de semelhanças entre os *scripts*. Por exemplo, os *scripts* da Tabela 9 e Tabela 10 basicamente diferem somente na classe a ser instanciada e nas propriedades dos objetos. Logo, escrever *scripts* WMI não é uma tarefa muito complicada, uma vez que envolve apenas três passos (WMI, 2008, *On-Line*):

- a conexão com o serviço WMI;
- recuperação de informações sobre a classe WMI; e
- realizar alguma tarefa com as informações.

Como exemplo da semelhança entre os *scripts* WMI a Tabela 11 apresenta um *script* que recupera e mostra os *logs* de eventos do Windows.

Tabela 11: Exemplo de Script WMI – Logs de Eventos (WMI, 2008, *On-Line*).

Linha	Sintaxe

```
1.  strComputer = "."
2.  Set objSWbemServices = GetObject("winmgmts:\\" & strComputer)
3.  Set colSWbemObjectSet =objSWbemServices.InstancesOf("Win32_ NTLogEvent")
4.  For Each objSWbemObject In colSWbemObjectSet
5.  Wscript.Echo "Log File: " & objSWbemObject.LogFile & vbCrLf & _
6.  "Record Number: " & objSWbemObject.RecordNumber & vbCrLf & _
7.  "Type: " & objSWbemObject.Type & vbCrLf & _
8.  "Time Generated: " & objSWbemObject.TimeGenerated & vbCrLf & _
9.  "Source: " & objSWbemObject.SourceName & vbCrLf & _
10. "Category: " & objSWbemObject.Category & vbCrLf & _
11. "Category String: " & objSWbemObject.CategoryString & vbCrLf & _
12. "Event: " & objSWbemObject.EventCode & vbCrLf & _
13. "User: " & objSWbemObject.User & vbCrLf & _
14. "Computer: " & objSWbemObject.ComputerName & vbCrLf & _
15. "Message: " & objSWbemObject.Message & vbCrLf
16. Next
```

Analisando o *script* da Tabela 11, as linhas 2, 3 e 4 mostram os três passos a serem seguidos para escrita de *scripts* WMI, assim como apresenta a estrutura quase que permanente do *script*. Comparando com os *scripts* dos exemplos anteriores, nota-se que houve alterações apenas com relação às classes instanciadas e as propriedades dos objetos. Essas características facilitam o entendimento e o trabalho de escrita de *scripts* WMI para as diversas finalidades que se propõe através das informações obtidas dos objetos gerenciáveis.

2.7.1. Arquitetura WMI

Apesar da facilidade para se construir *scripts* WMI que possibilitam obter informações gerenciais a partir de um *script* padrão, torna-se necessário conhecer quais elementos, classes, métodos e propriedades que podem ser utilizados no *script* padrão para alcançar o objetivo esperado. Vale ressaltar que também é importante conhecer como WMI trabalha e como e onde são armazenadas as informações do WMI. Para um bom trabalho baseado em *script* WMI é indispensável que o administrador saiba responder três questões básicas, como (WMI, 2008, *On-Line*):

- quais classes estão disponíveis;
- quais são os nomes dessas classes; e
- quais propriedades e métodos podem ser usados para cada classe.

O Modelo de Informações Comuns (CIM) fornece uma cobertura detalhada da arquitetura de WMI, o que traz uma compreensão mais abrangente para construção de ferramentas gerenciais baseadas na tecnologia WMI. A arquitetura WMI se desenvolve em três camadas principais a seguir, as quais são mostradas também na figura 16 (WMI, 2008, *On-Line*).

- clientes;
- infra-estrutura WMI; e
- recursos gerenciados.

Além dessas três camadas principais, será explorado também na seqüência a forma como WMI trabalha com o aspecto segurança, visto ser bastante salutar o administrador de redes ou de sistemas entender todo esse conjunto de informações para então construir *scripts* úteis e eficientes para gerência de redes.

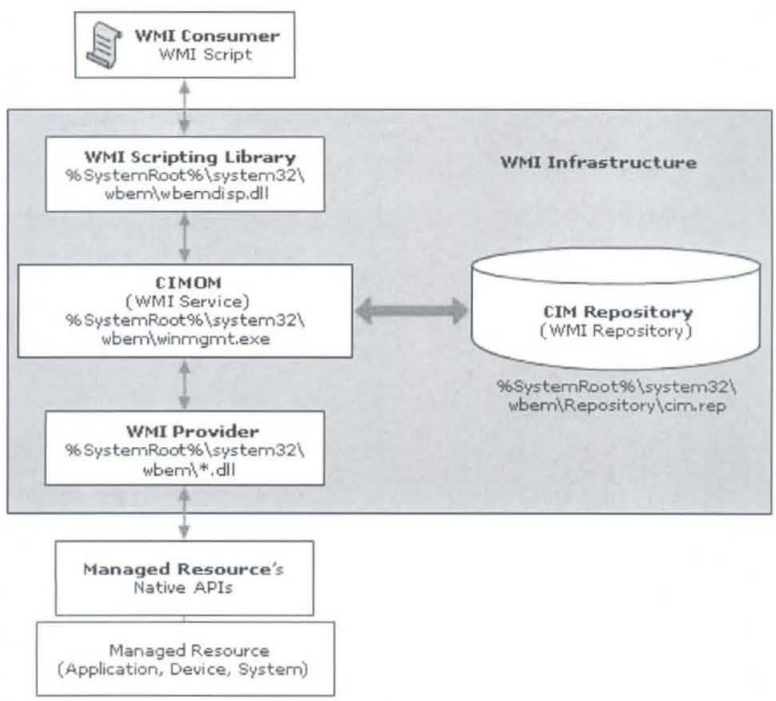

Figura 16: Arquitetura WMI (WMI, 2008, On-Line).

A estrutura, a composição e o funcionamento de cada camada apresentada na figura 16, serão expostos nos tópicos seguintes.

2.7.1.1. Recursos Gerenciados

Os recursos gerenciados formam a camada base da arquitetura WMI. Tais recursos podem ser componentes físicos ou lógicos, como: sistemas computacionais, discos, dispositivos periféricos, *logs* de eventos, arquivos e pastas, arquivos de sistemas, componentes de redes, subsistemas do Sistema Operacional Windows, impressoras, processos, configurações de registros, segurança, serviços, compartilhamentos, usuários e grupos, serviço de diretório do Windows (*Active Directory*), instalador do Windows, modelo de driver Windows (WDM) e também a base de informações de gerenciamento (MIB) (WMI, 2008, *On-Line*).

O recurso gerenciado comunica com o WMI através de um provedor. Como já descrito, a prática de construção de *scripts* para interagir com recursos gerenciados WMI faz uso do termo "instância", o qual se refere a uma representação de uma implementação atual de algum objeto que pode ser gerenciado pelo WMI, cuja representação é realizada através da classe instanciada. Por exemplo, o *script* da Tabela 9, que tem como resultado de sua execução uma representação real do total de memória física do computador, é realizada através da instância da Classe "*Win32_LogialMemoryConfiguration*".

2.7.1.2. Infra-Estrutura do WMI

A infra-estrutura é a camada intermediária do modelo arquitetural que consiste em 03 (três) componentes primários (WMI, 2008, *On-Line*):

- CIMOM: Gerente de objetos do Modelo de Informações Comum, conhecido também como WMI *Service*;
- CIM: Modelo de Informações Comum, conhecido também como repositório WMI; e
- WMI *providers*: Provedores WMI.

Esses três componentes formam a infra-estrutura sobre a qual as configurações e dados de gerenciamento são definidos, expostos, acessados e recuperados.

2.7.1.2.1. Provedores de WMI

Os provedores de WMI possuem a função de interligar o CIMON aos recursos gerenciados. Esses provedores solicitam informações e as enviam para os recursos gerenciados WMI em nome da aplicação cliente (WMI, 2008, *On-Line*). Por exemplo, o

script que recupera registro dos *logs* de eventos do Windows utiliza o provedor embutido *Event Log* e o que recupera o tamanho da memória utiliza o provedor Win32.

Outro fator importante reside na forma de comunicação que os provedores WMI fazem com seus respectivos recursos gerenciados, cuja comunicação é realizada através das APIs (*Application Programming Interfaces*). Já a comunicação com o CIMOM (Gerente de objetos do Modelo de Informações Comum) se concretiza com o uso de interfaces de programação WMI (WMI, 2008, *On-Line*). Por exemplo, o provedor *Event Log* chama a API Win32 *Event Log* para acessar os *logs* de eventos. Veja que o administrador responsável por criar os *scripts* não interage diretamente com as APIs, o próprio provedor WMI estabelece o uso das mesmas de forma transparente para o administrador, até porque diversas APIs, tipo a *Win32*, não permitem ser chamadas através de *scripts*, o que forçaria o uso de linguagens de programação diferente e a perda das características de simplicidade de se trabalhar com WMI, como: construir *scripts* em poucos minutos, sem necessidade de uso de ferramentas diferentes do Notepad; além de a construção de *scripts* não requerer que seja incluso cabeçalho de arquivos, nem fazer uso de compilador.

Outra vantagem refere-se ao fato de que uma pessoa que necessita construir *scripts* WMI não tem a necessidade de conhecer as APIs dos recursos gerenciados nem as diferenças que existem. WMI possui a responsabilidade de traduzir todos os comandos WMI inseridos nos *scripts* em comando entendíveis pelas APIs.

Para ficar mais claro, os provedores WMI são implementados como bibliotecas de *links* dinâmicos residentes na própria estrutura de instalação do sistema operacional Windows (2000, XP, 2003 e posteriores), especificadamente no diretório "*systemroot\System32\Wbem*". Por se tratar de recursos nativos nos sistemas operacionais Windows, os desenvolvedores podem utilizá-los para incluir recursos de gerenciamento em seus produtos (WMI, 2008, *On-Line*).

Como existe uma diversidade de provedores padrões embutidos nos sistemas operacionais Windows, a Tabela 12 traz uma amostra desses provedores.

Tabela 12: Lista parcial dos provedores padrões WMI

Provedor	DLL	*NameSpace*	Descrição
Active Directory	Dsprov.dll	Root\directory\ldap	Mapas dos objetos do *Active Diretory* para WMI.
Event Log	Ntevt.dll	Root\cimv2	Administração dos *Logs de*

			Eventos do Windows, como: leitura, backup, limpeza, cópias, exclusão, renomeação, compactação e descompactação dos arquivos de *logs* de eventos e alteração nas configurações dos *logs* de eventos.
Performance Counter	Wbemperf.dll	Root\cimv2	Prover acessos a dados de apresentação.
Registry	Stdprov.dll	Root\default	Leitura, gravação, monitoração, criação e exclusão de chaves e valores dos registros.
SNMP	Snmpincl.dll	Root\snmp	Prover acesso aos dados da MIB SNMP.
Security Provider	Cimwin32.dll	Root\cimv2	Recupera ou altera configurações de segurança que controla propriedades, auditoria, direitos de acessos a arquivos, diretórios e compartilhamentos.
WDM	Wmiprov.dll	Root\wmi	Prover acesso às informações sobre *drivers* dos dispositivos WDM.
Win32	Cimwin32.dll	Root\cimv2	Prover informações sobre o computador, discos, dispositivos periféricos, arquivos e pastas, compartilhamentos, arquivos de sistemas, componentes de redes, sistemas operacionais, segurança, serviços, usuários e grupos etc.
Windows installer	Msiprov.dll	Root\cimv2	Prover acessos às informações sobre softwares instalados.

2.7.1.2.2. CIMOM

O CIMOM lida com interações entre os clientes e os provedores. O termo CIMOM surgiu da iniciativa do WBEM (*Web-Based Enterprise Management*) e a especificação do CIM (*Comom Information Model*) é mantida pelo DMTF (*Distributed Management Task Force*) (WMI, 2008, *On-Line*).

Todas as informações e solicitações passam pelo CIMOM. Os *scripts* são direcionados para o CIMOM, entretanto, este não lida diretamente com as informações. O CIMOM recebe as solicitações através dos *scripts* e localiza um provedor que possa atender as solicitações. Quando o provedor processa as requisições, os resultados são enviados para o CIMOM, que possui a responsabilidade de entregá-las para o usuário solicitante (WMI, 2008, *On-Line*).

Os serviços WMI (*winmgmt.exe*) fornecem regras para o CIMOM sobre o Windows XP e são semelhantes à maioria dos serviços do Sistema Operacional, podendo ser parados e inicializados utilizando os seguintes comandos "*Net stop winmgmt*" e "*Net start winmgmt*".

O CIMOM provê uma interface comum para os clientes acessarem o WMI e ainda fornece os seguintes serviços para a infra-estrutura do WMI (WMI, 2008, *On-Line*):

- *Provider Registration*: registro de provedores WMI e informações de capacidade com o CIMOM. As informações são armazenas no CIM pelo CIMOM;
- *Request routing*: o CIMOM utiliza das informações dos provedores para direcionar as requisições dos clientes para o provedor apropriado;
- *Remote Access*: os clientes acessam de forma remota os sistemas WMI habilitados conectando ao CIMOM. Uma vez estabelecida a conexão, com poucas exceções, qualquer operação que pode ser feita localmente pode também, facilmente, ser realizada no computador remoto;
- *Security*: o CIMOM faz o controle dos acessos aos recursos gerenciados WMI validando o cadastro de acesso de cada usuário antes de conceder permissão para conexão WMI ao computador local ou remoto;
- *Query processing*: permite ao cliente fazer consulta de qualquer recurso gerenciado WMI usando a linguagem de consulta WMI (WQL). Através dessa linguagem pode ser feita consulta de apenas um tipo (ID) específico de eventos que por acaso tenha ocorrido em um determinado tempo; e
- *Event Processing*: permite ao cliente subescrever eventos que representam as alterações em um recurso gerenciado WMI. Assim, o cliente pode submeter *scripts* que disparam alertas quando, por exemplo, o espaço de uma unidade lógica do disco está abaixo do tamanho aceitável. O CIMOM gerencia os recursos em um intervalo especificado e gera eventos de notificações quando a subscrição é satisfeita.

Com base nesses serviços, quando é feita solicitações para o CIMOM recuperar dados, inscrever-se para eventos, ou alguma outra tarefa relacionada com gerenciamento, o CIMOM recupera o provedor e informações da classe necessárias para os serviços solicitados do repositório CIM.

2.7.1.2.3. Repositório CIM

WMI baseia-se na idéia de que configuração e informações sobre a gestão de diferentes fontes poderão ser uniformemente representadas com um esquema. O repositório CIM, também conhecido como repositório de objetos ou de classes detém o esquema que modela o ambiente gerenciado e define várias partes dos dados expostos através do WMI. Tal esquema é baseado nas Informações Comuns do Modelo Padrão (DTMF) (WMI, 2008, *On-Line*).

O esquema CIM é semelhante a um projeto arquitetônico da estrutura física de uma casa, o CIM modela o hardware, o sistema operacional, e o software que constitui um computador. O CIM é um modelo de dado para WMI. Portanto, conhecer e saber como interpretar o projeto WMI para gerenciamento, denominado de repositório CIM, é de fundamental importância para quem for escrever *scripts* WMI. Conhecer o repositório CIM implica em saber navegar na estrutura CIM e poder determinar o computador e os recursos de softwares expostos através do WMI. Saber como interpretar o projeto de gerenciamento WMI, ou o projeto de um recurso gerenciado, ajuda a ter o conhecimento sobre quais tarefas podem ser executadas sobre o recurso gerenciado.

A figura 17 mostra uma visão conceitual da estrutura interna e da organização do repositório CIM. Como ilustrado na figura 17, o CIM usa classes para criar o modelo de dados. O CIM contém muito mais classes do que as mostradas no diagrama. Isto é importante para entender que o repositório CIM é o armazém de classes que define o ambiente de gerenciamento WMI e todos os recursos controláveis expostos através do WMI, *(WMI, 2008, On-Line)*.

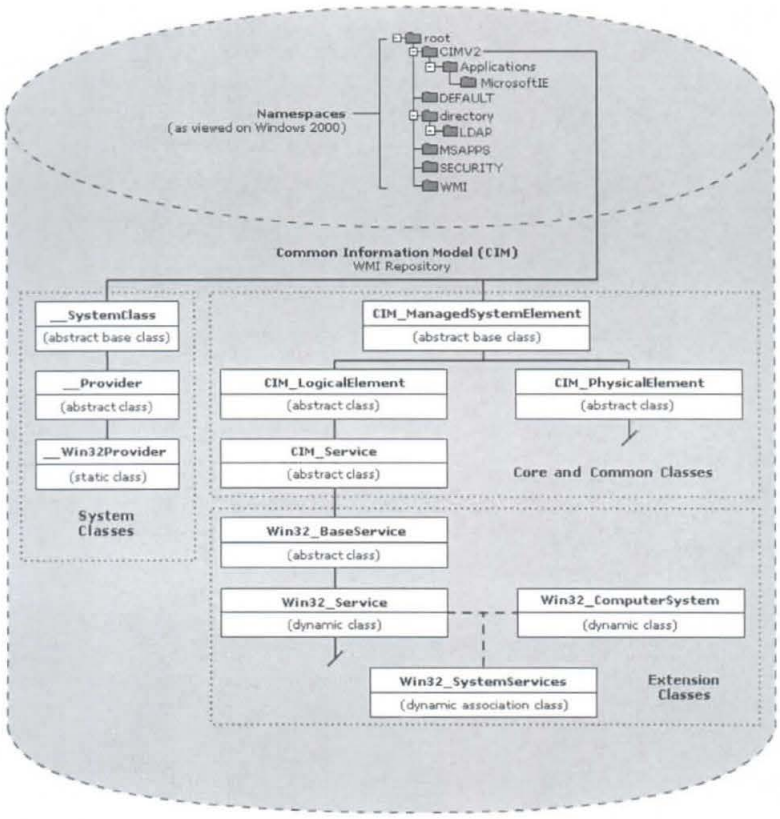

Figura 17: Estrutura interna do Repositório CIM (WMI, 2008, On-Line)

Detalhando a estrutura apresentada na figura17, destacam-se três conceitos do Modelo de Informações Comuns que são relevantes para aprender como navegar no repositório CIM de forma eficiente e saber interpretar o esquema WMI, (WMI, 2008, *On-Line*).:

1. O repositório CIM está dividido em vários *NameSpaces*;
2. Cada *NameSpace* pode conter um ou mais grupo das seguintes classes:
   - classes de sistemas;
   - classes comuns e de núcleo;
   - extensão de classes.
3. Há três tipos de classes primárias: abstrata, estática, e dinâmica. Um quarto tipo de classe, conhecida como classe de associação, é também suportado.
   - uma classe abstrata é o modelo usado para derivar novas classes abstratas ou não abstratas, e não pode ser usada para recuperar instâncias de recursos gerenciados;

- uma classe estática define os dados que são armazenados fisicamente no repositório CIM, dos quais os mais comuns é a configuração WMI e os dados operacionais;
- classes dinâmicas é classe que modela os recursos gerenciados WMI que são recuperados dinamicamente de um provedor;
- uma classe de associação pode ser abstrata, estática, ou dinâmica. Utilizada para descrever um relacionamento entre duas classes ou recursos gerenciados.

De forma análoga ao esquema do *Active Directory*, o CIM é construído com base nos conceitos de classes, sendo que uma classe é um modelo de um recurso gerenciado. A diferença reside no fato de que as classes do *Active Directory* representam objetos estáticos enquanto as classes CIM representam recursos dinâmicos. Para melhor entendimento dos recursos dinâmicos, as instâncias de recursos não são armazenadas no repositório CIM, porém, são dinamicamente recuperadas através de um provedor com base nas solicitações do cliente. O dinamismo também reside no fato de o *status* operacional da maioria dos recursos gerenciados mudarem com freqüência, então, torna-se necessário que as informações recuperadas sejam as mais atuais possíveis. Como a execução de *scripts* WMI apresenta boa performance, não se resume em um complicador o fato de buscar informações sempre atuais, mesmo que o volume seja grande, (WMI, 2008, *On-Line*).

Para melhor entendimento e clareza dos conceitos apresentados na figura 17, faz-se necessário uma abordagem mais abrangente sobre as *NameSpaces* e sobre os tipos de classes do repositório CIM.

2.7.1.2.3.1. NameSpaces

NameSpaces são mecanismos de partições empregado pelo CIM para controlar o escopo e ter visibilidade de definições de classes dos recursos gerenciados. A grosso modo, *NameSpaces* é semelhante a pastas de um *drive* de armazenamento de dados. Cada *NameSpaces* no CIM contém um grupo lógico de classes relacionadas que representa uma tecnologia especifica ou uma área de gerenciamento. Assim como as pastas, os *NameSpaces* contribuem para que o usuário identifique de forma única cada item, por exemplo, uma *NameSpace* de nome *MicrosoftActiveDirectory* provavelmente deve conter classes WMI usadas para gerenciar o *Active Directory*. De forma similar às pastas que não

permite dois arquivos com mesmo nome dentro da mesma pasta, também não é permitido classes de mesmo nome dentro da mesma *NameSpace*, podendo, é claro, existir classes de mesmo nome em *NameSpaces* distintas (WMI, 2008, *On-Line*).

Uma diferença importante entre pastas e *NameSpaces* diz respeito à profundidade de níveis de subpastas. Nas pastas a profundidade é livre, podendo ter vários níveis de profundidade. Já nas *NameSpaces* dificilmente se encontrará mais do que três níveis de profundidades.

É importante saber que a maiorias das classes que modelam recursos gerenciados Windows estão armazenadas na *NameSpace* root\cimv2, tais como: logs de ventos, analisador de desempenho, instalador do Windows, e todos os provedores Win32.

O *script* da Tabela 13 mostra como especificar uma *NameSpace* para conexão, bem como exemplifica o uso da variável *strComputer*. Assim como no Windows, que ao executar um arquivo sem informar o caminho completo, o sistema procura nos caminhos especificados na variável %PATH%, quando não se especifica a *NameSpace* no *script*, assume-se que trata-se da *NameSpace* padrão configurada no Registro. A *NameSpace* padrão é configurada no seguinte endereço: HKEY_LOCAL_MACHINE\ SOFTWARE\Microsoft\WBEM\Scripting\Default *NameSpace*. No entanto, faz parte das boas práticas para construções de *scripts* a definição da *NameSpace* que será utilizada no *script* para realização das tarefas desejadas diretamente no *script* (WMI, 2008, *On-Line*).

Tabela 13: Como especificar uma NameSpace no script (WMI, 2008, *On-Line*).

Linha	Sintaxe
1.	strComputer = "."
2.	`Set objSWbemServices = GetObject("winmgmts:\\" & strComputer & "\root\cimv2")`

A parte inicial de um *script* WMI deve constar sempre o nome do computador sobre o qual será realizada alguma atividade através do *script* WMI. Quando não está especificado o nome do computador (ou a variável *strComputer* tem como valor o ponto (.)), o *script* será executado no computador local. No entanto, percebe-se a facilidade para executar o *script* em um computador remoto, pois será necessário apenas alterar o valor da variável *strComputer*. Esta mesma facilidade estende-se à atribuição do nome da *NameSpace* e da classe, visto que somente será necessário alterá-las em um corpo básico do *script* WMI.

WMI é uma tecnologia bastante versátil e sem muitas complicações, podendo utilizar os próprios *scripts* para aprender sobre a própria tecnologia. Por exemplo, pode-se utilizar o mesmo padrão de *scripts* já demonstrados para recuperar informações das *NameSpaces* do CIM, alterando apenas o nome da classe destino. Isto porque todas as informações sobre as *NameSpaces* são armazenadas no repositório CIM como instâncias estática da classe *__NameSpace*, logo, percebe-se que as instâncias de classes estáticas são recuperadas diretamente do CIM, sem uso de provedor WMI. O *script* da Tabela 14 recupera e exibe todas as *NameSpaces* que estão abaixo da *namespace* principal (WMI, 2008, *On-Line*).

Tabela 14: Recuperando NameSpaces CIM usando script WMI (WMI, 2008, *On-Line*).

Linha	Sintaxe
1.	strComputer = "."
2.	
3.	Set objSWbemServices = GetObject("winmgmts:\\" & strComputer & "\root")
4.	Set col*NameSpaces* = objSwbemServices.InstancesOf("*__NAMESPACE*")
5.	
6.	For Each obj*NameSpace* In col*NameSpaces*
7.	Wscript.Echo obj*NameSpace*.Name
	Next

2.7.1.2.3.2. Categorias de Classes

As classes de sistemas suportam a configuração e operações internas do WMI como, por exemplo, as configurações de *NameSpaces*, segurança de *NameSpaces*, registro de provedores, e eventos de subscrições e notificações. As classes de sistemas são identificadas pelas duas sublinhas (*underscores*) que prefaciam cada nome de classe de sistema, conforme demonstrado na figura 17. Por exemplo, as classes *__SystemClass, __Provider,* e *__WIn32Provider* e *__NameSpace* são exemplos de classes de sistema, cujas classes dificilmente farão parte da construção de *script* WMI, a não ser com o objetivo de monitoração (WMI, 2008, *On-Line*).

Classes comuns e de núcleo servem para duas funções: primeiro, elas representam classes abstratas das quais sistemas e desenvolvedores de softwares podem derivar e criar classes de extensões específicas. Segundo, elas definem recursos comuns para áreas de gerenciamento particular.

As extensões de classes são tecnologias de classes específicas criadas pelo sistema e desenvolvedores de softwares aplicativos. As classes *Win32_BaseService, Win32_Service, Win32_SystemServices* e *Win32_ComputerSystem,* mostradas na figura 17, são extensões de classes. A maioria das classes de extensão Microsoft da *NameSpace root\cimv2* podem ser identificada através do prefixo *Win32_*, (WMI, 2008, *On-Line*).

Assim como é possível listar todas as *NameSpaces* através de *scripts* WMI, também é possível listar todas as classes armazenadas dentro de uma *NameSpace*. O *script* da Tabela 15 exemplifica como listar todas as classes definidas na *NameSpace root\cimv2*. Observa que esse *script* faz uso do método *SubclasseOf*, o qual tem como retorno todas as classes filhas da classe pai especificada.

Tabela 15: Recuperando as classes definidas na NameSpaces root\cimv2 (WMI, 2008, *On-Line*).

Linha	Sintaxe
1.	strComputer = "."
2.	
3.	Set objSWbemServices = GetObject("winmgmts:\\" & strComputer & "\root\cimv2")
4.	Set colClasses = objSwbemServices.SubClassesOf( )
5.	
6.	For Each objClass In colClasses
7.	Wscript.Echo objClass.Path.Path
	Next

2.7.1.2.3.3. Tipos de Classes CIM

De forma semelhante ao *Active Directory*, as classes CIM são organizadas de forma hierárquica, classes filhas herdam das classes topo ou pai propriedades, métodos, e qualificadores. O padrão DTMF mantém a base comum de classes, das quais sistemas e aplicações de desenvolvimento derivam e criam sistemas ou aplicações específicas de extensões de classes. Por exemplo, obedecendo a hierarquia de herança, a classe *Win32_Service* tem como classes superiores na ordem ascendente as seguintes classes: *Win32_BaseService, CIM_Service, CIM_LogicalElment, CIM_ManagedSystemElement* das quais herda propriedades, métodos e qualificadores, conforme mostra a figura 17 (WMI, 2008, *On-Line*).

A tabela 16 traz uma comparação das propriedades que não são de sistemas, encontradas em cada uma dessas classes, facilitando a percepção da herança aplicada às classes filhas, bem como as particularidades de cada uma em relação à sua classe pai.

Tabela 16: Comparação das propriedades das classes Pais e Filhas.

CIM_ManagedSystemElement and CIM_LogicalElement	CIM_Service	Win32_BaseService	Win32_Service
Description	Description	Description	Description
InstallDate	InstallDate	InstallDate	InstallDate
Name	Name	Name	Name
Status	Status	Status	Status
	CreationClass	CreationClass	CreationClass
	Name	Name	Name
	Description	Description	Description
	Name	Name	Name
	Started	Started	Started
	StartMode	StartMode	StartMode
	Status	Status	Status
	SystemCreation	SystemCreation	SystemCreation
	ClassName	ClassName	ClassName
	SystemName	SystemName	SystemName
		AcceptPause	AcceptPause
		AcceptStop	AcceptStop
		DesktopInteract	DesktopInteract
		DisplayName	DisplayName
		ErrorControl	ErrorControl
		ExitCode	ExitCode
		PathName	PathName
		ServiceSpecific	ServiceSpecific
		ExitCode	ExitCode
		ServiceType	ServiceType
		StartName	StartName
		State	State
		TagID	TagID
			Checkpoint
			ProcessID
			WaitHint

Como visto anteriormente, os tipos de classes CIM se resumem em classes abstratas, estáticas e Classes dinâmicas.

- As classes abstratas CIM são utilizadas como modelo para definir novas classes e servem como base para outras classes abstratas, estáticas e dinâmicas. Este tipo de classe pode ser identificado examinando o qualificador abstrato. Uma classe abstrata deve definir um qualificador abstrato e configurar o qualificador abstrato com valor "true". O uso mais comum de tipo de classe abstrata é para definir classes comuns e de núcleos. Classes abstratas são raramente usadas em *scripts* WMI, porque não se podem recuperar instâncias de classes abstratas (WMI, 2008, *On-Line*).
- As classes estáticas são responsáveis por definir os dados que são fisicamente armazenados no repositório CIM. As instâncias de classes estáticas são armazenadas e recuperadas diretamente do repositório CIM, pois não utilizam provedores. Estas classes são identificas examinando o qualificador da classe. Entretanto, diferente dos tipos abstrata e dinâmica que são identificadas pela presença de um qualificador específico, classes estáticas são identificada pela ausência dos qualificadores abstratos e dinâmicos.
- As classes dinâmicas são responsáveis por modelar (projeto) os recursos gerenciados WMI, os quais são dinamicamente recuperados dos provedores e são identificadas examinando o qualificador dinâmico. Uma classe dinâmica deve definir um qualificador dinâmico e configurar o qualificador dinâmico com valor igual a "true". O tipo de classe dinâmica é tipicamente utilizado para definir extensões de classes. Classes dinâmicas é o tipo mais comum de classes usadas em *script* WMI.
- Um quarto tipo de classe, conhecida como classe de associação, são classes que descrevem um relacionamento entre duas classes ou entre recursos gerenciados. Classe de Associação pode ser uma classe abstrata, estática ou dinâmica. A classe *Win32_SystemServices*, mostrada na figura 17, é um exemplo de uma classe de associação dinâmica que descreve o relacionamento entre um computador e o serviços que estão sendo executados nesse computador. Este tipo de classe pode ser identificado examinando o qualificador "associação". Uma classe de associação abstrata, estática ou dinâmica deve definir o qualificador "associação" e configurá-lo com o valor igual "true" (WMI, 2008, *On-Line*).

Como CIM é baseado nos princípios da orientação a objetos, as classes CIM incluem propriedades e métodos. As propriedades descrevem as configurações e o *status*

dos recursos gerenciados WMI, enquanto os métodos são funções executáveis que agem sobre os recursos gerenciados associados com as classes correspondentes.

2.7.1.2.3.4. Componentes de uma classe

Cada hardware e recursos de software que é gerenciável através do WMI são definidos por uma classe. Uma classe é um projeto (ou modelo) para um discreto recurso gerenciado WMI, e todas as instâncias do recurso usam o projeto. Por exemplo, todos os serviços (os quais são instâncias da classe *Win32_Service*) têm as mesmas propriedades. Isto não significa que terão os mesmos valores para as propriedades. Por exemplo, "*Alerter*" tem um nome "*Alerter*", enquanto o serviço WMI tem o nome *Winmgmt*. Entretanto, todos os serviços têm uma propriedade "Nome".

Cada definição de classe de recurso gerenciado consiste de propriedades, métodos, e qualificadores. Esta definição é aplicada em todas as instâncias de uma classe e determina o que pode e não pode ser feito para uma instância. Por exemplo, com serviços pode-se especificar ações que deveriam ser tomadas no caso do serviço falhar (reiniciar o serviço, reiniciar o computador, executar um programa, ou não tomar nenhuma ação). A figura 18 demonstra a estrutura de uma definição de classe de um recurso gerenciado (WMI, 2008, *On-Line*).

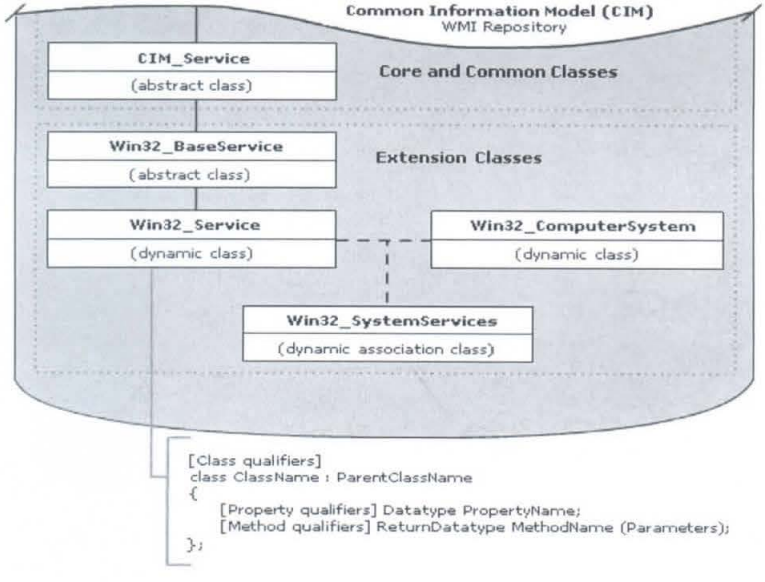

Figura 18: Estrutura de definição de classe de recurso gerenciado (WMI, 2008, On-Line).

Propriedades são como nomes (substantivo) que descreve um recurso gerenciado. Classes usam propriedades para descrever coisas tais como a identidade, configuração e o status do recurso gerenciado. Serviços, por exemplo, tem um nome, nome de exibição, descrição, tipo de inicialização e *status*. A classe *Win32_service* tem as mesmas propriedades. Cada propriedade tem um nome, um tipo e um opcional qualificador (atributo) da propriedade.

Métodos são como verbos que executam uma ação sobre um recurso gerenciado. Por exemplo, para a classe *Win32_Service* deve existir métodos iniciar, parar, pausar e reiniciar serviços. Cada método tem um nome, um tipo de retorno, parâmetros opcionais, e opcionais qualificadores (atributos) de métodos. Assim como propriedades, é possível usar o nome do método em combinação com SWbemObject para chamar o método.

Os qualificadores (atributos) são como adjetivos que provêem informações adicionais sobre a classe, propriedade, ou método para as quais são aplicadas. Os qualificadores definem as características operacionais da propriedade que está sendo atualizada ou do método que está sendo chamando. Há três tipos de qualificadores, os quais provêem três tipos de informações (WMI, 2008, *On-Line*):

- Qualificadores de classes: provêem informações operacionais sobre a classe:
  - qualificadores abstratos, dinâmicos, e de associação indica o tipo da classe;
  - o qualificador do provedor indica o provedor que serve a classe. Por exemplo, o qualificador do provedor para a classe *Win32_Service* indica que a classe utiliza o provedor *CIMWin32 (cimwin32.dll)*; e
  - o qualificador *EnumPrivileges* informa o privilégio especial requerido para usar uma classe.
- Qualificadores de propriedades provêem informações sobre cada propriedade:
  - o qualificador *CIMType* diz o tipo de dados da propriedade. WMI suporta um número de tipos de dados, incluindo *Strings*, inteiros, data e hora, *arrays*(vetores), e valores *Booleanos* (*boolean*);
  - o qualificador "*Read*" indica se a propriedade é legível;
  - o qualificador "*Write*" indica se pode modificar o valor da propriedade; e
  - o qualificador *Key* indica que a propriedade é a chave da classe (chave primária) e é usada para identificar a instância única do recurso gerenciado na coleção de recursos idênticos.

- Qualificadores de Métodos provêem informações sobre cada método. Por exemplo:
  - o qualificador "*implemented*" indica que o método tem uma implementação suprida por um provedor. Isto é importante porque classes WMI frequentemente incluem métodos que são válidos, mas, não realizam nenhuma tarefa porque não foram implementados ainda, estão vazios;
  - o qualificador *ValueMap* define a configuração de valores permitido para o parâmetro de um método ou tipo de retorno; e
  - o qualificador *Privileges* informa privilégios especiais requeridos para chamar o método.

Propriedades e métodos são importantes porque as versões do WMI diferem entre sistemas operacionais; a classe *Win32_ComputerSystem* no Windows XP tem várias novas propriedades e métodos não suportados na classe *Win32_ComputerSystem* do Windows 2000. Portanto, é interessante ter conhecimento sobre esses detalhes porque, diferente das interfaces de serviço do *Active Directory* (ADSI), as propriedades e métodos WMI devem estar disponíveis no computador destino em ordem para ser possível um *script* realizar alguma atividade.

Complementando, ainda, é importante saber o local de armazenamento do CIM, visto que o local varia de acordo com a versão do Sistema Operacional Windows. Para facilitar a localização, a Tabela 17 mostra o diretório onde fica armazenado o CIM nos diferentes Sistemas Operacionais Windows (WMI, 2008, *On-Line*).

Tabela 17: Local de armazenamento do CIM

Sistema Operacional – Versão	Diretório
Windows ME, 98 e 95 OSR 2.5	%windir%\System\Wbem\Repository\cim.rep
Windows 2000 e NT 4.0 SP4	Systemroot\System32\Wbem\Repository\cim.rep
Windows XP	Systemroot\System32\Wbem\Repository\FS

O Modelo de Informações Comuns (CIM) consiste basicamente de quatro arquivos, a saber:

- *index.btr*: árvore binária de índice de arquivos;
- *index.map*: arquivo de controle de transações;
- *objects.data*: repositório CIM onde são armazenadas definições dos recursos gerenciados; e
- *objects.map*: arquivo de controle de transações.

2.7.1.3. Clientes WMI

Conforme demonstrado na arquitetura do WMI, figura 16, os clientes formam a última camada na infra-estrutura do WMI. Quanto à identificação o cliente poder ser (WMI, 2008, *On-Line*):

- um *script;*
- uma aplicação de gerenciamento;
- uma aplicação baseada na Web;
- alguma ferramenta administrativa que acessa e controla informações de gerenciamento disponíveis através da infra-estrutura do WMI.

Os *scripts* das Tabelas 09, 10 e 11, os quais recuperam o tamanho da memória física, os serviços do Windows e os *logs* de eventos do Windows respectivamente, são exemplos de *script* cliente WMI.

2.7.1.4. Segurança em WMI

WMI é uma tecnologia extremamente poderosa e versátil para auxiliar os trabalhos de administração de sistemas. A aplicação dos recursos providos pelo WMI torna-se bastante simples com a escrita de *scripts* usando nada mais que o *Notepad*, cujos *scripts* podem ser facilmente executados tanto em computadores locais como remotos. O resultado da execução desses *scripts* pode ser útil para a administração de sistemas ou de redes, como também pode causar prejuízos irreversíveis dependendo da má intenção do construtor do *script* (WMI, 2008, *On-Line*). A ciência é sempre livre e isenta de maldades, o homem utilizador dessa ciência é que pode desvirtuar a sua aplicação para o lado ruim.

Pensando na segurança na aplicação dessa tecnologia, para evitar que *hackers* façam mau uso dos *scripts* para causar prejuízos a terceiros foram impostas algumas restrições na execução dos *scripts*, sem abrir mão da facilidade de construção dos mesmos, ou seja, construir *scripts* continua fácil, mas, a execução somente é permitida se atendido alguns critérios de segurança, os quais evitam ações de *hackers* como também execução de *scritps* por pessoas indevidas . Por exemplo, a execução de *scripts* somente é permitida por um administrador que tenha privilégios específicos.

A segurança de WMI é uma extensão do subsistema de segurança construído dentro do sistema operacional Windows e inclui os seguintes itens (WMI, 2008, *On-Line*):

- nível de segurança baseado no *WMI NameSpace*;

- *Distributed Component Object Model* (DCOM) *security*;
- segurança padrão do sistema operacional Windows.

2.7.1.4.1. WMI *NameSpace*

As permissões WMI são aplicadas a nível de *NameSpace* e por padrão aplicadas em todas classes que fazem parte da *NameSpace*. As permissões são aplicadas somente para a *NameSpace* principal (*root*) e replicada, por herança, para todas as *NameSpaces* filhas.

O nível segurança aplicado através do WMI *NameSpace* faz com que sejam validadas as permissões da conta do usuário com as permissões aplicadas e armazenadas no repositório CIM. Por padrão, para quem faz parte do grupo de administradores é dado o privilégio completo no WMI e no repositório CIM, tanto nos computadores locais como nos computadores remotos, enquanto os demais usuários são concedidos privilégios somente no computador local (WMI, 2008, *On-Line*).

A lista de permissões WMI disponíveis está na tabela 18, as quais são configuradas através da aba segurança do controle WMI acessado através do MMC, *systemroot\system32\Wmimgmt.msc* (WMI, 2008, *On-Line*).

Tabela 18: Permissões WMI NameSpace

Permissão	Descrição	Administrador	Todos
Executar Métodos	O usuário chama o método de um específico *NameSpace*, entretanto, a execução somente se realiza após o provedor checar se o usuário tem privilégio de executar a tarefa a ser realizada.	•	•
Escrita completa (*Full Write*)	Permissão para o usuário crier ou modificar uma *NameSpace*, uma classe de sistema ou uma instância.	•	
Escrita Parcial	Permissão para criar ou modificar classe estática ou alguma instância de classes que não seja de sistema.	•	
Modificar Provedor	Permissão para o usuário escrever classes e instâncias para provedores WMI.	•	•
Habilitar conta	Conceder permissão de leitura para um WMI *NameSpace*. Isto permite que o usuário execute *script* que recupere dados, porém, somente em computador local.	•	•
Habilitar remoto	Permite um usuário acessar um WMI *NameSpace* a partir de um computador remoto. Este direito é concedido somente para os administradores.	•	

Leitura de segurança	Permite o usuário ler, porém não modificar, a descrição de segurança para um WMI *NameSpace*.	•	
Editar Segurança	Permite um usuário modificar a descrição de segurança para um WMI *NameSpace*	•	

2.7.1.4.2. Segurança DCOM

DCOM pode ser definida como a arquitetura subjacente a interação da biblioteca de *script* WMI com o serviço WMI. Esta arquitetura provê um mecanismo de personificação capaz de permitir especificar de qual usuário (ou perfil do usuário) o serviço WMI irá utilizar os privilégios para executar uma tarefa, agindo como se fosse o próprio usuário. Essa personificação permite o serviço WMI agir em nome de um usuário usando suas credenciais. É possível também atribuir o privilégio de delegação, ou seja, dar o direito do serviço WMI permitir o uso das credenciais do usuário também em outros serviços ou em um terceiro computador (WMI, 2008, *On-Line*).

2.7.1.4.3. Padrão de Segurança do Windows

De forma adicional à segurança em WMI, as permissões para realização de tarefas através dos *scripts* estão subordinadas aos critérios de segurança do próprio sistema operacional Windows. Por exemplo, se o usuário executar um *script* para adicionar algum arquivo em uma determinada pasta e caso este mesmo usuário não possuir permissão Windows de escrita nessa pasta, a execução do *script* não terá sucesso. É isto que obrigará o administrador primeiramente a definir as permissões no Windows para, assim, seja possível realizar da tarefa definida no *script* WMI (WMI, 2008, *On-Line*).

2.7.2. Descrição das principais classes WMI para a solução proposta

Foi apresentado nos tópicos anteriores, dentro da estrutura do repositório CIM, as categorias e tipos de classes WMI, bem como os componentes que formam uma classe. Foi apresentado, ainda, o conceito e a estrutura das *namespaces* e como as classes estão organizadas nas mesmas. No entanto, dentre as diversidades e quantidade de classes existentes não faz sentido para o escopo do presente trabalho o enorme esforço de

enumerar e detalhar todas as classes WMI, até porque deixaria o trabalho bastante extenso. Portanto, serão apenas listadas e detalhadas as principais classes correlacionadas com o trabalho.

Para o escopo de auditoria em redes Windows, WMI disponibiliza o provedor "*Event Log*", o qual fornece os seguintes serviços (MSDN, 2008, *On-Line*):

- acesso aos dados do serviço de *logs* de eventos; e
- notificações de ventos.

O provedor de *logs* de eventos (*Event Log*) usa a classe *Win32_NTEventLogFile* para mapear dados de *logs* de eventos para objetos WMI, e usa a classe *Win32_NTLogEvent* para representar os eventos. O provedor "*Event Log*" implementa a interface padrão *IWbemProviderInit*, comum a todos os provedores, disponibilizando os seguintes métodos que fazem parte da interface *IWbemServices*: *CreateInstanceEnumAsync, ExecMethodAsync, GetObjectAsync* e *PutInstanceAsync* (MSDN, 2008, *On-Line*).

Dentro da estrutura do repositório do WMI (CIM) existem os arquivos que formatam os objetos gerenciáveis, denominados de arquivos MOFs. Esses arquivos contêm a definição de classes que descreve as capacidades providas pelos provedores. Por exemplo, o arquivo "*Ntevt.mof* " contém informações do provedor "*Event Log*", as associações e classes registradas. Todas as classes suportadas pelo provedor "*Event Log*" estão armazenadas na *namespace root\cimv2*, as quais serão listadas e detalhadas a seguir por serem importantes para o escopo do presente trabalho. Vale ressaltar que estas seções foram baseadas no tutorial oferecido pela Microsoft e disponibilizadas em (*MSDN, 2008, On-Line*).

2.7.2.1. Classe *Win32_NTEventLogFile*

A classe WMI *Win32_NTEventlogFile* representa um arquivo ou diretório lógico dos eventos do sistema operacional. A definição da classe demonstrada na Tabela 19 é o código simplificado do MOF – Formato do objeto gerenciado, e inclui todas as propriedades herdadas (*MSDN, 2008, On-Line*):

Tabela 19: Definição da classe Win32_NTEventlogFile

Definição da classe *Win32_NTEventlogFile*

```
class Win32_NTEventlogFile : CIM_DataFile
{
        uint32 AccessMask;
        boolean Archive;
        String Caption;
        boolean Compressed;
        String CompressionMethod;
        String CreationClassName;
        datetime CreationDate;
        String CSCreationClassName;
        String CSName;
        String Description;
        String Drive;
        String EightDotThreeFileName;
        boolean Encrypted;
        String EncryptionMethod;
        String Extension;
        String FileName;
        uint64 FileSize;
        String FileType;
        String FSCreationClassName;
        String FSName;
        boolean Hidden;
        datetime InstallDate;
        uint64 InUseCount;
        datetime LastAccessed;
        datetime LastModified;
        String LogfileName;
        String. Manufacturer;
        uint32 MaxFileSize;
        String Name;
        uint32 NumberOfRecords;
        uint32 OverwriteOutDated;
        String OverWritePolicy;
        String Path;
        boolean Readable;
        String Sources[];
        String Status;
        boolean System;
        String Version;
        boolean Writeable;
};
```

A Tabela 20 traz a lista e a descrição dos métodos definidos pela classe *Win32_NTEventlogFile* (MSDN, 2008, *On-Line)*:

Tabela 20: Descrição dos Métodos da classe Win32_NTEventlogFile

Métodos	Descrição
TakeOwnerShip	Método da classe que obtém propriedade do arquivo lógico especificado na propriedade "*Name*".
changeSecurityPermissions	Método que altera a permissão de segurança do arquivo lógico especificado na propriedade "*Name*".
Copy	Método que copia o arquivo ou diretório lógico especificado na propriedade "*Name*" para o local especificado pelo parâmetro "*input*".
Rename	Método que renomeia o arquivo lógico (ou diretório) especificado na propriedade "*Name*".
Delete	Método que deleta o arquivo lógico (ou diretório) especificado na propriedade "*Name*".
Compress	Método que compacta o arquivo lógico (ou diretório) especificado na propriedade "*Name*".
Uncompress	Método que descompacta o arquivo lógico (ou diretório) especificado na propriedade "*Name*".
TakeOwnerShipEx	Método para obter a propriedade do arquivo lógico especificado na propriedade "*Name*".
ChangeSecurityPermissionEx	Método que altera a permissão de segurança do arquivo lógico especificado na propriedade "*Name*".
CopyEx	Método para copia o arquivo lógico (ou diretório) especificado na propriedade "*Name*" para o local especificado pelo parâmetro "*FileName*".
deleteEx	Método que deleta o arquivo lógico (ou diretório) especificado na propriedade "*Name*".
compressEx	Método que usa compactação NTFS para compactar o o arquivo lógico (ou diretório) especificado na propriedade "*Name*" .
uncompressEx	Método que descompacta o arquivo lógico (ou diretório) especificado na propriedade "*Name*" .
GetEffectivePermission	Método que determina se o visitante tem permissões agregadas especificadas não somente pelo argumento "Permission" do arquivo do objeto, mas sobre o arquivo ou diretório residentes no compartilhamento.
ClearEventLog	Limpa o *log* do evento especificado.
backupEventLog	Salva o *log* do evento especificado em um arquivo de backup.

A Tabela 21 traz a lista e a descrição das propriedades definidas pela classe *Win32_NTEventlogFile* (MSDN, 2008, *On-Line)*:

Tabela 21: Descrição das Propriedades da classe Win32_NTEventlogFile

Propriedades	Qualificadores	Descrição
AcessMask	Tipo de dados: *uint32* Tipo de acesso: *Read-only*	Representa os direitos de acesso exigidos para acessar ou executar uma operação

		específica sobre o arquivo de *log* de evento.
Archive	Tipo de dados: *boolean* Tipo de acesso: *Read-only*	Se verdadeiro, o arquivo que contém os eventos do Windows deve ser arquivado.
Caption	Tipo de dados: *String* Tipo de acesso: *Read-only*	Resumo da descrição do objeto.
Compressed	Tipo de dados: *boolean* Tipo de acesso: *Read-only*	Se verdadeiro, o arquivo que contém os eventos do Windows é compactado.
CompressionMethod	Tipo de dados: *String* Tipo de acesso: *Read-only*	Algoritmo ou ferramenta utilizada para compactar o arquivo lógico que contém os eventos Windows.
CreationClassName	Tipo de dados: *String* Tipo de acesso: *Read-only* Qualifiers: Key, Maxlen(256)	Nome da primeira classe concreta que aparece em um corrente de herança usada na criação de uma instância. Quando usada com a outra propriedade "*key*" da classe, esta propriedade permite todas as instâncias dessa classe e subclasses serem unicamente identificadas.
creationDate	Tipo de dados: *datetime* Tipo de acesso: *Read-only*	Data em que o arquivo que contém os eventos Windows foi criado.
CSCreationClassNam e	Tipo de dados: *String* Tipo de acesso: *Read-only*	Classe do sistema do computador.
CSName	Tipo de dados: *String* Tipo de acesso: *Read-only*	Nome do sistema do computador
Description	Tipo de dados: *String* Tipo de acesso: *Read-only*	Descrição do objeto
Drive	Tipo de dados: *String* Tipo de acesso: *Read-only*	Letra do *drive* do arquivo que contem os eventos Windows. Por exemplo: "C:".
EightDotThreeFileNa me	Tipo de dados: *String* Tipo de acesso: *Read-only*	Nome do arquivo DOS compatível para o arquivo que contém os eventos Windows. Por exemplo: "C:\progra~1"
Encrypted	Tipo de dados: *boolean* Tipo de acesso: *Read-only*	Arquivo que contém os eventos Windows está criptografado.
encryptionMethod	Tipo de dados: *String* Tipo de acesso: *Read-only*	Algoritmo ou ferramenta usada para criptografar o arquivo lógico.
Extension	Tipo de dados: *String* Tipo de acesso: *Read-only*	Extensão do arquivo (sem o ponto) para o arquivo que contém os eventos do Windows. Por exemplo: "txt", "mof".
fileName	Tipo de dados: *String* Tipo de acesso: *Read-only*	Nome do arquivo (sem extensão) definido para o arquivo que contém os eventos do Windows. Por exemplo: *"autoexec"*.
fileSize	Tipo de dados: *uint64* Tipo de acesso: *Read-only*	Tamanho (em *bytes*) do arquivo que contém eventos do Windows.
fileType	Tipo de dados: *String* Tipo de acesso: *Read-only*	Tipo de arquivo (indicado pela propriedade *Extension*).
FSCreationClassNam e	Tipo de dados: *String* Tipo de acesso: *Read-only*	Classe do sistema de arquivo.
FSName	Tipo de dados: *String*	Nome do sistema de arquivo.

	Tipo de acesso: *Read-only*	
Hidden	Tipo de dados: *boolean* Tipo de acesso: *Read-only*	Se verdadeiro, o arquivo que contém os eventos do Windows está oculto.
InstallDate	Tipo de dados: *datetime* Tipo de acesso: *Read-only*	Objeto está instalado. Esta propriedade não necessita de um valor para indicar que o objeto está instalado.
InUseCount	Tipo de dados: *uint64* Tipo de acesso: *Read-only*	Número de arquivos abertos que estão atualmente ativos em relação ao arquivo que contém os eventos Windows.
LastAccessed	Tipo de dados: *datetime* Tipo de acesso: *Read-only*	Data e hora que o arquivo que contém eventos Windows foi acessado pela última vez.
LasModified	Tipo de dados: *datetime* Tipo de acesso: *Read-only*	Data e hora que o arquivo que contém eventos Windows foi modificado pela última vez.
LogfileName	Tipo de dados: *String* Tipo de acesso: *Read-only*	Nome do arquivo que contém eventos Windows. O padrão para nomes dos arquivos de log inclui: aplicação, sistema e segurança.
Manufacturer	Tipo de dados: *String* Tipo de acesso: *Read-only*	Elaborador do recurso da versão, se está presente.
MaxFileSize	Tipo de dados: uint32 Tipo de acesso: *Read-only*	Tamanho máximo (em *bytes*) permitido para o arquivo que contém eventos Windows.
Name	Tipo de dados: *String* Tipo de acesso: *Read-only* *Qualifiers: Key*	Nome herdado que serve como uma chave para uma instância do arquivo lógico que contém eventos Windows dentro de um sistema de arquivos. Deve ser informado o caminho completo dos nomes. Por exemplo: *"c:\windows\system\win.ini"*
NumberOfRecords	Tipo de dados: *uint32* Tipo de acesso: *Read-only*	Número de registros no arquivo que contém eventos windows. Esse valor é determinado ao chamar a função Windows *GetNumberOfEventLogRecords*
OverwriteOutDated	Tipo de dados: *uint32* Tipo de acesso: *Read/Write* *Qualifiers: Units (Days)*	Número de dias após os quais um evento pode ter sido sobrescrito.
OverWritePolicy	Tipo de dados: *String* Tipo de acesso: *Read-only*	*Policy* atual que sobrescreve o serviço de *log* de eventos empregados para esse arquivo de *log*. O dado pode ser nunca sobrescrito, ou sobrescrito quando necessário ou quando fora de data(época). Quando o dado está fora de data depende do valor de *OverwriteOutDated*.
Path	Tipo de dados: *String* Tipo de acesso: *Read-only*	O caminho do arquivo que contém o evento do Windows. Por exemplo: *"\windows\system\"*.
Readable	Tipo de dados: *boolean* Tipo de acesso: *Read-only*	Se verdadeiro, o arquivo que contém eventos Windows pode ser lido.

Sources	Tipo de dados: *String array* Tipo de acesso: *Read-only*	Lista de aplicações que registram *logs* dentro do arquivo de *logs*.
Status	Tipo de dados: *String* Tipo de acesso: *Read-only*	Status atual do objeto. Os possíveis valores são: OK, *error, degraded, unknown, pred fail, starting, stopping, service, stressed, nonrecover, no contact, lost comn.*
System	Tipo de dados: *boolean* Tipo de acesso: *Read-only*	Se verdadeiro, o arquivo que contém eventos Windows é um arquivo de sistema.
Writeable	Tipo de dados: *boolean* Tipo de acesso: *Read-only*	Se verdadeiro, o arquivo que contém os eventos Windows pode ser escrito (alterável).

2.7.2.2. Classe *Win32_NTLogEvent*

A classe WMI *Win32_NTLogEvent* pertencente ao provedor "*Event Log*" (*Ntevt.dll*) é utilizada para traduzir as instâncias do *log* de evento do Windows. Para receber *logs* de eventos de segurança a aplicação deve ter o privilégio definido por *SeSecurityPrivilege*, caso não tenha será retornado para aplicação a mensagem de "*Access Denied*".

A definição da classe demonstrada na Tabela 22 é o código simplificado do MOF – Formato do objeto gerenciado (*Ntevt.mof*), e inclui todas as propriedades herdadas (MSDN, 2008, *On-Line*):

Tabela 22: Definição da classe Win32_NTLogEvent

Definição da classe *Win32_NTLogEvent*

```
class Win32_NTLogEvent
{
      uint16 Category;
      String CategoryString;
      String ComputerName;
      uint8 Data[];
      uint16 EventCode;
      uint32 EventIdentifier;
      uint8 EventType;
      String InsertionStrings[];
      String Logfile;
      String Message;
      uint32 RecordNumber;
      String SourceName;
      datetime TimeGenerated;
      datetime TimeWritten;
```

```
        String Type;
        String User;
};
```

A classe *Win32_NTLogEvent* não possui métodos definidos. A Tabela 23 traz a lista e a descrição das propriedades definidas pela classe *Win32_NTLogEvent* (MSDN, 2008, *On-Line)*:

Tabela 23: Descrição das Propriedades da classe Win32_NTLogEvent

Propriedades	Qualificadores	Descrição
Category	Tipo de dados: *uint16* Tipo de acesso: *Read-only*	Subcategoria para os eventos. Essa subcategoria está em local especifico.
CategoryString	Tipo de dados: *String* Tipo de acesso: *Read-only*	Tradução de subcategoria. A tradução está em local especifico.
ComputerName	Tipo de dados: *String* Tipo de acesso: *Read-only*	Nome do computador que gerou o evento.
Data	Tipo de dados: *uint8 array* Tipo de acesso: *Read-only*	Lista de dados binários que acompanham o relatório do evento do Windows.
EventCode	Tipo de dados: *uint16* Tipo de acesso: *Read-only*	Valor 16-bits inferiores da propriedade *EventIdentifier*. Esse valor está presente corresponder com o valor exibido no visualizador de eventos do Windows.
EventIdentifier	Tipo de dados: *uint32* Tipo de acesso: *Read-only*	Identificador de ventos. Esse é específico para a origem que gerou a entrada do *log* de evento e é usado, juntamente com *SourceName*, para identificar de forma única o tipo de evento do Windows.
EventType	Tipo de dados: *uint8* Tipo de acesso: *Read-only*	Os Sistemas Windows server 2003, Windows 2000 e XP possui os seguintes tipos de eventos: Valores: 1 → *Error* - *2* → *Warning* *3* → *Information;* *4* → *Security Audit Success;* *5* → *Security Audit Failure.*
InsertionStrings	Tipo de dados: *String array* Tipo de acesso: *Read-only*	Lista das *Strings* de inserção que acompanham o relatório dos eventos Windows.
Logfile	Tipo de dados: *String* Tipo de acesso: *Read-only* *Qualifiers: Key*	Nome do arquivo de *log* de eventos Windows. Juntamente com *RecordNumber*, é utilizado para identificar de forma única uma instância dessa classe.
Message	Tipo de dados: *String* Tipo de acesso: *Read-only*	Como aparece a mensagem do evento no *log* de ventos do Windows. Esta é uma mensagem padrão com zero ou mais *Strings* de inserção fornecida pela origem

		do evento do Windows. As *Strings* de inserção são inseridas dentro da mensagem padrão em um formato predefinido.
RecordNumber	Tipo de dados: *uint32* Tipo de acesso: *Read-only* *Qualifiers: Key*	Identifica o evento dentro do arquivo de *log* de eventos Windows. Esse identificador é específico para o arquivo de *log* e é usado juntamente com o nome do arquivo de *log* para identificar de forma única a instância dessa classe.
SourceName	Tipo de dados: *String* Tipo de acesso: *Read-only*	Nome da origem (aplicação, serviço, driver ou subsistema) que gerou a entrada do *log*. Este é usado juntamente com o *EventIdentifier* para identificar de forma única o tipo de vento do Windows.
TimeGenerated	Tipo de dados: *datetime* Tipo de acesso: *Read-only*	Data e hora que o evento foi gerado.
TimeWritten	Tipo de dados: *datetime* Tipo de acesso: *Read-only*	Data e hora que o evento foi gravado no arquivo de *log*.
Type	Tipo de dados: *String* Tipo de acesso: *Read-only*	Tipo do evento. Este é uma *String* enumerada. É preferencial usar a propriedade *EventType* do que a propriedade *Type*. Os tipos podem ter os seguintes valores: *1 → Error;* *2 → Warning;* *4 → Information;* *8→ Security Audit Sucess;* *16 → Security Audit Failure.*
User	Tipo de dados: *String* Tipo de acesso: *Read-only*	Nome do usuário logado quando o evento ocorreu. Se o nome do usuário não pode ser determinado, será atribuído o valor *NULL*.

A Tabela 24 mostra o exemplo de utilização da classe *Win32_NTLogEvent* para consultar registros de *logs* de ventos de segurança (MSDN, 2008, *On-Line)*.

Tabela 24: Exemplo de utilização da classe *Win32_NTLogEvent* (TECHNET, 2008, *On-Line*)

Exemplo de utilização da classe *Win32_NTLogEvent*

```
strComputer = "."
Set objWMIService = GetObject("winmgmts:" _
    & "{impersonationLevel=impersonate}!\\" _
    & strComputer & "\root\cimv2")
Set colLoggedEvents = objWMIService.ExecQuery _
    ("Select * from Win32_NTLogEvent " _
        & "Where Logfile = 'Security' AND Eventcode=528")
```

```
For Each objEvent in colLoggedEvents
    Wscript.Echo "Category: " & objEvent.Category & VBNewLine _
    & "Computer Name: " & objEvent.ComputerName & VBNewLine _
    & "Event Code: " & objEvent.EventCode & VBNewLine _
    & "Message: " & objEvent.Message & VBNewLine _
    & "Record Number: " & objEvent.RecordNumber & VBNewLine _
    & "Source Name: " & objEvent.SourceName & VBNewLine _
    & "Time Written: " & objEvent.TimeWritten & VBNewLine _
    & "Event Type: " & objEvent.Type & VBNewLine _
    & "User: " & objEvent.User
Next
```

Percebe-se, através do *script* do exemplo anterior, a facilidade para obter os registros em computadores remotos com alteração do valor da variável *strComputer*, bem como o uso da linguagem de consulta (WQL) para filtrar os dados desejados. Vale ressaltar que as propriedades desta classe serão as mais utilizadas para obtenção das informações registradas nos *logs* dos eventos relacionados à auditoria. A figura 19 apresenta os tipos de registros de eventos relacionados à auditoria.

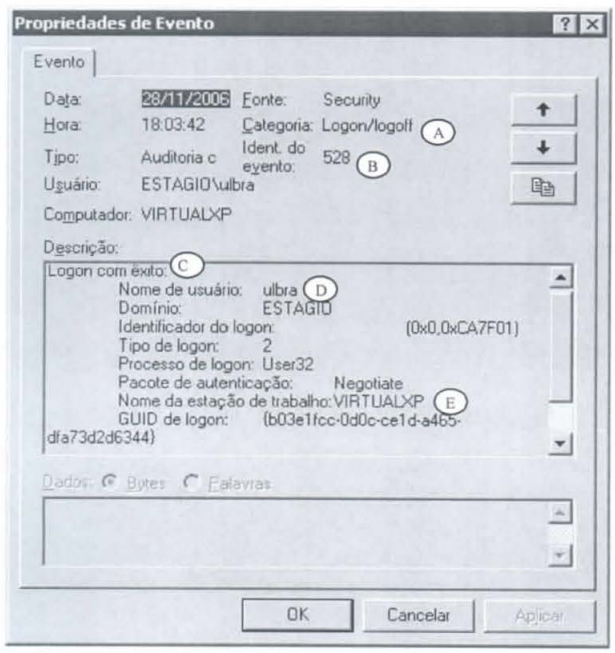

Figura 19: Exemplo de registro de log do evento de logon (JNLC, 2006, p. 92).

Analisando a figura 19 torna-se notável a relação das propriedades listadas na figura com as propriedades da classe WMI *Win32_NTLogEvent*. Desse tipo de registro no arquivo de *logs* de eventos, por exemplo, seria importante para a auditoria o armazenamento das seguintes informações: a categoria do evento em A, o código do evento em B, o tipo do evento em C, o nome do usuário que efetuou *logon* em D e a estação de trabalho acessada pelo usuário em E.

Convém salientar que os procedimentos de configuração das diretivas de segurança para ativar a geração de registros de *logs* de eventos de auditoria, bem como a descrição dos códigos de eventos importantes para auditoria estão documentados no trabalho de conclusão de estágio intitulado "Documentação sobre Diretivas de Auditoria de Segurança em Domínio Windows 2003 Server" (JNLC, 2006).

2.7.2.3. Classe Win32_NTLogEventComputer

A classe WMI *Win32_NTLogEventComputer* faz parte do provedor "*Event Log*" (*Ntevt.dll*) e é definida como uma classe de associação WMI por fazer o relacionamento entre um computador e o um evento.

A definição da classe demonstrada na Tabela 25 é o código simplificado do MOF – Formato do objeto gerenciado (*Ntevt.mof*), e inclui todas as propriedades herdadas (MSDN, 2008, *On-Line*):

Tabela 25: Definição da classe Win32_NTLogEventComputer

Definição da classe *Win32_NTLogEventComputer*

```
class Win32_NTLogEventComputer
{
      Win32_ComputerSystem ref Computer;
      Win32_NTLogEvent ref Record;
};
```

A classe *Win32_NTLogEventComputer* não possui métodos definidos. A Tabela 26 traz a lista e a descrição das propriedades definidas pela classe *Win32_NTLogEventComputer*:

Tabela 26: Descrição das Propriedades da classe Win32_NTLogEventComputer

Propriedades	Qualificadores	Descrição

Computer	Tipo de dados: *Win32_ComputerSystem* Tipo de acesso: *Read-only* *Qualifiers: Key*	Faz referência à instância sobre a qual o evento ocorreu.
Record	Tipo de dados: *Win32_NTLogEvent* Tipo de acesso: *Read-only* *Qualifiers: Key*	Faz referência à instância que representa o evento.

2.7.2.4. Classe Win32_NTLogEventLog

A classe WMI *Win32_NTLogEventLog* faz parte do provedor "*Event Log*" (*Ntevt.dll*) e é definida como uma classe de associação WMI por fazer o relacionamento entre um evento do Windows com um arquivo de log de ventos do Windows.

A definição da classe demonstrada na Tabela 27 é o código simplificado do MOF – Formato do objeto gerenciado (*Ntevt.mof*), e inclui todas as propriedades herdadas (MSDN, 2008, *On-Line*):

Tabela 27: Definição da classe Win32_NTLogEventLog

Definição da classe *Win32_NTLogEventLog*

```
class Win32_NTLogEventLog
{
        Win32_NTEventLogFile ref Log;
        Win32_NTLogEvent ref Record;
};
```

A classe *Win32_NTLogEventLog* também não possui métodos definidos. A Tabela 28 traz a lista e a descrição das propriedades definidas pela classe *Win32_NTLogEventLog*:

Tabela 28: Descrição das Propriedades da classe Win32_NTLogEvenLog

Propriedades	Qualificadores	Descrição
Log	Tipo de dados: *Win32_NTEventLogFile* Tipo de acesso: *Read-only* *Qualifiers: Key*	Faz referência à instância que representa o arquivo de *log* de evento.
Record	Tipo de dados: *Win32_NTLogEvent* Tipo de acesso: *Read-only* *Qualifiers: Key*	Faz referência à instância que representa o evento que está sendo registrado.

2.7.2.5. Classe Win32_NTLogEventUser

A classe de associação WMI *Win32_NTLogEventUser* faz parte do provedor "*Event Log*" (*Ntevt.dll*) e é responsável por fazer o relacionamento entre o evento Windows e o usuário correntemente logado no sistema quando da ocorrência do evento.

A definição da classe demonstrada na Tabela 29 é o código simplificado do MOF – Formato do objeto gerenciado (*Ntevt.mof*), e inclui todas as propriedades herdadas (MSDN, 2008, *On-Line*):

Tabela 29: Definição da classe Win32_NTLogEventUser

Definição da classe *Win32_NTLogEventUser*

```
class Win32_NTLogEventUser
{
        Win32_NTLogEvent ref Record;
        Win32_UserAccount ref User;
};
```

A classe *Win32_NTLogEventUser* também não possui métodos definidos. No entanto, a Tabela 30 mostra a lista e a descrição das propriedades definidas pela classe *Win32_NTLogEventUser*:

Tabela 30: Descrição das Propriedades da classe Win32_NTLogEvenUser

Propriedades	Qualificadores	Descrição
Record	Tipo de dados: *Win32_NTLogEvent* Tipo de acesso: *Read-only* *Qualifiers: Key*	Faz referência à instância que representa o evento que está sendo associado com o usuário.
User	Tipo de dados: *Win32_UserAccount* Tipo de acesso: *Read-only* *Qualifiers: Key*	Faz referência à instância que representa o usuário associado com o evento.

Diversas outras classes WMI importantes para gerenciamento do ambiente Windows podem ser localizadas no próprio site da Microsoft destinados à comunidade técnica como também às equipes de desenvolvimento de aplicações, conhecidas como Microsoft Technet (WMI, 2008) e Microsoft MSDN respectivamente (MSN, 2008).

3. MATERIAIS E MÉTODOS

Nesta seção serão demonstrados os recursos e metodologia empregada para desenvolvimento do presente Trabalho.

3.1. Local e Período

O trabalho foi desenvolvido nas dependências da Secretária de Tecnologia da Informação do Tribunal Regional Eleitoral, em horários extra-expediente, no primeiro semestre do ano de 2008.

3.2. Materiais

Para realização do trabalho foram utilizados materiais que podem ser classificados como: hardware, softwares e fontes de pesquisas bibliográficas.

- **Hardware**
  - Computador Pentium IV, 2.8Ghz com 512MB de memória RAM - IBM
  - Computador Pentium Duo Core2, 1.86 Ghz com 4GB de memória RAM - HP
  - Notebook IBM ThinkPad Lenovo, mod. T60, Processador Intel Centrino Duo, com 1GB de memória RAM.
  - Obs.: Todos equipamentos de propriedade do Tribunal Regional Eleitoral do Tocantins.
- **Softwares**
  - Microsoft Windows XP Professional;
  - Microsoft Windows Server 2003 Professional;
  - Internet Explorer 7.0;

- Acrobat Reader 7.0

- **Fontes Bibliográficas**
  - Livros;
  - Dissertações;
  - Monografias; e
  - Sites com informações técnicas.

3.3. Metodologia

A primeira parte da revisão de literatura foi construída exclusivamente com base em pesquisas sobre gerenciamento de redes, incluindo os modelos de gerenciamento OSI e Internet. Por ser pouco adotado e, consequentemente, pouco explorado, existem poucas fontes de pesquisas que tragam um conteúdo consistente sobre o modelo de gerenciamento OSI. Apesar da literatura encontrada sobre o assunto, principalmente em livros, não ser de fácil assimilação, procurou-se tecer uma ordem textual de forma a facilitar o entendimento do leitor, através de citações, demonstrações em figuras, tabelas e textos explicativos sobre o gerenciamento OSI, bem como sobre as diferenças entre os dois modelos de gerenciamento de redes, forma de funcionamento e protocolos utilizados.

Através das explanações textuais foi possível também fazer uma conexão entre o modelo de gerenciamento OSI e o funcionamento da estrutura interna de gerenciamento Windows, principalmente através do WMI, visto que no modelo internet o gerenciamento ocorre somente em nível de objeto e não em níveis de propriedades de cada objeto.

Para um bom entendimento sobre a segurança e auditoria de redes Windows e a utilização dos recursos disponíveis através do instrumento de gerenciamento Windows, denominado de WMI, foi construída uma documentação procurando dar seqüência lógica ao texto. Foram utilizadas figuras e scripts para exemplificar e facilitar o entendimento do WMI, demonstrando sua aplicação e eficiência na gerência da plataforma Windows, destacando o tratamento das informações contidas nos registros de *logs* de segurança nas estações de trabalho, para uma efetiva aplicação de auditoria em redes Windows, conteúdo este que será muito utilizado no decorrer do desenvolvimento do trabalho.

Para demonstrar a real possibilidade de gerenciamento da segurança de rede Windows através da auditoria foi detalhada a relação entre a visualização dos registros dos

logs com as propriedades da classe WMI, *Win32_NTLogEvent*, a qual possibilita o tratamento das informações, conforme a utilização de exemplo de script capaz de realizar essa tarefa.

Vale informar que todos os exemplos de *scripts* apresentados foram testados no ambiente de TI da Secretaria de Tecnologia da Informação do TRE-TO, os quais podem ser também executados em qualquer outro ambiente windows. A execução desses exemplos ajuda a sedimentar os conceitos apresentados, bem como o aprendizado na construção de *scripts* WMI para fins de gerenciamento.

4. CONSIDERAÇÕES FINAIS

Este trabalho teve como principal objetivo a apresentação de um embasamento teórico que dê sustentação e possibilidade aos administradores de redes criarem soluções de gerenciamento, principalmente, relacionadas à realização de auditoria eficiente em redes windows. O processo de auditoria em redes windows é um recurso importante para o gerenciamento da rede e muito pouco utilizado, principalmente, devido a falta de conhecimento sobre o real significado das propriedades exibidas nos *logs* registrados e pela dificuldade em obter informações úteis para auditoria acessando cada equipamento da rede de forma individualizada. Mapear o perfil do uso da rede e dos usuários é indispensável para manter a rede sob controle gerencial.

Portanto, foi apresentada de forma bem simples a utilização dos recursos já disponíveis na plataforma windows já instalados juntamente com o sistema operacional ou baixados do próprio site da Microsoft, e que tornam viável a construção de soluções aplicáveis ao gerenciamento de redes, em especial, para o processo de auditoria.

A viabilidade de construção dessas soluções foi defendida desde o início do trabalho e comprovada através dos exemplos de *scripts* capazes de tratarem as informações registradas dos *logs* de eventos de auditoria, utilizando para isto a tecnologia WMI.

Contudo, apesar das vantagens frisadas e da facilidade de construções dos *scripts* WMI para administração da auditoria, este trabalho é apenas o ponto de partida para a próxima etapa que se resumirá na construção da ferramenta de apoio ao processo de auditoria em redes windows. O trabalho a ser desenvolvido envolverá a aplicação dos conceitos já apresentados, a utilização da tecnologia WMI para construção da solução de gerenciamento de auditoria, por meio da qual serão desenvolvidos os *scripts* para coletar dos *logs* de segurança as informações específicas para auditoria, que estão registrados nos

computadores da rede. Essas informações coletadas serão armazenadas em uma base de dados através dos recursos disponíveis pelo próprio instrumento de gerenciamento windows (WMI) que possibilitam conexão em base de dados. Uma vez estando com estes registros de *logs* da rede em uma base de dados centralizada, consultas serão construídas utilizando a linguagem SQL para melhor disponibilizar as informações. Facilitando o trabalho dos administradores e tornando mais eficiente o processo de auditoria em redes windows.

Contudo, o presente trabalho abre caminhos para construções de ferramentas com fins diversos do processo de auditoria, como, por exemplo, inventários dos equipamentos, gerenciamento dos recursos disponíveis na rede, instalações e atualizações remotas e muitas outras.

5. REFERÊNCIAS BIBLIOGRÁFICAS

(BATTISTI, 2006) Battisti, Júlio Cesar Fabris. Tutorial de *Active Directory*. 2006. Disponível em http://www.juliobattisti.com.br/artigos/default.asp. Acesso em 05/04/2007.

(DOMÍNIOS, 2004) _____. Configurando a infra-estrutura de domínio em um ambiente *Windows Server* 2003. Atualizado em 06/04/2004. Disponível em http://www.microsoft.com/brasil/security/guidance/prodtech/win2003/secmod118.mspx Acesso em 07/04/2007.

(*DIRECTORY*, 2005) _____. *Active Directory*. Atualizado em 21/01/2005. Disponível em https://www.microsoft.com/technet/prodtechnol/windowsserver2003/pt-br/library/ServerHelp/a9d684f0-90b1-4c67-8dca-7ebf803a003d.mspx?mfr=true. Acesso em 06/04/2007.

(WADLOW, 2000) Wadlow, Thomas A. Segurança de Redes: Projeto e Gerenciamento de Redes Seguras. Rio de Janeiro: Campus, 2000. 268p.

(HORTON, 2003) Horton, Mike; Mugge, Clinton. Hack Notes: Segurança de Redes - referência rápida. Rio de Janeiro: Campus, 2003.250p.

(BRISA, 1993) BRISA. Sociedade Brasileira Para Interconexão de Sistemas Aberto. Gerenciamento de Redes. São Paulo: Makron Books, 1993. 364p.

(AUDITORIA, 2004) _____. Diretiva de Auditoria. Atualizado em 17/05/2004. Disponível em https://www.microsoft.com/brasil/security/guidance/topics/ameaca/secmod50.mspx#EVE. Acesso em 10/04/2007.

(EMBRATEL, 1994) Carvalho, Tereza Cristina Melo de Brito, Organizado e Coordenado por. Arquitetura de Redes de Computadores – OSI e

TCP/IP - BRISA. São Paulo: Makron Books; Rio de Janeiro:EMBRATEL; Brasília, DF: SGA, 1994.

(RAIMIR, 1998) Filho, Raimir Holand. SAGRES: Um Sistema Baseado em Conhecimento para Apoio à Gerência de Falhas em Redes de Computadores. Dissertação (Mestrado em Ciência da Computação) - Universidade Federal do Ceará, 1998.

(FREITAS, 2001) Freitas, Gerado Magela Lopes. Uma Estratégia para Implementação de Gerenciamento de Redes – Estudo de Caso do TCU. Dissertação de Mestrado - Universidade Federal de Santa Catarina, 2001.

(FERNANDEZ, 2007) Fernadez, Marcial Porto. Gerenciamento de Redes. Universidade Estadual do Ceará, 2007. Disponível em http://www.larces.uece.br/~marcial/. Acesso em 15/05/2007.

(WMI, 2008) Microsoft TechNet. Windows Management Instrumentation. Disponível em http://www.microsoft.com/technet/scriptcenter/guide/sas_wmi_overview.mspx?mfr=true. Acesso em 18/05/2008.

(MSDN, 2008) Microsoft Developer Network. Windows Management Instrumentation. Atualizado em 19/06/2008. Disponível em http://msdn.microsoft.com/en-us/library/aa390413(VS.85).aspx. Acesso em 20/06/2008.

(TECHNET, 2008) Microsoft TechNet. Repositório de Script. Disponível em http://www.microsoft.com/technet/scriptcenter/scripts/templates/default.mspx?mfr=true. Acesso em 23/06/2008.

(JNLC, 2006) Carneiro, José Neto Luz. Documentação sobre Diretivas de Auditoria de Segurança em Domínio Windows 2003 Server. Trabalho de Conclusão de Estágio do Curso de Sistema de Informação do CEULP/ULBRA. Disponível em http://www.ulbra-to.br/ensino/monografias/VerMonografia.aspx?im=210. Acesso em 23/06/2008.

(MINASE, 2001) Minase, Mark; Anderson, Christa; Smith Brian; Toombs, Doug. Dominando o Microsoft Windows 2000 Server. São Paulo: Makron Books, 2001. 1275p.