# TRABALHO DE GRADUAÇÃO

# PROPOSTA DE UM SISTEMA DE *BUSINESS INTELLIGENCE* PARA EXPLORAÇÃO DE INDICADORES DE GERÊNCIA DE REDES

**Alberto Gonçalves dos Santos Júnior**
**Catarina Chaves Bernardino**

**Brasília, dezembro de 2009**

**UNIVERSIDADE DE BRASILIA**

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia

TRABALHO DE GRADUAÇÃO

# PROPOSTA DE UM SISTEMA DE *BUSINESS INTELLIGENCE* PARA EXPLORAÇÃO DE INDICADORES DE GERÊNCIA DE REDES

**Alberto Gonçalves dos Santos Júnior**
**Catarina Chaves Bernardino**

**Orientador: Prof. Rafael Timóteo de Sousa Júnior**
**Co-Orientador: Luiz Phillipy M. Sampaio**

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia

TRABALHO DE GRADUAÇÃO

# PROPOSTA DE UM SISTEMA DE *BUSINESS INTELLIGENCE* PARA EXPLORAÇÃO DE INDICADORES DE GERÊNCIA DE REDES

**Alberto Gonçalves dos Santos Júnior**
**Catarina Chaves Bernardino**

Relatório submetido como requisito parcial para obtenção
do grau de Engenheiro de Redes de Comunicação

**Banca Examinadora**

_____
Prof. Rafael Timóteo de Sousa Jr., Doutor, UnB
(Orientador)

_____
Fábio Lúcio Lopes de Mendonça, Mestre, UnB
(Examinador Interno)

# FICHA CATALOGRÁFICA

SANTOS JÚNIOR, ALBERTO GONÇALVES
BERNARDINO, CATARINA CHAVES
Proposta de um Sistema de *Business Intelligence* para Exploração de Indicadores de Gerência de Redes, [Distrito Federal] 2009.
xi, 61 p., 297 mm (ENE/FT/UnB, Engenheiro, Engenharia de Redes de Comunicação, 2009).

Trabalho de Graduação – Universidade de Brasília, Faculdade de Tecnologia. Departamento de Engenharia Elétrica.

1. Redes de Computadores 2. Gerência de Redes
3. Protocolo SNMP 4. *Business Intelligence*

I. ENE/FT/UnB. II. Título (Série)

## REFERÊNCIA BIBLIOGRÁFICA

SANTOS JR., ALBERTO GONÇALVES e BERNARDINO, CATARINA CHAVES (2009). Proposta de um Sistema de *Business Intelligence* para Exploração de Indicadores de Gerência de Redes. Trabalho de Graduação do Curso de Engenharia de Redes de Comunicação. Faculdade de Tecnologia. Universidade de Brasília, DF. 61p.

## CESSÃO DE DIREITOS

NOME DO AUTOR: Alberto Gonçalves dos Santos Júnior e Catarina Chaves Bernardino
TÍTULO DO TRABALHO DE GRADUAÇÃO: Proposta de um Sistema de *Business Intelligence* para Exploração de Indicadores de Gerência de Redes.
GRAU/ANO: Engenheiro de Redes de Comunicação/2009.

*Aos meus pais, Joana e Alberto, e à minha irmã, Karine, por terem guiado meus caminhos com base sólida no amor e na educação, e pelo apoio incondicional que sempre me deram. Ao meu irmão Kaio, que inicia sua jornada na longa estrada do aprendizado e conhecimento.*

*Alberto Júnior*

*Aos meus pais, Ana Cristina e Marco Aurélio, que gostariam de ter presenciado essa conquista, e aos meus pais, Vera Lúcia e Silvio, que nunca mediram esforços para me ajudar e são parte importante desse momento.*

*Catarina Chaves Bernardino*

## Agradecimentos

*Primeiramente agradeço aos céus, único lugar onde foi possível encontrar forças e resignação necessárias para vencer os obstáculos que se apresentaram nessa caminhada. Agradeço eternamente à minha mãe, que sempre me mostrou a importância da educação, do amor e da perseverança. Agradeço ao meu pai, pela firmeza, exemplo e carinho que sempre me deu. À minha irmã, que sempre esteve disposta a me ouvir e me reconfortar nas horas mais difíceis. Me desculpem pela ausência! Amo todos vocês!*
*Agradeço à minha grande amiga e parceira de grupo, Catarina, por ter me tolerado nos momentos de estresse e pelo apoio, fundamental para a conclusão desse trabalho.*
*Agradeço ao prof. Rafael, pela amizade e por tantos conhecimentos compartilhados ao longo de todo esse curso, sempre com bom humor e otimismo. Ao nosso co-orientador Luiz Phillipy, pelo apoio e ajuda que nos deu nessa empreitada.*
*Aos amigos Marcelo, Roberto, Wesley e Wandemberg, sempre dispostos a discutir e trocar idéias, mesmo nas horas menos oportunas! Muito obrigado!*
*Por fim, agradeço aos grandes amigos da segunda turma de 2004, que foram a minha família nos últimos anos, compartilhando todas as dificuldades e alegrias da vida universitária. Vocês são a melhor turma do mundo!*

*Alberto Júnior*

*Agradeço à minha família, que me forneceu os meios necessários para chegar até aqui, me deu apoio, motivação e compreendeu a minha ausência nesses tempos complicados. Obrigada Dad, Veroca, Utha, Beto e Majô, amo muito vocês!*
*Ao meu grande amigo e parceiro de projeto e trabalhos em geral, Alberto Júnior, pelo ótimo trabalho em conjunto, pela paciência e pela cumplicidade, sempre.*
*Aos colegas e amigos Roberto, Marcelo, Wesley e Wandemberg, pelas ajudas e trocas enriquecedoras de idéias.*
*Aos nossos professores, Rafael Timóteo e Luiz Phillipy, pela orientação, pelas boas idéias e por terem acreditado na conclusão desse trabalho.*
*Agradeço, finalmente, a todos os amigos que me deram apoio emocional, vocês também fazem parte disso, mesmo que indiretamente.*
*Muito obrigada!*

*Catarina Chaves Bernardino*

## RESUMO

No decorrer dos últimos anos, as redes de comunicação como um todo vêm assumindo uma tendência cada vez mais forte de crescimento e heterogeneidade. Nesse contexto, se faz necessária a definição de padrões e metodologias para tornar possível o gerenciamento dessas redes. O presente trabalho versa sobre o processo de gerência de redes, apresentando seus fundamentos básicos e o ferramental existente que dá suporte a ela. No decorrer da explanação, serão abordados os conceitos de gerência de redes, o funcionamento dos protocolos que estão envolvidos nesse processo e serão apresentadas algumas ferramentas existentes no mercado que se propõem a realizar tais tarefas. Finalmente, será apresentada uma proposta para uma ferramenta de gerenciamento de redes envolvendo a tecnologia de *Business Intelligence*, explicando seu funcionamento, fundamentos em que se baseia e benefícios obtidos com a implantação de tal proposta.



Palavras-chave: Redes de Computadores; Gerência de Redes; Protocolo SNMP; *Business Intelligence*.



## ABSTRACT

Over the last years, communication networks in general have been showing a strong tendency of growth and heterogeneity. In this context, it is necessary to define standards and methodologies in order to make the management of those networks possible. The present work studies the network management process, presenting its basic foundations and the existing tools that support it. During the explanation, the concepts of network management will be considered, as well as the operation of the protocols involved in the process. Moreover, some network management software solutions available on the market will be presented. Finally, we propose a new tool that involves the Business Intelligence technology, explaining its functioning, foundations on which it is based and the benefits that are obtained by the implantation of such proposal.



Keywords: Computer Networks; Network Management; SNMP Protocol; Business Intelligence.

# SUMÁRIO

# LISTA DE FIGURAS

# LISTA DE TABELAS

# LISTA DE ACRÔNIMOS

| | |
|---|---|
| ASN.1 | Abstract Syntax Notation One |
| BI | Business Intelligence |
| CCITT | Commite' Consultatif International de Telegraphique et Telephonique |
| CMIP | Common Management Information Protocol |
| DHCP | Dynamic Host Configuration Protocol |
| DM | Data Mart |
| DNS | Domain Name System |
| DW | Data Warehouse |
| EGP | Exterior Gateway Protocol |
| ETL | Extract Transform Load |
| HTML | Hypertext Markup Language |
| IAB | Internet Architecture Board |
| ICMP | Internet Control Message Protocol |
| IP | Internet Protocol |
| ISO | International Organization for Standardization |
| ITU | International Telecommunication Union |
| ITU-T | Telecommunication Standardization Sector |
| MIB | Management Information Base |
| MRTG | Multi Router Traffic Grapher |
| NLS | Near Line Store |
| NMS | Network Management Station |
| ODS | Operational Data Store |
| OID | Object Identifier |
| OLAP | Online Analytical Processing |
| OLTP | Online Transaction Processing |
| OSI | Open Systems Interconnection |
| PDU | Protocol Data Unit |
| PHP | PHP Hypertext Processor |
| RDBMS | Relational Database Management System |
| RFC | Request for Comments |
| RRDtool | Round Robin Database tool |
| SGMP | Simple Gateway Management Protocol, |
| SMI | Structure of Management Information |
| SNMP | Simple Network Management Protocol |
| SQL | Structured Query Language |
| TCP | Transmission Control Protocol |
| TI | Tecnologia da Informação |
| UDP | User Datagram Protocol |
| XML | Extensible Markup Language |

# 1 INTRODUÇÃO

As redes de comunicação, dia após dia, vêm crescendo e tomando grande participação no cenário mundial. Tal crescimento vem acompanhado de uma gama de novos softwares e tecnologias, gerando uma grande heterogeneidade nas redes de comunicação. Além disso, o mercado global como um todo se torna cada vez mais dependente da tecnologia da informação (TI), sendo quase impensável sua existência sem o suporte de TI. Nesse cenário, surge então a necessidade do estabelecimento de padrões de gerência de modo a se obter recursos tão estáveis quanto possível. Atualmente, o principal padrão utilizado se baseia no protocolo SNMP, *Simple Network Management Protocol*, que se refere a um conjunto de padrões para gerenciamento [1] composto de um conjunto de objetos gerenciáveis e uma base de dados onde tais objetos são gerenciados. Existem também outros protocolos, como o CMIP, *Common Management Information Protocol*, que não é muito popular, pois é bastante complexo e seu processo de padronização ainda é lento [2].

A maioria das ferramentas de gerenciamento de redes tem como base o protocolo SNMP, utilizando-o na obtenção de dados necessários à gerência. No decorrer do trabalho, serão estudadas algumas dessas ferramentas e será apresentada uma nova abordagem à gerência de redes, sem deixar de utilizar, porém, o SNMP. Essa nova abordagem se baseia nos conceitos de BI, Business Intelligence, uma filosofia que vem se mostrando muito útil em diversas áreas do conhecimento.

## 1.1 MOTIVAÇÃO

As ferramentas de gerência disponíveis atualmente vêm se mostrando cada vez mais deficientes e incapazes de administrar com eficiência redes de grande porte, principalmente redes heterogêneas. Em sua grande maioria, as ferramentas são baseadas em um modelo gerente-agente que requer a instalação de softwares adicionais nas entidades a serem gerenciadas e muitas vezes um grande esforço de configuração. Esse fato por si só já é um entrave quando tratamos de um grande parque de máquinas. Outro problema sério que surge conforme a quantidade de entidades gerenciadas cresce é o *overhead* de comunicação na rede gerado pela interação entre gerente-agente, que pode trazer prejuízos a determinadas aplicações, principalmente as que demandam grande disponibilidade de tráfego.

A saída para esse problema pode ser, de algum modo, tornar o gerenciamento eficiente e transparente para redes de grande e pequeno porte, sem a necessidade de instalação de *softwares* adicionais nas entidades gerenciadas e com o menor *overhead* de comunicação possível.

## 1.2 OBJETIVO GERAL

Com base na motivação vista na seção anterior, o objetivo do trabalho é propor um meio eficiente, simples e transparente de gerenciamento para redes de computadores, sem que a atividade de gerência interfira no desempenho da rede. Para alcançar esse objetivo, utilizaremos os dados provenientes do protocolo SNMP e, a partir deles, levantaremos indicadores que reflitam as condições da rede, permitindo que sejam tomadas as decisões cabíveis no âmbito da gerência de redes. Para dar suporte ao processo de levantamento de indicadores e de tomada de decisões, faremos uso de uma tecnologia que vem se mostrando muito útil e poderosa, aplicável a todos os ramos do conhecimento: o BI, *Business Intelligence*.

## 1.3 ORGANIZAÇÃO DO TRABALHO

O presente trabalho foi organizado em sete capítulos como se segue:

- Capítulo 1: apresenta uma breve introdução ao assunto em questão, a motivação e o objetivo geral do trabalho, assim como sua organização.
- Capítulo 2: contém conceitos importantes à gerência de redes e apresenta um modelo criado pela ITU-T e pela ISO que divide a gerência em cinco áreas funcionais.
- Capítulo 3: explica em detalhes a arquitetura e o funcionamento do protocolo SNMP, assim como os conceitos pertinentes à compreensão do protocolo.
- Capítulo 4: apresenta o estudo de algumas ferramentas de gerência de redes disponíveis atualmente, todas baseadas no protocolo SNMP.
- Capítulo 5: explica a tecnologia de *Business Intelligence*, seu funcionamento, utilidades e processos envolvidos em sua implantação.
- Capítulo 6: mostra os passos seguidos no decorrer do desenvolvimento do sistema proposto, bem como suas características e resultados.
- Capítulo 7: trata de uma conclusão do trabalho, com a análise dos resultados obtidos, de acordo com a proposta do sistema, e também com projetos futuros.

# 2 GERÊNCIA DE REDES

Nos tempos atuais é notória a presença das redes de computadores, nas mais diversas expressões, em nosso dia-a-dia. Nos centros urbanos é praticamente improvável que alguém passe um dia inteiro sem usar uma rede de comunicação, seja realizando uma transação de cartão de crédito ou usando seu aparelho celular. Para garantir o funcionamento de todas as redes que nos servem diariamente é preciso gerenciá-las de maneira adequada e, com base nesse contexto, surge a atividade da Gerência de Redes.

Em um modelo [3] recomendado por órgãos internacionais (ITU-T, ISO/OSI) para gerência de redes, presente na recomendação M.3010, o gerenciamento é subdividido em cinco áreas funcionais: gerência de falhas, gerência de configuração, gerência de desempenho, gerência de contabilidade e, por fim, gerência de segurança. Na literatura, esse modelo é também conhecido por FCAPS (*Fault*, *Configuration*, *Accounting*, *Performance* e *Security*), sigla derivada do inglês para as cinco atividades descritas anteriormente. Apesar da subdivisão, a gerência de redes deve ser entendida como uma atividade única, onde as subdivisões se apóiam mutuamente, não sendo possível que uma desempenhe bem o seu papel sem o auxílio prestado pela outra. Nas seções 2.1 a 2.5 seguem breves descrições de cada uma das áreas funcionais acima.

## 2.1 GERÊNCIA DE FALHAS

A gerência de falhas tem um papel muito importante e crítico no processo de gerência de redes, tratando eventos que venham ou possam vir a comprometer o desempenho e funcionalidades de uma rede de comunicação. Falhas irão acontecer por diversos motivos, algumas vezes até em decorrência de deficiências em outras atividades de gerência. Por exemplo, se há uma deficiência na gerência de desempenho e certo equipamento é sobrecarregado, a chance de haver uma falha certamente aumentará. A gerência de falhas comumente é descrita em três passos: identificação ou detecção da falha, isolamento e, por fim, correção da falha quando possível. É interessante manter registros das ocorrências de falhas e construir uma base de dados, de modo a facilitar um posterior tratamento em caso de uma nova ocorrência, tornando a gerência mais eficiente. Dessa forma, pode-se chegar a um nível tal que será possível tratar uma falha antes que seus impactos sejam percebidos pelos usuários da rede. Nessa ocasião, temos o que se chama de gerência pró-ativa de falhas.

## 2.2 GERÊNCIA DE CONFIGURAÇÃO

A gerência de configuração é uma atividade que tem por objetivo monitorar a configuração de hardware e software dos elementos que compõem a rede, bem como manter registros dessas configurações de modo a tornar possível o acompanhamento e rastreamento de versões de softwares e hardwares, tornando possível gerenciá-los. Em outras palavras, é manter registros dos recursos da rede, sejam eles lógicos ou físicos, de modo a facilitar o gerenciamento dos mesmos. Com a grande heterogeneidade das redes atuais, a gerência de configuração ganha um destaque especial, dando suporte a outras áreas da gerência de redes, como a de desempenho e de falhas. De maneira simplista e resumida, a gerência de configuração tem o papel de gerir um grande registro dos recursos, listando-os em detalhes e mantendo informações que torne possível localizá-los, saber seu estado e, sobretudo, os gerenciar.

## 2.3 GERÊNCIA DE CONTABILIDADE

A gerência de contabilidade é responsável por criar maneiras de se medir o quanto um usuário utiliza certo recurso da rede, qual o impacto dessa utilização para a rede e, por conseqüência, quanto deve ser cobrado pela utilização desse recurso. Pode-se também definir cotas de utilização por usuário, serviços diferenciados para grupos de usuários e assim por diante. Com o apoio da gerência de desempenho e de configuração, a gerência de contabilidade também pode prover informações sobre a subutilização dos recursos e otimizar o compartilhamento dos mesmos na rede.

## 2.4 GERÊNCIA DE DESEMPENHO

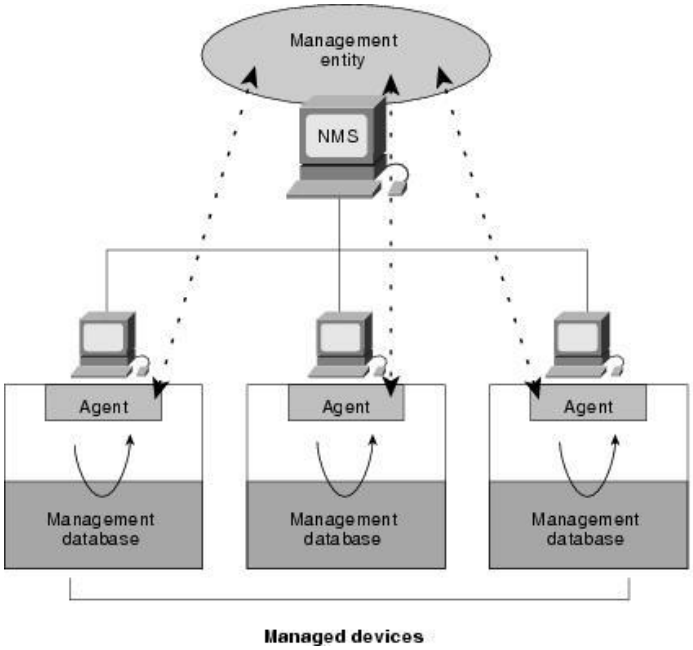
A gerência de desempenho, numa visão idealista, começa antes mesmo da implementação da rede ser feita. Ela começa no planejamento e no dimensionamento da rede, de maneira a atender todos os requisitos do sistema sem que haja falhas, desgaste excessivo dos equipamentos, indisponibilidade de serviços e etc. Isso pode ser feito por meio de simulações da rede com base em parâmetros de desempenho pré-estabelecidos, que podem ser carga de processadores, tempo de resposta, quantidade máxima de acessos suportada e etc. Quando não há um planejamento prévio, a gerência de desempenho pode também se basear em dados estatísticos da própria rede. Com base nessas análises, são definidos limiares aceitáveis de operação, limiares de alerta e limiares de remoção de alerta. Dessa forma, tem-se um modelo de gerência pró-ativa do desempenho da rede.

## 2.5 GERÊNCIA DE SEGURANÇA

A gerência de segurança abrange vários aspectos relacionados à segurança da rede como um todo, que vão desde políticas de acesso aos ambientes nos quais as máquinas se encontram até prevenção de ataques à estrutura lógica da rede. Primeiramente, a gerência de segurança é responsável por estabelecer políticas de acesso aos ambientes onde estão os equipamentos de rede, propondo esquemas de restrição à pessoal não autorizado e, sobretudo, fazendo com que tais políticas sejam cumpridas. É responsável também por determinar políticas de segurança do ponto de vista lógico da rede, como configuração de firewalls, sistemas de detecção de intrusão, sistemas de prevenção de intrusão, antivírus e outros.

# 3 PROTOCOLO SNMP

O SNMP, *Simple Network Management Protocol*, é um protocolo que trabalha na camada de aplicação e que permite que uma máquina servidora, ou gerente, obtenha informações essenciais à gerência de redes por meio de agentes instalados nas máquinas clientes, por assim dizer. A idéia de máquina cliente/servidor, aqui, diz respeito às máquinas nas quais estão instalados, respectivamente, o agente do SNMP, responsável pela coleta de dados, e o servidor SNMP (também chamada de NMS – *Network Management Station*), responsável pela monitoração, gerência e obtenção dos dados dos agentes. A figura abaixo ilustra como se dá o funcionamento básico do protocolo.




Fonte: [4]

Figura 3.1 – Componentes de uma rede gerenciada por SNMP

## 3.1 HISTÓRIA

Derivada do SGMP, *Simple Gateway Management Protocol*, que se limitava à monitoração de *Gateways*, a primeira versão do protocolo SNMP foi definida no RFC 1098, em abril de 1989, que, posteriormente, em maio de 1990, foi atualizado e republicado no RFC 1157. Na versão 1, foram definidas as operações de GET, GETNEXT, SET e TRAP, que serão explicadas em detalhes na seção 3.3.

O SNMPv2, definido no RFC 1901, foi lançado em janeiro de 1996, e tinha como objetivo trazer melhorias à versão anterior. As operações previamente definidas foram mantidas e as operações de GETBULK e INFORM foram adicionadas. A PDU (*Protocol Data Unit*) de TRAP, que, na primeira versão, era diferente das PDUs de GET e SET, foi modificada para ficar em conformidade com as demais PDUs. A segunda versão introduziu, também, a criptografia dos pacotes SNMP, deixando sem criptografia apenas o endereço de destino. Anteriormente, o único mecanismo de segurança eram os nomes de comunidade que, ironicamente, trafegavam sem criptografia juntamente com os pacotes de dados. A versão 2 possui alguns variantes, que diferem nas implementações de segurança do protocolo [5]. Desse modo, as PDUs de todos os variantes são as mesmas, porém as mensagens são diferentes.

A versão mais recente, SNMPv3, trouxe um aumento no nível de segurança e possibilidades de configuração remota do protocolo. Foi lançada em dezembro de 2002 e está contida nos RFCs 3411 e 3412. Foram definidos três níveis de segurança: com autenticação e com privacidade, com autenticação e sem privacidade ou sem autenticação e sem privacidade.

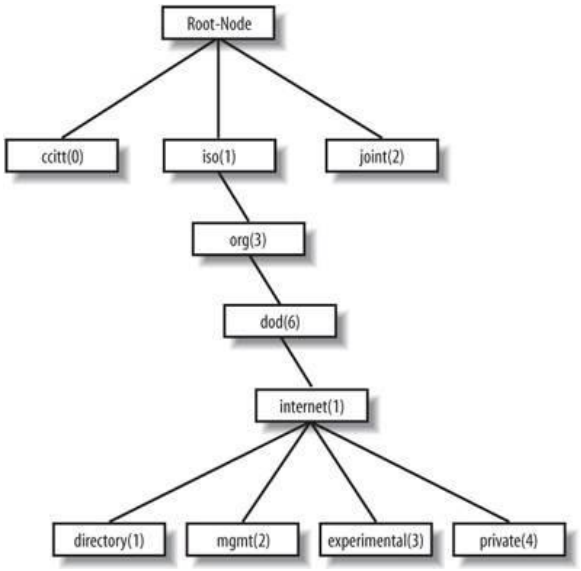
## 3.2 ESTRUTURA DA INFORMAÇÃO DE GERENCIAMENTO

A SMI, *Structure of Management Information*, define como uma MIB, *Management Information Base*, deve ser construída. Desse modo, são definidos os tipos de dados que podem estar contidos na MIB, que são escalares e vetores de duas dimensões compostos de escalares, assim como o modo como os objetos são representados e nomeados. Ao contrário do modelo OSI de gerenciamento, a SMI não permite estruturas de dados complexas, de modo a tornar a implementação mais simples e melhorar a interoperabilidade [2]. O protocolo SNMP, na realidade, se limita a obter escalares.

### 3.2.1 Estrutura da MIB

A MIB é uma base de dados virtual, composta dos objetos gerenciáveis. É importante que os objetos da MIB sejam os mesmos para todos os sistemas a serem gerenciados, caso contrário a interoperabilidade seria prejudicada. A primeira MIB, MIB-I, foi definida no RFC 1066 e baseada na pilha de protocolos TCP/IP. Três anos depois, na RFC 1213, foi definida uma extensão desta, a MIB-II, também baseada em TCP/IP, que contém informações gerais sobre o recurso gerenciado e inclui novos objetos que podem ser gerenciados.

Cada um dos objetos gerenciados possui um OID (*Object Identifier*), e estão organizados hierarquicamente em uma estrutura de árvore. A figura seguinte mostra os principais ramos desta árvore.




Fonte: [6]

Figura 3.2 – Hierarquia em árvore da MIB

Cada nó recebe um número que, posteriormente, servirá para atribuir os OIDs aos objetos gerenciáveis. No primeiro nível, há três nós:

- ccitt(0): administrado pela ITU-T, antiga CCITT.
- iso(1): administrado pela ISO.
- joint-iso-ccitt(2): administrado em conjunto pela ITU-T e pela ISO.

Os nós ccitt e joint-iso-ccitt saem do escopo do presente trabalho e, por isso, não serão tratados. Abaixo do nó iso, há o nó org, que foi criado para o uso de outras organizações. Uma delas é o Departamento de Defesa dos Estados Unidos, representado pelo nó dod, sob o qual está o nó internet. Finalmente, abaixo do nó internet, são definidos quatro nós [2]:

- directory(1): reservado para usos futuros com o diretório da OSI.
- mgmt(2): utilizado para objetos definidos em documentos aprovados pela IAB.
- experimental(3): utilizado para objetos experimentais.
- private(4): utilizado para objetos definidos por organizações privadas.

A MIB é definida sob o nó mgmt. Como a MIB-II é uma extensão da MIB-I, somente uma delas é utilizada por vez, desse modo seus identificadores são os mesmos.

#### 3.2.1.1 Objetos Gerenciáveis

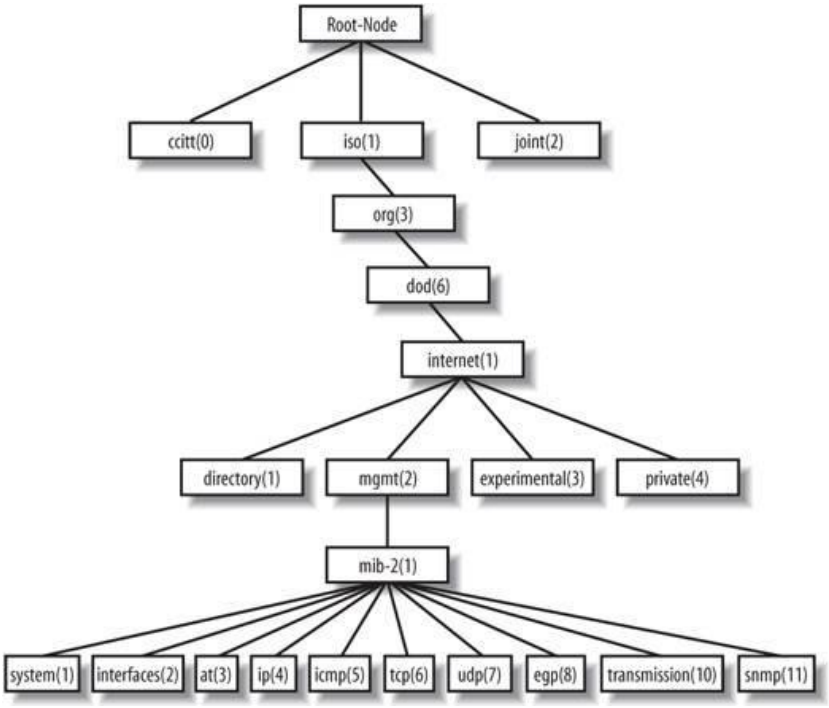
Cada folha da árvore representa um objeto gerenciável da MIB. Tais objetos são recursos ou alguma informação a ser monitorada. A definição dos objetos é padronizada e é feita de acordo com a ASN.1. Apenas um número limitado de tipos de dados da ASN.1, chamados tipos primitivos, é utilizado: inteiros, string de octetos, nulo, identificador de objeto e seqüência. Para se definir um objeto, é necessária a definição dos seguintes parâmetros:

- Objeto: é o nome do objeto, como, por exemplo, o objeto sysUpTime.
- Sintaxe: é o tipo de dado da ASN.1, podendo ser um dos tipos primitivos ou outros definidos a partir dos tipos primitivos, como é o caso de IpAddress, dito tipo de aplicação, criado a partir de strings de octetos. No exemplo de sysUpTime, a sintaxe é TimeTicks.
- Definição: descreve, textualmente, o objeto. A definição de sysUpTime é "O tempo desde que a porção de gerenciamento de rede do sistema foi reinicializada pela última vez." [2].
- Acesso: define o tipo de acesso ao objeto, podendo ser somente leitura, leitura e escrita, somente escrita ou sem acesso. Para sysUpTime, por exemplo, o acesso é somente leitura.
- Status: define se o objeto é obrigatório, opcional ou obsoleto.

Cada objeto possui um identificador único, o OID. Um OID é simplesmente uma seqüência de números separados por pontos, que, além de identificar o objeto, permite localizá-lo na hierarquia previamente apresentada. O objeto utilizado como exemplo, sysUpTime, possui o OID 1.3.6.1.2.1.1.3, referenciando os nós iso(1), org(3), dod(6), internet(1), mgmt(2), mib-2(1), system(1) e, por fim, o próprio objeto, que é o terceiro abaixo do nó system.

#### 3.2.1.2 Grupos da MIB-II

A MIB-II, extensão da MIB-I, é a MIB mais recente e contém alguns objetos e grupos adicionais, sendo mais completa. Os grupos nas quais a MIB-II é subdividida podem ser vistos na figura 3.3.

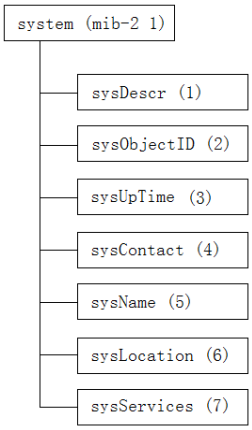



Fonte: [6]

Figura 3.3 – Subdivisões da MIB-II

Os grupos e suas breves descrições serão vistos a seguir. Informações mais detalhadas dos objetos pertencentes aos grupos podem ser vistas em [2].

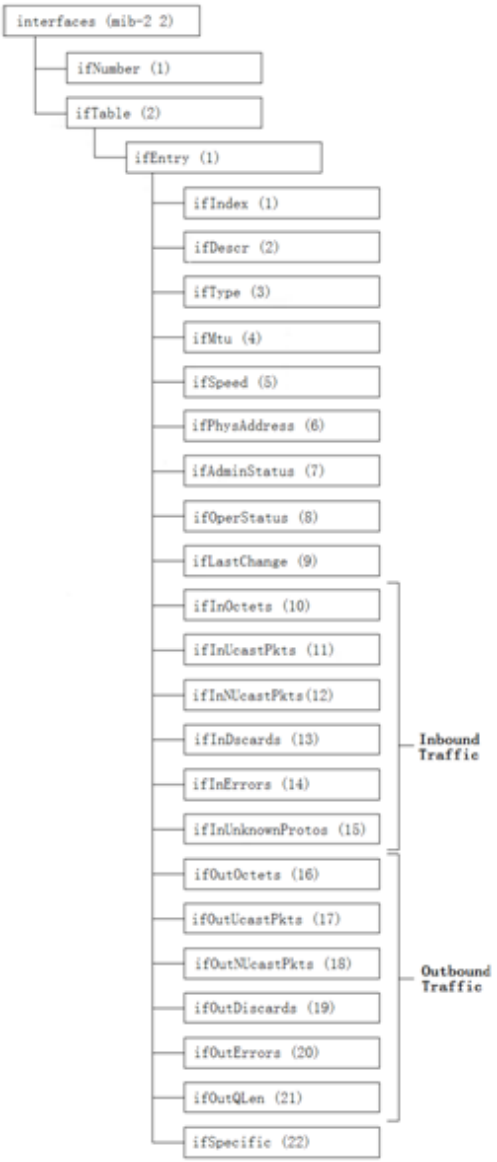
- Grupo system: contém as informações gerais do sistema que está sendo gerenciado.




Fonte: Adaptado de [2]

Figura 3.4 – Grupo system

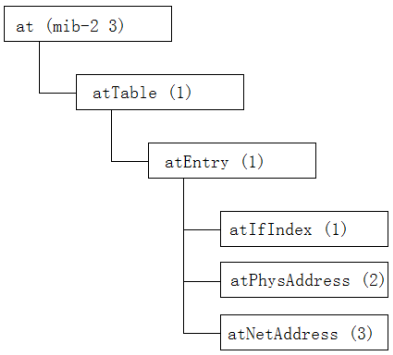
- Grupo interface: contém informações das interfaces físicas do sistema. Um sistema pode ter inúmeras interfaces, por isso há a ifTable, que é uma lista das interfaces e suas informações.




Fonte: Adaptado de [2]

Figura 3.5 – Grupo interface

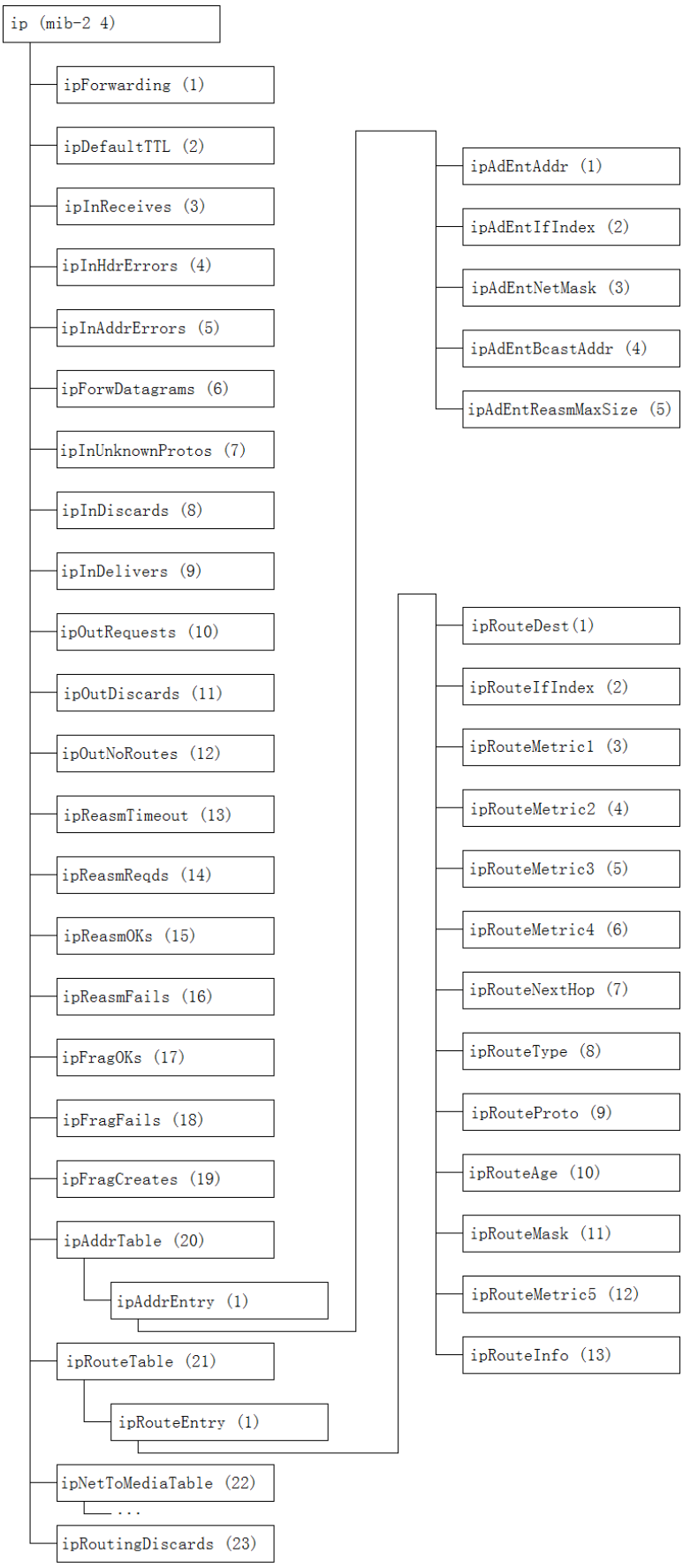
- Grupo at (*Address Translation*): contém informações que tornam possível a tradução de endereços de rede (como, por exemplo, o endereço IP) para endereços físicos (como, por exemplo, endereço MAC). O grupo at está obsoleto na MIB-II, porém não foi removido por questões de compatibilidade com a MIB-I.




Fonte: Adaptado de [2]

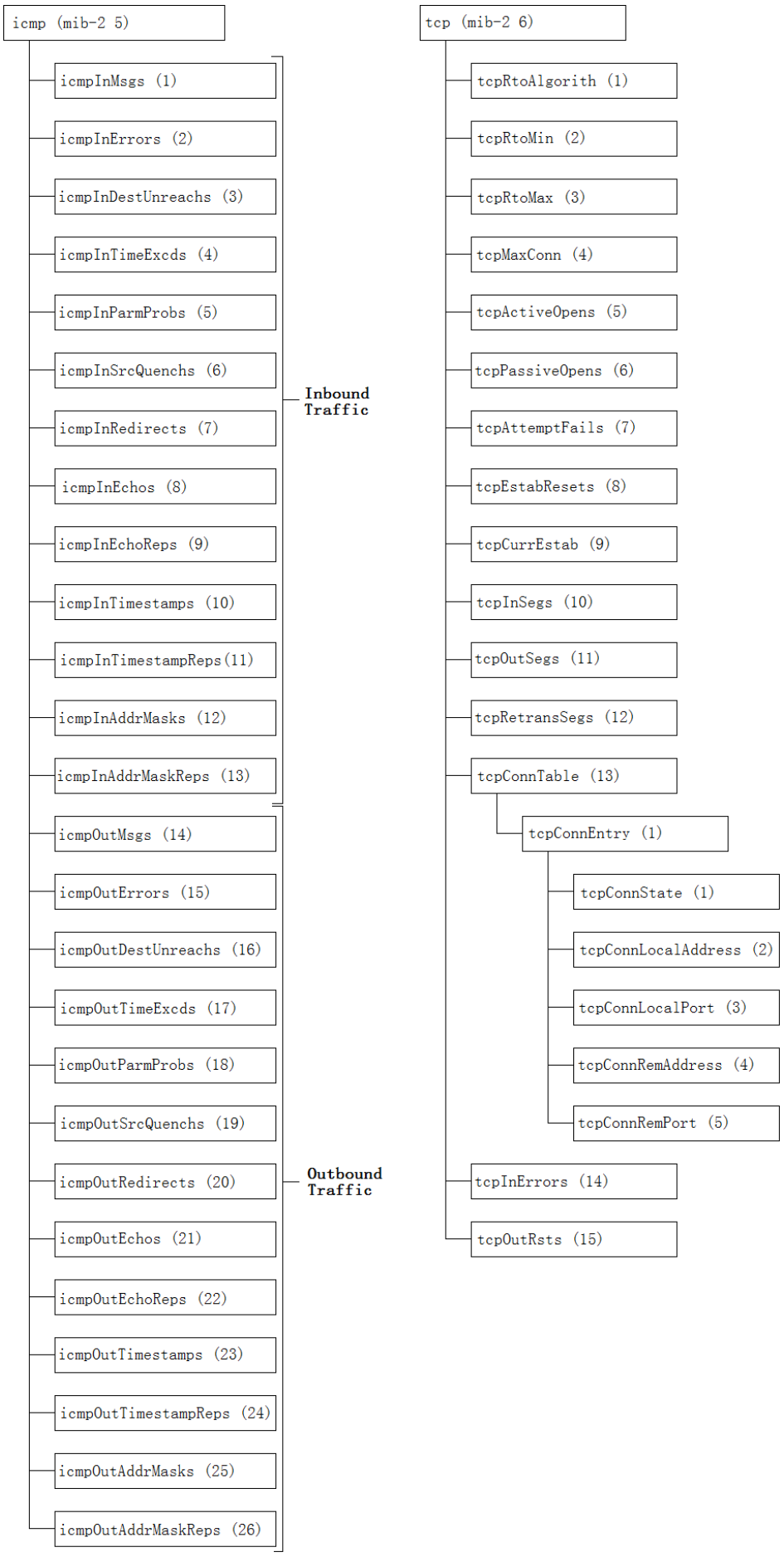
Figura 3.6 – Grupo at

- Grupo ip: contém informações do protocolo IP de dada entidade. Algumas informações do grupo ip podem não ser aplicáveis para certas entidades, como, por exemplo, ipRouteDest, que contém o endereço ip de destino da rota. Tal informação, portanto, pode ser pertinente para roteadores, porém não para hosts. Neste último caso, o objeto possui valor *null*. Alguns objetos do grupo ip estão ilustrados na figura 3.7.
- Grupo icmp: contém informações do protocolo ICMP, que é implementado junto ao protocolo IP. O grupo icmp pode ser visto na figura 3.8.
- Grupo tcp: contém informações do protocolo TCP. O grupo tcp pode ser visto na figura 3.8.
- Grupo udp: contém informações do protocolo UDP. O grupo udp pode ser visto na figura 3.9.
- Grupo egp: contém informações do protocolo EGP. O grupo egp pode ser visto na figura 3.9.
- Grupo *transmission*: contém grupos específicos para cada interface, como é o caso da MIB da interface Ethernet.
- Grupo snmp: contém informações referentes à implementação e operação do protocolo SNMP. Assim como no grupo ip, alguns objetos podem apresentar valor *null* dependendo da implementação do SNMP no sistema, ou seja, se o sistema é um servidor ou apenas agente SNMP. Parte do grupo snmp pode ser visto na figura 3.10.




Fonte: Adaptado de [2]

Figura 3.7 – Parte do grupo ip

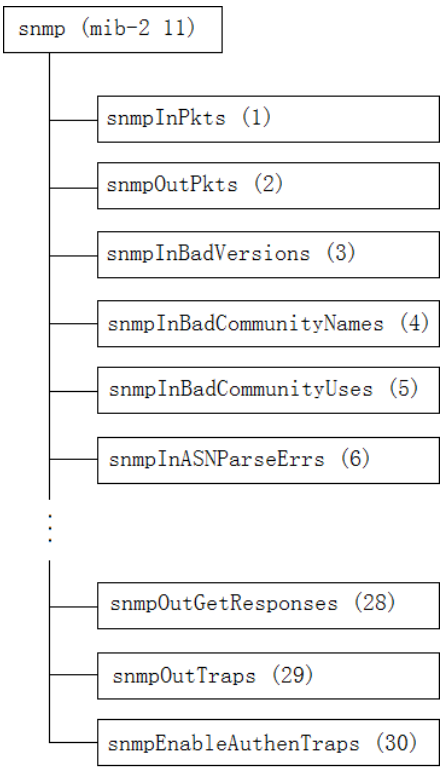



Fonte: Adaptado de [2]

Figura 3.8 – Grupos icmp e tcp


Fonte: Adaptado de [2]

Figura 3.9 – Grupos udp e egp




Fonte: Adaptado de [2]

Figura 3.10 – Parte do grupo snmp

## 3.3 OPERAÇÕES DO SNMP

Existem duas técnicas básicas de operação do protocolo SNMP. A primeira, de *polling*, consiste em uma interação de pergunta e resposta, onde o gerente solicita informações aos agentes e estes respondem. A segunda, de *traps*, consiste no envio de mensagens dos agentes aos gerentes sem a requisição destes. As operações do SNMP se baseiam, portanto, nessas duas técnicas, e estão descritas a seguir:

- GET: obtém o valor de certo objeto, que deve ser identificado por seu nome ou OID.
- GET NEXT: obtém o valor do próximo objeto, sem a necessidade de informar o nome ou o OID. Tal operação é útil caso se deseje percorrer toda ou parte da MIB.
- SET: altera o valor de certo objeto, também identificado pelo nome ou OID. Somente objetos com acesso que permita escrita podem ser alterados.
- TRAP: utilizada pelos agentes para comunicar o gerente da ocorrência de certo evento. É a única operação que não requer resposta. Há sete tipos de *trap* pré-definidos:
    - coldStart(0): se refere a uma reinicialização que pode ter alterado a configuração do agente ou a implementação do protocolo.
    - warmStart(1): se refere a uma reinicialização que não resultou em conseqüências na configuração do agente ou na implementação do protocolo.
    - linkDown(2): se refere a uma falha de comunicação em um dos links do agente.
    - linkUp(3): se refere ao restabelecimento de um *link* de comunicação do agente.
    - authenticationFailure(4): se refere ao recebimento de uma mensagem que não obteve sucesso na autenticação.
    - egpNeighborLoss(5): se refere à perda de conexão com um dos vizinhos EGP.
    - enterpriseSpecific(6): se refere a algum erro específico não-básico.

Além das operações listadas acima, o SNMPv2 adicionou dois tipos de operação: o GETBULK e INFORM. A primeira é utilizada para se obter blocos de dados ao invés de apenas um dado, e a segunda é utilizada para um gerente enviar informações para outro gerente e receber uma resposta.

Um ponto importante a ser ressaltado é que, como a troca de informações do protocolo SNMP é feita utilizando o protocolo UDP, o cliente deve implementar soluções que tratem das perdas de informações, já que o protocolo UDP não garante a entrega das mensagens.

## 3.4 MENSAGENS SNMP

A estrutura das mensagens SNMP pode ser vista na figura seguinte.


Fonte: Adaptado de [2]
Figura 3.11 – Estrutura das mensagens SNMP

A estrutura das mensagens é a mesma tanto para a primeira versão quanto para a segunda versão do protocolo SNMP. Nesta seção, porém, serão abordadas as PDUs do SNMPv2 que, em sua maioria, também são iguais às da primeira versão do SNMP. Somente as PDUs de TRAP foram modificadas, de modo a terem formato igual às demais, e foram adicionadas as PDUs de InformRequest e GetBulkRequest.

Basicamente, a mensagem SNMP é composta por três partes principais:

- Versão: indica a versão do SNMP que está sendo usada.
- Comunidade: o nome da comunidade é usado como uma senha de acesso aos objetos gerenciáveis. Pode haver mais de um nome de comunidade, sendo que cada um possui acesso a certos objetos previamente definidos. No caso de um pedido de um objeto sob o qual a comunidade não possui acesso, é retornada uma mensagem de erro.
- PDU: é a unidade de dados do protocolo. As diferentes PDUs serão detalhadas na próxima seção.

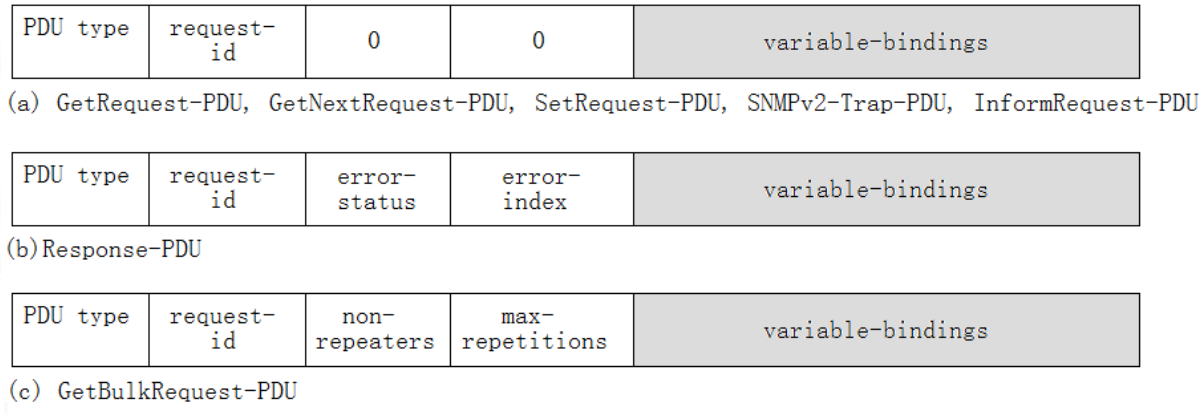
### 3.4.1 *Protocol Data Units*

As diferentes PDUs podem ser vistas na figura 3.12. Nota-se que o formato de todas as PDUs é bastante parecido, porém o significado dos campos pode ser diferente dependendo do tipo da PDU.

Os campos da PDU são definidos da seguinte maneira:

- PDU *type*: identifica o tipo da PDU.
- *request-id*: é um número que identifica a requisição. Na PDU de resposta, este campo terá um valor idêntico ao da PDU de requisição correspondente.
- *error-status*: na PDU de resposta, este campo poderá ser diferente de zero, e apresentará um inteiro que se refere ao código de um certo erro. Caso não ocorra nenhum erro, o campo apresentará valor zero.
- *error-index*: também apresenta valor zero no caso de não haver erros. Caso o campo *error-status* seja diferente de zero, o campo *error-index* indica em qual dos objetos presentes no campo *variable-bindings* o erro ocorreu.

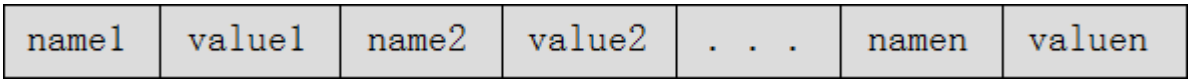
- *non-repeaters*: determina o número de objetos contidos na *variable-bindings* para os quais deve ser feito apenas um GETNEXT, ou seja, para os quais deve ser retornado apenas seu sucessor.
- *max-repetitions*: determina o número de sucessores que serão retornados para o restante dos objetos na *variable-bindings*, ou seja, para o número total de objetos da *variable-bindings* menos o número especificado em *non-repeaters*.
- *variable-bindings*: consiste em uma seqüência de pares, sendo o primeiro elemento um OID e o segundo elemento pode ser um dos seguintes:
    - *value*: o valor do objeto identificado pela OID.
    - *unSpecified*: é utilizado para *Requests*, sendo igual a NULL.
    - *noSuchObject*: retornado no caso do OID não existir.
    - *endOfMibView*: significa que o objeto pedido está além dos limites da MIB do agente responsável pela resposta.




Fonte: Adaptado de [2]

Figura 3.12 – Estrutura das PDUs do SNMP

A estrutura do campo *variable-bindings* pode ser vista na figura 3.13.




Fonte: Adaptado de [2]

Figura 3.13 – Estrutura do campo *variable-bindings*

A padronização das mensagens e das PDUs do protocolo SNMP simplificam consideravelmente as implementações que utilizam o SNMP como base. A grande maioria das ferramentas de gerência utiliza o modelo estudado nesse capítulo como base. Algumas dessas ferramentas serão estudadas no capítulo seguinte.

# 4 FERRAMENTAS DE GERÊNCIA

Algumas ferramentas de gerência que são utilizadas atualmente foram estudadas, principalmente para entender, na prática, como é feita a gerência, revelando, assim, seus problemas e possibilidades de melhor aproveitamento com base no protocolo SNMP. Foram estudados o MRTG, o CACTI e o Nagios, todos softwares livres e de código aberto.

## 4.1 MRTG

O MRTG, *Multi Router Traffic Grapher*, é uma ferramenta gratuita, bastante customizável e de alto desempenho de monitoramento de redes. Possui uma interface Web que contém gráficos diários, semanais, mensais e anuais, que são atualizados a cada cinco minutos, com qualquer informação que possa ser obtida por meio do protocolo SNMP.

A primeira publicação do MRTG data da primavera de 1995, feita por Tobias Oetiker, motivado pela necessidade de se observar o status da rede no campus em que trabalhava à época, na *De Montfort University* em Leicester, Reino Unido. O uso da ferramenta, porém, excedeu as expectativas de seu criador, levando-o a se unir com outros colaboradores para criar uma ferramenta melhorada, mais portável, eficiente e com maior escalabilidade. A primeira versão, limitada ao monitoramento de dez links, evoluiu para uma versão quarenta vezes mais rápida e muito mais amigável ao usuário, lançada em janeiro de 1997. A partir daí, o MRTG ganhou cada vez mais espaço no cenário mundial, tornando-se uma das ferramentas de monitoramento mais utilizadas.

A versão atual do MRTG é escrita principalmente em Perl, sendo as seções críticas escritas em C, tornando-o mais eficiente em questão de tempo de execução. As informações utilizadas são coletadas através do protocolo SNMP. O MRTG possui seu próprio módulo SNMP, fazendo-se desnecessária a utilização de *softwares* externos que implementem o SNMP, o que torna a ferramenta mais portável.

A ferramenta gera, então, gráficos diários, semanais, mensais e anuais das informações coletadas. O intervalo de amostragem é de cinco minutos, dentro dos quais a amostra mais alta é salva e o restante da informação é deletada, diminuindo o consumo de memória. O MRTG utiliza um algoritmo de consolidação de dados que faz com que os arquivos de *log* tenham sempre um tamanho constante. As informações mais antigas constantes no *log* datam de dois anos.

O MRTG é uma ferramenta bastante customizável. É possível modificar a aparência das páginas HTML geradas, assim como a dos gráficos, como cores, legendas dos eixos ou ordem de grandeza das medidas. Há, também, uma ferramenta de criação de índices automática, que cria uma página HTML para cada interface monitorada, bastante útil para organizar as informações quando diversas interfaces são monitoradas.

Uma grande vantagem do MRTG é sua ferramenta de configuração automática, que permite o monitoramento de todas as interfaces de rede da máquina cliente, identificada pelo seu endereço IP, sem a necessidade de configurações manuais. Caso se deseje monitorar outras informações é necessário editar o arquivo de configuração manualmente, informando o OID desejado.

Algumas desvantagens dignas de mencionar são a limitação de só se poder apresentar uma interface por gráfico, não sendo possível a comparação entre duas interfaces diferentes, e a obrigatoriedade de se apresentar duas variáveis em cada gráfico, já que o MRTG foi criado com a intenção de monitorar informações de entrada e saída.

O MRTG é uma das melhores ferramentas para o que ele se propõe a realizar. É bastante simples, podendo ser utilizada por usuários que possuam pouco conhecimento na área de gerência e que desejem, por exemplo, monitorar suas interfaces de rede. Também permite configurações bastante complexas, para usuários mais avançados.

## 4.2 CACTI

O Cacti é uma ferramenta de monitoramento gráfico de redes, baseada numa ferramenta criada por Tobias Oetiker (o mesmo criador do MRTG), o RRDtool, *Round Robin Database tool*,. As informações podem ser coletadas pelo protocolo SNMP e a visualização dos gráficos criados pelo Cacti, assim como a administração do sistema, podem ser feitas por uma interface Web. Os dados são armazenados em um banco de dados MySQL.

O Cacti é um *frontend* para o RRDtool, que é uma ferramenta que permite armazenar, de forma eficiente, diversos tipos de dados, além de os analisar, criando representações gráficas dos dados em função de períodos de tempo definidos. Mesmo que a ferramenta esteja configurada para obter dados em intervalos de um minuto, os dados do último ano, por exemplo, não ocupam muito espaço em disco e podem ser visualizados rapidamente. Isso é possível, devido ao fato de o RRDtool possuir a funcionalidade de consolidar os dados, com opções de qual intervalo de consolidação utilizar e qual função de consolidação utilizar, sendo possível consolidar os dados por meio de médias, máximos, mínimos, total ou último valor [7]. Com a consolidação, o tamanho da base de dados possui um tamanho máximo que não é excedido.

Os dados podem ser obtidos pelo protocolo SNMP, em qualquer uma de suas versões, e por meio de scripts externos. Uma grande vantagem em relação ao MRTG é que a configuração do Cacti é mais simples, podendo ser feita via interface Web ao invés de manualmente no arquivo de configuração. Os gráficos também são bastante customizáveis, e podem conter todo tipo de informação disponível pelo SNMP. É possível organizar os gráficos em árvores definidas pelo administrador, de modo a tornar o monitoramento de um número grande de dispositivos mais fácil e organizado.

Ao se adicionar um novo host, são disponibilizados diversos *templates*, que criam padrões mais inteligentes para aquele host, porém não restringem a configuração de modo algum [8]. Existem, também, *templates* para os dados coletados e para os gráficos. O Cacti, portanto, pode ser utilizado tanto por usuários avançados como por usuários básicos, que podem se basear nos *templates* existentes e, a partir daí, expandir aos poucos o uso das funcionalidades disponíveis.

O Cacti é escrito em PHP e é, por natureza, uma ferramenta ágil e rápida [1]. Assim como as outras ferramentas estudadas neste trabalho, é possível utilizar *plugins* para adicionar novas funcionalidades à ferramenta. Já existem diversos *plugins* disponíveis, criados por usuários colaboradores. A documentação da ferramenta é bastante extensa, além de existir um fórum oficial no qual a maioria dos questionamentos sobre a ferramenta podem ser solucionados.

Em relação ao MRTG, a grande vantagem do Cacti é a configuração mais amigável ao usuário, feita via interface Web, e o uso de *templates* que trazem padrões pré-definidos. Apesar disso, o resultado final de ambos é bastante similar: obtêm-se gráficos para monitorar praticamente qualquer tipo de dados. O MRTG, porém, é mais leve que o Cacti, e exige menos recursos da máquina servidora. A escolha entre uma dessas duas ferramentas depende, portanto, das necessidades do administrador, que deve optar por uma ferramenta mais enxuta, um pouco mais complicada de configurar e mais leve, como o MRTG, ou por uma ferramenta repleta de funcionalidades, com configurações padrão disponíveis, porém que exige mais recursos, como o Cacti.

## 4.3 NAGIOS

Primeiramente conhecido pelo nome de NetSaint, o Nagios é uma das mais antigas e conhecidas ferramentas de gerência de redes. Sua função primordial é a de fornecer informações de forma rápida e concentrada ao administrador da rede. Além disso, possui recursos que possibilitam a configuração de uma série de *thresholds* que podem gerar alertas e enviar mensagens instantaneamente ao administrador, possibilitando uma gerência eficiente da rede.

Ethan Galstad foi o responsável pelo primeiro *release* do Nagios, o NetSaint, e nos dias atuais ainda é quem mantêm o *software*. No decorrer dos anos, várias foram as contribuições feitas, tanto por Galstad quanto pela comunidade, de modo a criar uma ferramenta consistente e robusta na gerência de redes. O *release* atual da ferramenta é o "Nagios Core 3.x".

O Nagios é um *software* baseado no modelo cliente-servidor que, de maneira prática, se traduz em um modelo gerente-agente. Com essa estrutura, monitora "hosts" e serviços com a funcionalidade de gerar alertas, que podem ser enviados por email, mensagens instantâneas e etc., baseados em limiares previamente definidos pelo administrador do sistema. Basicamente, todos os dados que são utilizados e processados pelo Nagios são decorrentes do protocolo SNMP, que é a pedra fundamental de seu funcionamento. O *software* agente, que por assim dizer é instalado na máquina cliente, captura os dados locais e os transmite por intermédio do protocolo UDP ao *software* gerente, que analogamente seria o servidor. Com essa simples estrutura, torna-se, então, possível monitorar um parque relativamente grande de máquinas, de modo a se obter informações preciosas sobre o desempenho da rede, desde que os dados sejam tratados de maneira adequada.

Essas informações são exibidas em um *front-end* baseado em páginas web, o que torna possível o acompanhamento remoto da evolução da rede por seu administrador. A interface web é bem amigável e possui recursos como resumos de desempenho das máquinas, estado de serviços, mapas de rede e mais.

Por ser um *software* livre e de código aberto, o Nagios recebeu várias contribuições ao decorrer dos anos, o que resultou em um software que utiliza diversos plug-ins em seu funcionamento. Essa característica traz consigo muitos benefícios, mas também alguns problemas no que diz respeito à configuração e utilização do software, que pode vir a ser bastante complexa e dispendiosa do ponto de vista de recursos de máquina.

O Nagios é uma boa plataforma para monitoramento e gerência de redes, com *front-end* de fácil utilização. Porém, sua configuração pode se revelar complicada e trabalhosa quando se exige um pouco mais do *software*. Além disso, a funcionalidade de emitir alertas para o administrador é de grande valia em um ambiente que requer alta disponibilidade. Um ponto não favorável em sua utilização nesse projeto é a necessidade de um agente instalado na máquina que se quer monitorar, o que não é estritamente necessário quando é utilizado o protocolo SNMP, já que ele é do tipo "*request-response*", sendo então desnecessário o uso de softwares agentes adicionais.

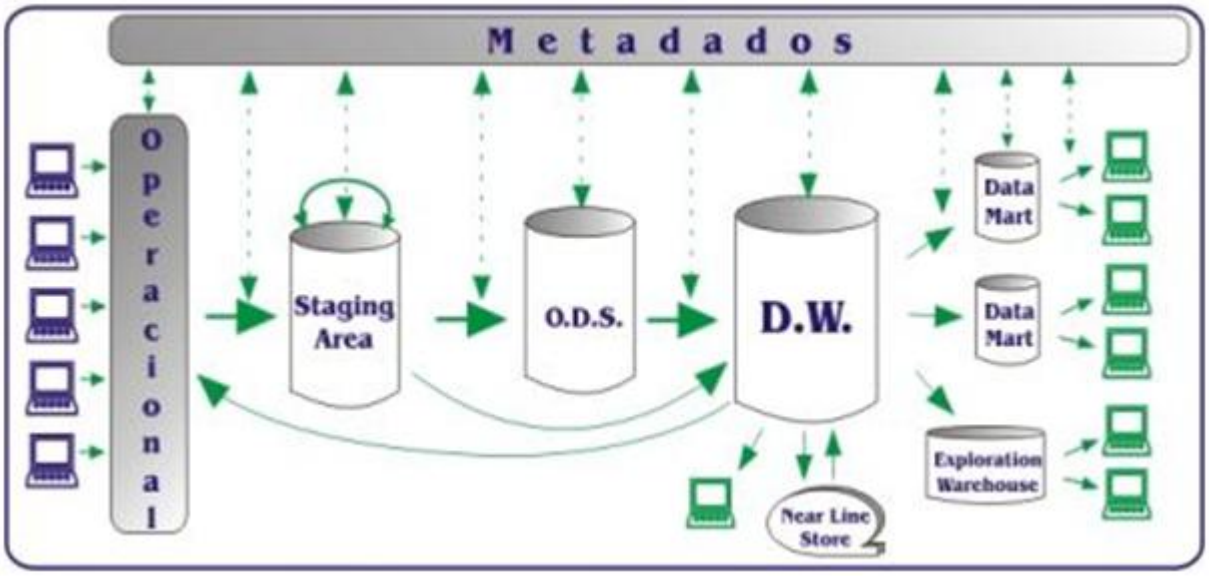
# 5 *BUSINESS INTELLIGENCE*

Uma grande tendência atual, conseqüente da alta competitividade do mercado, é a utilização de BI, *Business Intelligence*, para melhorar o desempenho das organizações. Não há, ainda, a definição de um conceito único para BI, mas pode-se entender BI como "um conjunto de ferramentas e aplicativos que oferece aos tomadores de decisão possibilidade de organizar, analisar, distribuir e agir, ajudando a organização a tomar decisões melhores e mais dinâmicas." [9]. O BI, portanto, é uma ferramenta muito poderosa que pode ser aplicada a diversos tipos de negócios, desde um pequeno comércio a organizações de grande porte, que permite a geração dinâmica de conhecimento a partir da análise de grandes quantidades de dados.

As características de um sistema de BI são [10]:

- Extrair e integrar dados de múltiplas fontes;
- Fazer uso da experiência;
- Analisar dados contextualizados;
- Trabalhar com hipóteses;
- Procurar relações de causa e efeito;
- Transformar os registros obtidos em informação útil para o conhecimento empresarial.

A figura seguinte ilustra a arquitetura de um sistema de BI e suas diferentes etapas.




Fonte: [10]
Figura 5.1 – Arquitetura do BI

Podem-se definir, resumidamente, os componentes da figura como:

- Operacional: Consiste nos bancos de dados ou outras fontes de dados que contém as informações referentes ao negócio.
- *Staging Area*: Neste ponto, as informações da etapa Operacional são tratadas para que sejam carregadas nas bases de dados do software de BI.
- *Operational Data Store* (ODS): É um banco de dados intermediário e volátil, onde as informações tratadas são armazenadas temporariamente para depois serem inseridas nos DWs (*Data Warehouses*). Possui somente informações detalhadas, e também pode ser entendido como "uma visão integrada do mundo operacional". A cada ciclo do BI,

as informações que foram armazenadas anteriormente no ODS são substituídas pelas informações do novo ciclo de transformação.

- *Data Warehouse* (DW): Nesta camada, ficam armazenadas todas as informações após as devidas transformações.
- *Near Line Store* (NLS): É onde ficam armazenadas informações complementares ao DW, que são pouco acessadas.
- *Data Mart* (DM): Os *Data Marts* podem ser vistos como subdivisões dos DW, contendo, por exemplo, informações sobre um determinado setor de uma organização.
- *Exploration Warehouse*: Pode ser entendido como uma base de dados experimental, onde são realizadas análises inovadoras antes que estas sejam implantadas definitivamente no DW.

Nas próximas seções, alguns conceitos importantes ao entendimento do funcionamento dos componentes acima serão estudados em mais detalhes, assim como alguns dos componentes propriamente ditos.

## 5.1 *DATA WAREHOUSES*

Os *Data Warehouses* são bancos de dados onde ficam armazenadas todas as informações históricas necessárias ao processo de BI. As características principais de um DW são:

- Orientação por assunto: diferentes temas são armazenados em uma ou mais tabelas diferentes. Em um exemplo relacionado à área de redes de computadores, é possível armazenar o tráfego da rede em um conjunto de tabelas, o processamento dos hosts em outra, etc. Além disso, cada assunto é armazenado em diversos níveis de detalhamento como, por exemplo, em função do tempo, com registros diários, semanais, mensais e anuais.
- Integração: os dados armazenados devem ser padronizados, ou seja, diferentes dados do mesmo tipo devem ser armazenados da mesma maneira. No exemplo de se desejar armazenar o tráfego, é possível que uma fonte de informação forneça o tráfego em bits ou em kilobits, por exemplo. Antes de serem armazenados, os dados devem ser transformados para que todos estejam na mesma ordem de grandeza e mesma unidade de medida. A padronização não se limita, porém, a unidades de medida. Os dados devem estar totalmente uniformes, o que também se aplica à formatação dos mesmos.
- Variações no tempo: cada entrada em um DW corresponde a um momento específico. Assim, os DW podem ser vistos como um histórico de dados. Caso ocorra uma mudança, ela será inserida no DW como uma nova entrada, e não como alteração de uma entrada anterior. Os DWs, portanto, costumam possuir um grande volume de dados.
- Não volátil: a não volatilidade significa que os dados armazenados no DW não são apagados ou alterados, com a exceção de erros na etapa de armazenamento, que permitem que a inserção errônea seja corrigida.

Essas características permitem que um DW seja utilizado como um repositório de informações para que as análises e a geração de estratégias de competitividade sejam realizadas. O conceito de DW surgiu da necessidade de se integrar diversas bases de dados espalhadas, utilizando somente as informações úteis presentes em cada uma dessas e gerando uma base de dados consolidada. O trabalho de buscar informações em diferentes bases de dados ou até mesmo em diferentes máquinas torna-se muito simplificado com a utilização de *Data Warehouses*.

## 5.2 *DATA MARTS*

Os *Data Marts* possuem, basicamente, as mesmas características dos *Data Warehouses*. A diferença entre os dois é o escopo ou a abrangência das informações contidas em cada um deles. Enquanto nos DWs há informações sobre a organização como um todo, em cada *Data Mart* há informações sobre uma área específica da organização. Essa especificidade dos DM é de extrema importância no que diz respeito a dados que são modificados freqüentemente. Como a quantidade de dados no DM é menor, o desempenho na execução de consultas aos dados é muito melhor que no DW. A tabela 5.1 apresenta um comparativo entre DWs e DMs.

TABELA 5.1
Comparativo entre *Data Marts* e *Data Warehouses*

| ***Data Marts*** | ***Data Warehouses*** |
|---|---|
| Nível departamental | Nível corporativo |
| Alto nível de granularidade | Baixo nível de granularidade |
| Pequena quantidade de dados históricos | Grande quantidade de dados históricos |
| Tecnologia otimizada para acesso de consultas rápidas | Tecnologia otimizada para armazenamento e gerência de grandes quantidades de dados |
| Cada área departamental possui suas características específicas | As estruturas são re-construídas para um entendimento a nível de corporação |

Fonte: [11]

## 5.3 OLTP

Grande parte dos sistemas presentes em empresas e corporações têm, em sua constituição, vários *softwares* que desempenham uma variedade de funções. Tomando como exemplo uma empresa de telecomunicações, tais funções poderiam ser cadastramento de clientes, gerenciamento dos recursos, controle de vendas, controle e cadastro de funcionários, tarifação e etc. Algumas dessas funções são realizadas inúmeras vezes por dia e há a necessidade de que as operações sejam realizadas rapidamente. Por exemplo, em um cadastramento de clientes via telefone, não se pode ter uma grande demora do sistema nas transações, de modo a não causar desconforto ao cliente, o que seria prejudicial para a empresa. Para tornar essas atividades possíveis, é necessário ter uma base de dados com características específicas de modo a atender uma carga de acessos intensa de maneira rápida, mantendo a consistência dos dados.

É nesse contexto que se insere a tecnologia OLTP, *Online Transaction Processing*, dando suporte às atividades de bases de dados que precisam ser acessadas muitas vezes, com alta velocidade de resposta, sem que se perca a consistência dos dados. Ou seja, sistemas OLTP são caracterizados por um grande volume de pequenas transações *online*, na maioria das vezes do tipo INSERT, DELETE e UPDATE.

Para atender a tais características, arquiteturas baseadas em sistemas OLTP precisam ser extremamente normalizadas, com uma grande quantidade de tabelas, sendo voltada para a rapidez na execução de operações no banco de dados. Essa arquitetura é bastante útil na

realização de tarefas diárias operacionais das empresas, porém as informações decorrentes desse processo, da maneira que estão dispostas nas bases de dados OLTP, não auxiliam no processo de tomada de decisões. Para esse propósito, é necessário utilizar uma outra tecnologia, que trabalhe de uma maneira mais analítica nos dados obtidos.

## 5.4 OLAP

Nos dias atuais, praticamente todas as empresas e corporações possuem uma estrutura de TI associada aos seus modelos de negócio, de modo a auxiliar e otimizar seu funcionamento como um todo. Uma parte bastante usual dessa estrutura são as bases de dados, que armazenam informações de interesse para a empresa. Normalmente, essas bases de dados estão associadas a alguma atividade diretamente ligada aos negócios da empresa. Retomando o caso de uma empresa de telecomunicações, poderíamos encontrar diversas bases de dados como, por exemplo, a base de dados referente ao processo de cadastramento de clientes ou a base de dados de tarifação, referente ao consumo dos clientes.

Essas bases de dados são um ponto crítico para o funcionamento da empresa, uma vez que nelas estão dados operacionais que são essenciais ao funcionamento da companhia. Comumente, os sistemas que desempenham essas funções básicas são transacionais, baseados em uma tecnologia OLTP. Embora sejam de tremenda importância do ponto de vista operacional, os dados presentes em bases OLTP, da maneira que estão dispostos, não auxiliam no processo de tomada de decisões.

Nesse contexto se insere a tecnologia OLAP, *Online Analytical Processing*, com o intuito de suprir a necessidade de se obter informações das bases de dados que auxiliem no processo de tomada de decisão. Essa tecnologia permite que seja feita uma análise dinâmica, multidimensional e interativa de uma grande quantidade de dados, normalmente armazenadas em um repositório (que costuma ser preenchido com dados provenientes dos sistemas transacionais), permitindo, assim, que o usuário final tenha uma visão mais elaborada e analítica dos dados como um todo.

Na tecnologia OLAP os dados não precisam ser tão normalizados quanto no OLTP e a base de dados não possui uma grande quantidade de tabelas, porém as tabelas são muito mais extensas. Isso porque em uma base de dados OLAP, a operação que é comumente utilizada é o INSERT, enquanto que UPDATES e DELETES são usados apenas quando acontece algum erro no processo de carga dos dados. Por ter uma construção multidimensional, os dados presentes em bases OLAP são chamados de cubos. As dimensões presentes no cubo são tabelas provenientes de várias fontes distintas, que podem ser referentes a setores do negócio, por exemplo, e a união das dimensões é feita por meio de uma tabela normalmente chamada de "tabela fatos.". Os componentes e operações que acontecem em um cubo serão descritas em breve.

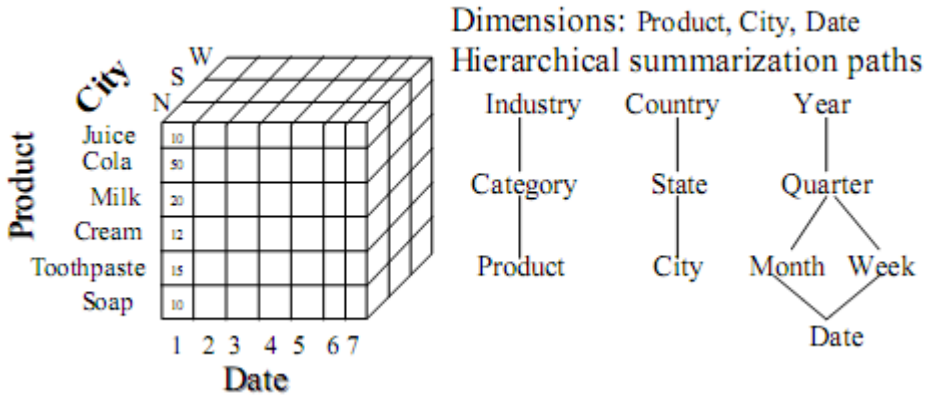
Na próxima página, pode-se ver um quadro comparativo que relaciona as principais diferenças e particularidades das tecnologias OLTP e OLAP.

TABELA 5.2
Comparativo entre tecnologias OLTP e OLAP

| Características | OLTP | OLAP |
|---|---|---|
| **Fonte dos Dados** | Dados operacionais decorrentes de atividades do negócio. | Dados consolidados, geralmente decorrentes de vários OLTPs. |
| **Propósito** | Controlar e executar tarefas fundamentais ao andamento do negócio. | Auxiliar o planejamento, solução de problemas e tomada de decisões. |
| **Constância dos dados** | Dados recentes. | Grandes séries históricas de transações. |
| **Design da Base de Dados** | Altamente normalizada com uma grande quantidade de tabelas. | Normalmente pouco normalizado, com poucas tabelas num esquema estrela ou floco de neve. |
| **Velocidade de Processamento** | Alta velocidade de processamento (baixa quantidade de dados e sem cruzamentos). | Dependente da quantidade de dados e da complexidade dos cruzamentos, podendo levar de minutos a horas. |
| **Visualização dos dados** | Visualização imutável das tabelas. | Altamente adaptável, definida pelo usuário. |

### 5.4.1 Modelagem Multidimensional

A técnica de modelagem dimensional consiste em um projeto lógico que visa apresentar um grande volume de dados de uma maneira intuitiva, simples e rápida para o usuário final. Um exemplo visual de uma base de dados multidimensional é um cubo, conforme ilustra a figura seguinte:




Fonte: [12]
Figura 5.2 – Exemplo visual de uma base de dados multidimensional

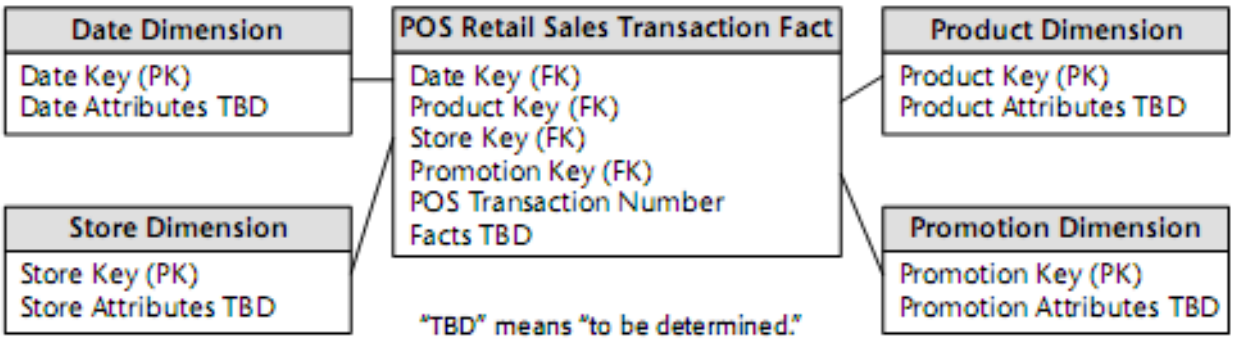
Os principais componentes de um modelo dimensional são os "fatos" e os "atributos".

- Fatos: são dados ou observações inerentes ao andamento do processo em questão, como por exemplo, negócios de uma empresa. O "valor" dos fatos não são conhecidos de antemão, sendo necessário que aconteçam os eventos relacionados a eles para que seja revelado seu "valor". Exemplos de fatos são quantidade de produtos vendidos, número de clientes cadastrados, número de ocorrências de um dado evento. Basicamente os valores atribuídos aos fatos são numéricos.
- Atributos: são características ou descrições de elementos componentes do processo. Podem-se entender os atributos como sendo as características dos fatos. Tomando como exemplo uma empresa do ramo de vendas, teríamos os seguintes exemplos de atributos: nome de produtos, nome de clientes, descrição de algum elemento que compõe o negócio.

As tabelas que formam o modelo dimensional são:

- Tabelas Fato: são as tabelas que guardam os valores dos fatos, além de referências a cada uma das dimensões que compõem o cubo.
- Tabelas Dimensão: são as tabelas que guardam os atributos relacionados aos fatos e podem ser usadas de maneira a proporcionar uma visualização simples e útil dos dados.

A figura abaixo ilustra os conceitos que foram descritos:




Fonte: [13]

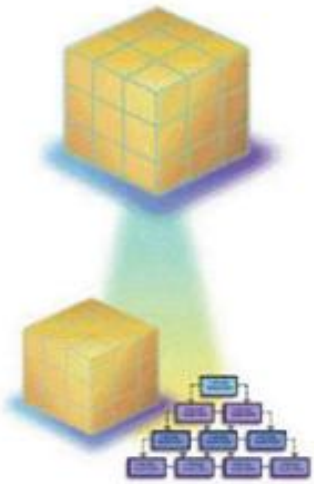
Figura 5.3 – Tabelas dimensão e tabela fatos, com seus atributos e fatos

Na figura 5.3, podemos ver a relação das tabelas dimensão com a tabela fatos e notamos também a maneira como a estrutura está disposta. Esta disposição, em que a tabela fatos está no centro em relação às tabelas dimensão, é chamada de esquema em estrela (*star schema*) e é amplamente utilizado por proporcionar uma resposta rápida nas consultas, apesar de ocupar um grande espaço em disco. Um outro modelo comumente usado é o modelo em floco de neve (*snowflake*), que difere do modelo em estrela por apresentar, nas tabelas dimensão, outras dimensões. A utilidade desse esquema é a construção de hierarquias explícitas que facilitam a compreensão e a organização, além de eliminar parte das redundâncias, melhorando a eficiência em termos de ocupação de disco. Porém, por apresentar mais hierarquias e dimensões que o esquema em estrela, modelos em floco de neve exigem consultas mais complexas, dificultando o cruzamento dos dados e resultando em respostas mais lentas.

Quando se tem uma base de dados construída por meio de um modelo multidimensional, seja ele num esquema estrela ou floco de neve, é possível observar os dados de muitos modos diferentes. De maneira simples, existem operações que permitem ao usuário final "navegar"

nos dados, observando-os por várias dimensões distintas. As operações que possibilitam essa visualização são:

- *Drill Down*: permite ao usuário aumentar a granularidade, ou grau de detalhamento, da informação visualizada.



| Produção | | Celular e Pagers | | |
|---|---|---|---|---|
| | | 1º tri | 2º tri | 3º tri |
| Região Sul | RGS | 78 | 67 | 22 |
| | SC | 90 | 67 | 88 |

| Produção | | Celular e Pagers | | |
|---|---|---|---|---|
| | | Jan | Fev | Mar |
| Região Sul | RGS | 30 | 26 | 22 |
| | SC | 28 | 30 | 32 |

Fonte: [10]
Figura 5.4 – *Drill Down*

- *Drill Up* ou *Roll Up*: é o inverso do *drill down*. Permite que o usuário diminua a granularidade da informação visualizada. Uma ilustração da operação de *drill up* pode ser vista na figura 5.5.
- *Slice and Dice*: permite ao usuário visualizar os dados por várias perspectivas diferentes, modificando a posição da informação com relação às dimensões, alternando linhas, colunas e etc. De maneira simplista, o *slice and dice* permite girar o cubo de informação, proporcionando a visualização que for de maior interesse ao usuário.
    - No *slice*, o cubo é "cortado" em fatias, mantendo a mesma perspectiva de visualização dos dados, porém eliminando fatias que não são interessantes. A figura 5.6 ilustra a operação de *slice*.
    - No *Dice*, há a mudança na perspectiva de visualização dos dados, tornando possível enxergá-los por outros ângulos. A figura 5.7 ilustra a operação de *dice*.

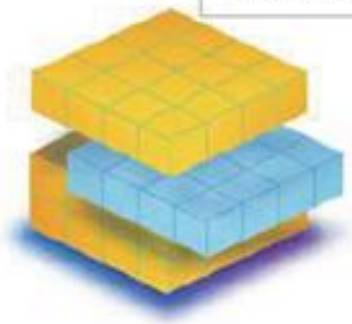
| Produção | | 2004 | | | |
|---|---|---|---|---|---|
| | | Celular | | Pagers | |
| | | Nokia | Siemens | Gradiente | Ericson |
| RGS | Canoas | 13 | 4 | 2 | 5 |
| | Porto Alegre | 20 | 8 | 6 | 7 |

| Produção | | 2004 | | | |
|---|---|---|---|---|---|
| | | Celular | | Pagers | |
| | | Nokia | Siemens | Gradiente | Ericson |
| Região Sul | RGS | 33 | 12 | 8 | 12 |
| | SC | 45 | 34 | 20 | 23 |

Fonte: [10]

Figura 5.5 – *Drill Up*

| Produção | | Celular e Pagers | | |
|---|---|---|---|---|
| | | Jan | Fev | Mar |
| Região Sul | RGS | 30 | 26 | 22 |
| | SC | 28 | 30 | 32 |



| Produção | | Celular | | |
|---|---|---|---|---|
| | | Jan | Fev | Mar |
| Região Sul | RGS | 22 | 18 | 18 |
| | SC | 19 | 27 | 25 |

Fonte: [10]

Figura 5.6 – *Slice*

| Produção | | 2004 | | | |
|---|---|---|---|---|---|
| | | Celular | | Pagers | |
| | | Nokia | Siemens | Gradiente | Ericson |
| RGS | Canoas | 13 | 4 | 2 | 5 |
| | Porto Alegre | 20 | 8 | 6 | 7 |

| Produção | | 2004 | |
|---|---|---|---|
| | | RGS | |
| | | Canoas | Porto Alegre |
| Celular | Nokia | 13 | 20 |
| | Siemens | 4 | 8 |
| Pagers | Gradiente | 2 | 6 |
| | Ericson | 5 | 7 |

Fonte: [10]

Figura 5.7 – *Dice*

## 5.5 ETL

Como foi visto nas seções 5.1 e 5.2, os *Data Warehouses* e *Data Marts* possuem algumas características que requerem que os dados provenientes das diversas fontes sejam modificados antes de serem inseridos nos bancos OLAP. Nesse contexto, surge o processo de ETL, que pode ser enquadrado na *Staging Area* da figura 5.1. ETL é a sigla para *Extract*, *Transform* e *Load*, ou seja, extração, transformação e carga.

A primeira etapa do ETL, de extração, consiste basicamente em obter os dados de fontes que podem ser bancos de dados SQL, planilhas Excel, dentre outros.

A segunda etapa, de transformação, é utilizada para eliminar erros, uniformizar e realizar manipulações, quando necessárias, com os dados. Normalmente, os dados extraídos são bastante numerosos, portanto existe uma grande probabilidade de haver dados com erros resultantes, por exemplo, do mau preenchimento de um formulário ou até mesmo erros lógicos. Uma parte da etapa de transformação, portanto, consiste na limpeza dos dados, eliminando ou corrigindo dados inconsistentes. Feita a limpeza, é necessário transformar os dados, enquadrando-os no padrão definido para os bancos OLAP. Finalmente, os dados podem ser manipulados, por exemplo, para gerar novos dados que serão úteis ao BI.

A última etapa do ETL consiste em carregar os dados transformados nos bancos OLAP. A redundância inerente ao OLAP faz com que cargas em bancos deste tipo sejam muito mais demoradas que cargas em bancos operacionais. Por isso, o período em que será realizada a carga deve ser planejado previamente e depende diretamente do negócio em questão. Normalmente, a carga é realizada em uma pequena janela de tempo na qual o sistema de OLAP pode ficar fora do ar, como, por exemplo, durante a noite. Somente no caso de ser necessária a análise de dados atuais, as cargas serão feitas à medida que os dados são atualizados [12]. Uma maneira de resolver este problema é realizar a carga em paralelo. Cria-se um novo banco de dados para carregar os dados e, somente ao final desse processo, o banco antigo é substituído pelo novo. Desse modo, o banco continua suportando *queries* durante a carga.

O processo de ETL é de extrema importância ao BI, e deve ser planejado cuidadosamente para tratar todas as questões que possam interferir nos resultados finais.

## 5.6 *DATA MINNING*

O *Data Minning*, ou Mineração de Dados, é um processo que consiste em extrair padrões úteis a partir de um conjunto de dados. Há diversos algoritmos utilizados na Mineração de Dados, que podem ser divididos em três tipos principais:

- Algoritmos de classificação: são utilizados para descobrir padrões que tornam possível a classificação dos dados em diferentes classes. Uma aplicação de algoritmos desse tipo pode ser, por exemplo, descobrir se um indivíduo é propenso a fraudar uma empresa de seguro ou não com base em seus dados pessoais. Nesse caso, as classes poderiam ser "fraudador" e "não-fraudador".
- Algoritmos de descobrimento de regras de associação: são utilizados para descobrir regras de causa e efeito, do tipo "Se isso, então aquilo.". Um exemplo clássico desse tipo de padrão se aplica a compras em supermercados, que relaciona a compra de fraldas com a compra de cervejas para consumidores do sexo masculino: Se fraldas, então cerveja.
- Algoritmos de clusterização: são utilizados para agrupar registros com características semelhantes em *clusters*. A clusterização é útil, por exemplo, para o mercado

financeiro, na descoberta de ações que possuem variações semelhantes em função do tempo.

As funcionalidades da Mineração de Dados são, portanto, extremamente importantes na descoberta de padrões, especialmente quando se trata de uma grande quantidade de dados. Uma etapa crucial, anterior à realização da mineração propriamente dita, é o pré-processamento de dados, que consiste em padronizar os dados e escolher que atributos são úteis. No caso das tecnologias de *Business Intelligence*, a mineração é realizada sobre os dados que constituem os *Data Warehouse* ou os *Data Marts*. Como foi visto nas seções 5.1 e 5.2, pela própria característica desses bancos, os dados já estarão uniformizados, restando apenas a seleção dos atributos, quando necessário.

# 6 DESENVOLVIMENTO DO SISTEMA DE BI PARA EXPLORAÇÃO DE INDICADORES DE GERÊNCIA DE REDES

Nesse capítulo serão vistas as ferramentas e tecnologias utilizadas, bem como toda a metodologia e procedimentos que foram adotados no desenvolvimento e implementação do sistema. Serão discutidas as razões de tais escolhas e seus impactos no projeto.

## 6.1 FERRAMENTAS UTILIZADAS

Nesta seção, serão expostas as ferramentas utilizadas na realização do projeto. Apesar da estruturação em tópicos, as tecnologias e ferramentas foram escolhidas como um todo, de maneira tal que a escolha de uma justifica a de outra, e vice-versa. Dessa forma, não faria sentido falar de cada uma de maneira individual, porém, por questões de organização, a exposição será feita dessa maneira.

### 6.1.1 Linux

A primeira escolha a ser feita foi o sistema operacional que seria utilizado. O Linux, mais especificamente a distribuição CentOS, foi escolhido por ser um software livre, o que é muito bom para um ambiente de desenvolvimento, e não traz custos desnecessários.

### 6.1.2 SNMP

O protocolo SNMP é um dos padrões mais utilizados pelas ferramentas de gerência de redes atualmente. Suas características foram exploradas em detalhes no capítulo 3. Neste trabalho, a obtenção dos dados utilizados para a geração de indicadores de gerência foi feita baseando-se no protocolo SNMP. Os principais motivos para essa escolha foram a simplicidade e grande abrangência do SNMP, no sentido de que o protocolo permite que sejam monitorados praticamente todos os dados referentes a redes de computadores, desde dados físicos básicos, como a temperatura de processadores, até dados lógicos, como a quantidade de pacotes retransmitidos.

As extensas MIBs do protocolo SNMP são, portanto, uma fonte muito interessante e confiável para popular os *Data Warehouses* utilizados no *Business Intelligence*.

### 6.1.3 MySQL

Existem muitas opções a serem avaliadas quando se trata da escolha de um sistema de gerenciamento de banco de dados relacional (RDBMS, *Relational Database Management System*), desde sistemas proprietários, como o Oracle ou o Microsoft SQL Server, até sistemas de software livre, como o MySQL ou o PostgreSQL. O custo dos sistemas proprietários, porém, é bastante alto, o que não justificaria sua escolha para a realização deste trabalho.

O MySQL é um dos melhores RDBMS de distribuição livre, pois é bastante estável, dispõe de uma interface gráfica de fácil utilização e possui amplo suporte e documentação da comunidade. Além desses motivos, o fato de a integração do MySQL com os scripts de

captura de dados e com a ferramenta de BI ser mais simples e documentada contribuiu para a escolha do MySQL como RDBMS dos bancos de dados utilizados neste trabalho.

### 6.1.4 Apache

A necessidade de utilizar um servidor web surgiu naturalmente em decorrência do cenário de desenvolvimento em que se situa o projeto. O Apache (HTTPD) foi escolhido por ser amplamente utilizado, estável e por possuir uma facilidade quanto à instalação dos módulos PHP.

### 6.1.3 Pentaho Suite

Como ferramenta de *Business Intelligence*, foi escolhido o Pentaho Suite. Tal escolha foi feita porque o Pentaho é uma ferramenta que a cada dia vem se destacando dentre todas as outras disponíveis no mercado, sejam pagas ou open source. Ele possui módulos referentes a todos os processos do *Business Intelligence*, que vão desde o ETL até a mineração de dados. Além disso, a ferramenta oferece conectividade a inúmeros bancos de dados, o que torna sua adaptação fácil a qualquer tipo de ambiente. Duas edições são disponíveis, a *Community*, que é *open source*, e a *Enterprise*, que é a versão paga. No projeto foi utilizada a versão *Community*.

## 6.2 DESCRIÇÃO DA FONTE DE DADOS

Os dados utilizados no processo de BI foram obtidos da MIB do protocolo SNMP. Pelo fato de haver uma limitação de tempo quanto ao desenvolvimento do projeto, os indicadores de gerência utilizados foram escolhidos com base em [14]. Caso não houvesse limitação de tempo, toda a MIB seria investigada e analisada para se definir uma quantidade maior de indicadores. A figura 6.1 mostra parte dos grupos da MIB com os objetos utilizados em destaque.

Desse modo, foi criado um banco de dados com tabelas responsáveis por armazenar os objetos do protocolo SNMP que seriam utilizados no cálculo dos indicadores. A estrutura desse banco é ilustrada na figura 6.2.

A tabela “maquinas” foi populada manualmente com as informações das máquinas do cenário utilizado para teste, que foram os servidores de E-mail, Web, DHCP e DNS que estão no LabRedes, Laboratório de Redes da UnB. Para popular as demais tabelas, foi utilizado um script PHP configurado para ser acionado de cinco em cinco minutos. Com essas informações, foi possível avançar para a próxima etapa do processo de BI.

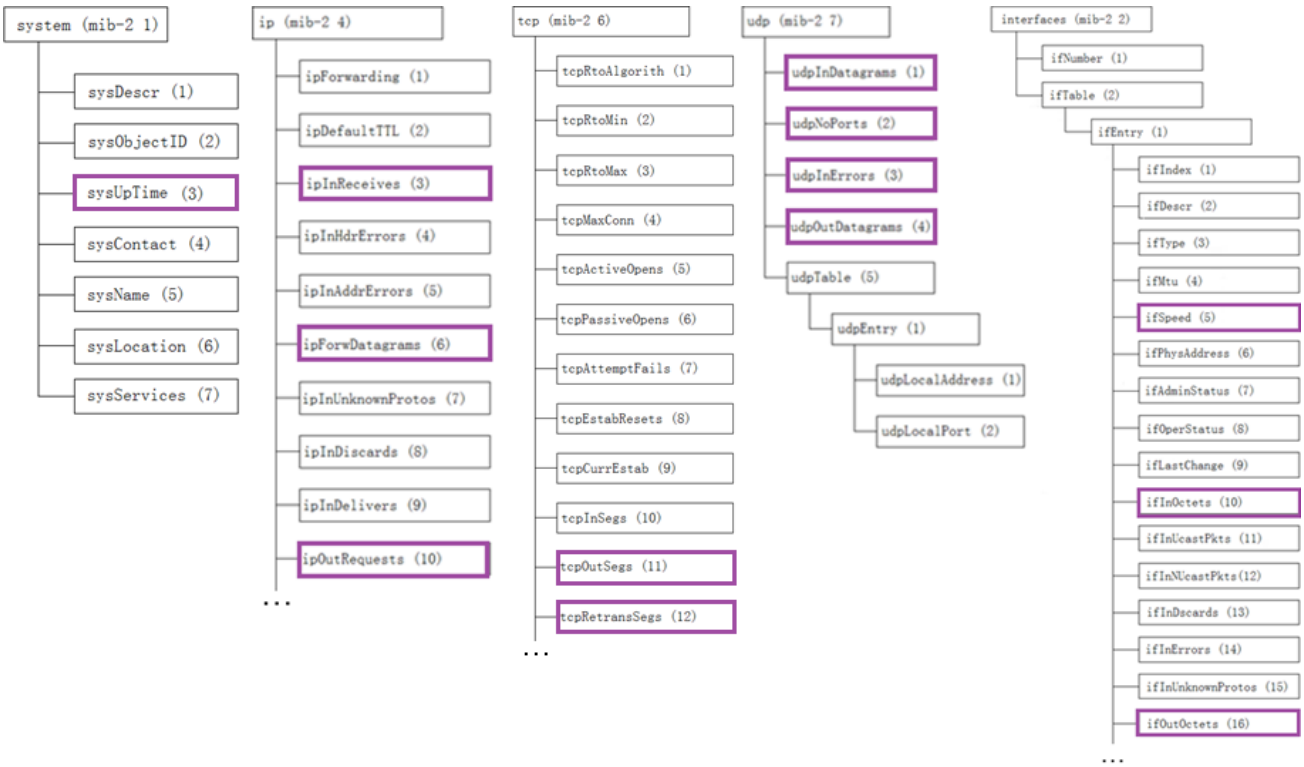



Figura 6.1 – Objetos gerenciáveis coletados

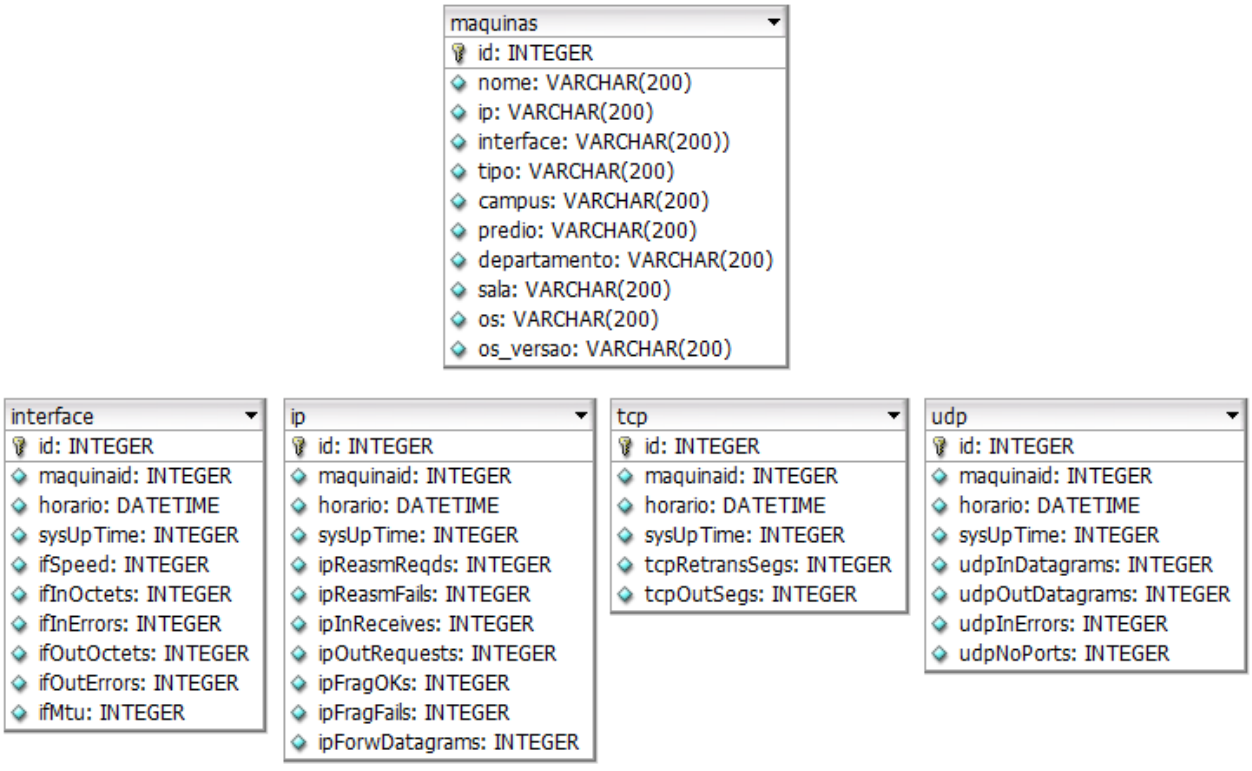



Figura 6.2 – Estrutura da fonte de dados

## 6.3 MODELAGEM DO DW

A modelagem de DW é uma parte muito importante do processo de BI. Uma modelagem inadequada pode trazer informações que não são úteis ao negócio. Esse processo é também

muito particular, sendo que não é possível fazer uma modelagem padrão que se encaixe em vários cenários. Cada uma deve ser feita com base nas necessidades específicas e atuais de cada aplicação.

O primeiro passo de nossa modelagem foi a definição das dimensões do DW. O critério para a escolha foi baseado no cenário e no entendimento de quais informações seriam relevantes ao processo de gerência. Como o espaço para testes era bastante restrito, nem todas as dimensões puderam ser preenchidas, porém foram criadas para ilustrar uma implementação mais real. As dimensões criadas foram: tempo, equipamento, localidade e sistema operacional. A tabela fato, conseqüentemente, contém referências a cada uma das dimensões e os indicadores utilizados. O cálculo dos indicadores será visto na seção seguinte. A estrutura do DW utilizado pode ser vista na figura 6.3.

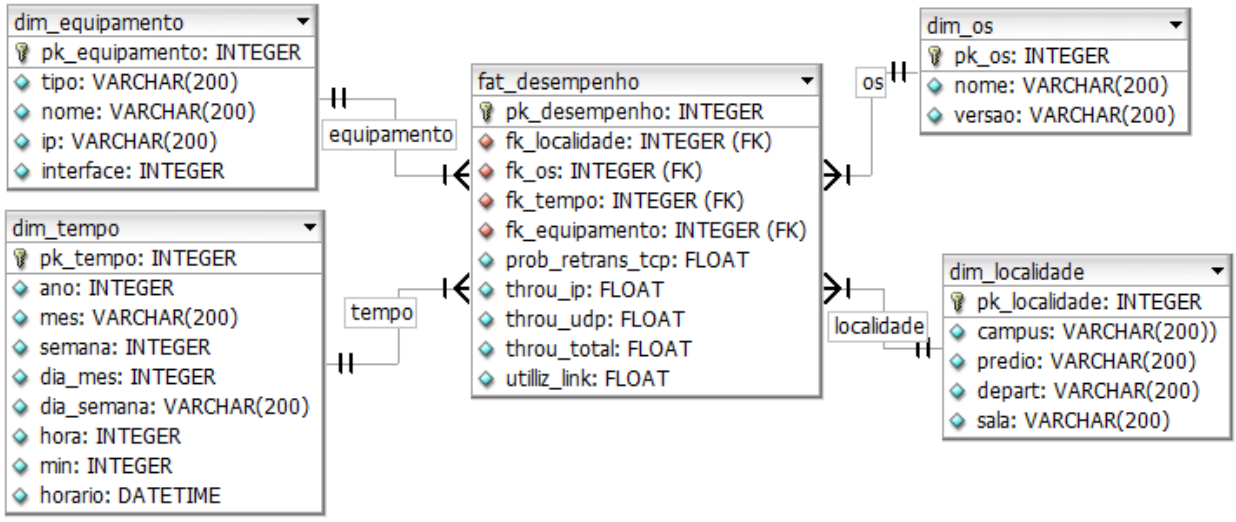



Figura 6.3 – Estrutura do DW.

Feita essa modelagem, é necessário popular o DW, procedimento que é feito na etapa de ETL.

## 6.4 DESCRIÇÃO DO ETL

Na etapa de ETL, foram escolhidos os indicadores, dentre os descritos em [14], que faziam sentido em nosso cenário de testes. Os escolhidos foram:

- TCP *Segments Retransmission Probability*: refere-se à probabilidade de retransmissão de segmentos TCP e é dado por:

$$\text{TCP } Segments\ Retransmission\ Probability = \frac{\Delta\text{tcpRetransSegs}}{\Delta\text{tcpOutSegs}} \quad (1)$$

- IP *Throughput*: refere-se ao *throughput* do protocolo IP de uma dada interface e é dado por:

$$\text{IP } Throughput = \frac{(\Delta\text{ipInReceives} + \Delta\text{ipOutRequests} + \Delta\text{ipForwDatagrams})}{\Delta\text{sysUpTime}} \quad (2)$$

- UDP *Throughput*: refere-se ao *throughput* do protocolo UDP de uma dada interface e é dado por:

$$\text{UDP } Throughput = \frac{(\Delta\text{udpInDatagrams} + \Delta\text{udpNoPorts} + \Delta\text{udpInErrors} + \Delta\text{udpOutDatagrams})}{\Delta\text{sysUpTime}} \quad (3)$$

- *Throughput* Total: refere-se ao *throughput* total da interface em questão e é dado por:

$$Throughput\ Total = \frac{(\Delta ifInOctets\ + \Delta ifOutOctets\ )}{\Delta sysUpTime} \quad (4)$$

- *Link Utilization*: refere-se à utilização relativa de uma dada interface e é dado por:

$$Link\ Utilization = \frac{Throug\ hput\ Total * 8}{ifSpeed} \quad (5)$$

Foi necessário realizar alguns ajustes para tornar as medidas mais familiares. Os indicadores IP *Throughput* e UDP *Throughput*, pelo cálculo acima, resultavam em uma unidade de datagramas por centésimo de segundo. Foi feita uma transformação simples para que eles fossem exibidos em datagramas por segundo. O indicador *Throughput* Total resultava em uma unidade de bytes por centésimo de segundo. Uma segunda modificação foi feita com tal indicador para que fosse exibido em kilobits por segundo. Essa modificação afetou indiretamente o indicador *Link Utilization*, que foi devidamente corrigido. O cálculo e o ajuste dos indicadores foram feitos por meio de um script PHP, também responsável por armazenar os resultados em uma tabela chamada "indicadores".

A estrutura utilizada para apoiar o processo de ETL pode ser vista na figura abaixo:

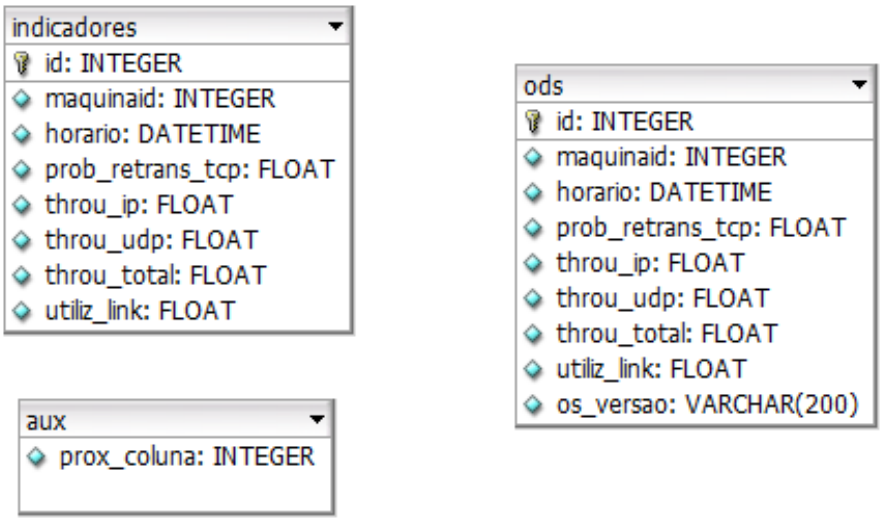



Figura 6.4 – Estrutura de apoio ao ETL

Além da tabela "indicadores" e da tabela "aux", utilizada apenas para auxiliar o preenchimento da primeira, foi criada a tabela "ods", o *Operational Data Store*, utilizada como intermediária para carga dos dados no DW.

O Pentaho possui uma ferramenta de ETL chamada Kettle, que possui uma interface gráfica amigável onde as operações são feitas em um modelo *drag and drop*. Apesar de ser uma ferramenta cômoda, especialmente para usuários não familiarizados com a linguagem SQL, o Kettle pode se tornar um contratempo para usuários avançados. Por esse motivo, o processo de ETL realizado no projeto foi feito pelo uso direto de *scripts* PHP e *queries* SQL.

## 6.5 FORMAS DE APRESENTAÇÃO DOS INDICADORES DE GERÊNCIA DE REDES

Após a realização das etapas que levaram à construção e carga do DW, tem-se a possibilidade de analisar os dados, nesse caso os indicadores de gerência, de diversas maneiras. Os modos de exibição dos dados no Pentaho se dividem em quatro categorias principais: consultas *ad*

*hoc*, relatórios, gráficos e painéis. No presente trabalho, foram priorizadas as consultas *ad hoc* e a construção de relatórios.

Assim como no processo de ETL, a criação das consultas *ad hoc* pode ser realizada através de um módulo do Pentaho, chamado Schema Workbench, ou manualmente através da criação de arquivos XML que definem as dimensões e medidas a serem apresentadas. Neste trabalho, optamos por realizar a criação manual das visualizações. Definido o modelo das consultas, é gerada uma página web que permite a análise interativa dos dados, envolvendo as operações de *drill up*, *drill down*, *slice and dice* e outras suportadas pela plataforma Pentaho. Na figura 6.5 pode ser visto um exemplo de uma possível visualização para análise das informações. Os gráficos de pizza representam uma medida relativa, comparando apenas as quatro máquinas monitoradas, ou seja, por meio dele é possível visualizar a contribuição proporcional de cada uma das máquinas. Nesta análise, são exibidas as médias temporais de todos os indicadores para cada um dos equipamentos monitorados. É importante ressaltar que as unidades de medida dos indicadores *Throughput* IP e *Throughput* UDP são diferentes da unidade do indicador *Throughput* Total, o que não permite uma comparação direta entre eles. Além disso, não seria possível relacioná-los através de uma medida proporcional, dividindo o *Throughput* Total para cada um dos *throughputs* dos protocolos. Isso ocorre pois há outros protocolos, como o TCP e o ICMP, que influem no valor do *Throughput* Total. Outro ponto importante a ser ressaltado são as altas probabilidades de retransmissão TCP para os servidores DNS e Web. Esses valores sugerem que há algum tipo de ataque acontecendo com tais máquinas, especialmente no caso do DNS, que usualmente se comunica via UDP. Por esse motivo, foi criada outra análise, que permite a visualização da probabilidade de retransmissão TCP para o servidor DNS por dias da semana. Essa visualização pode ser vista na figura 6.6.

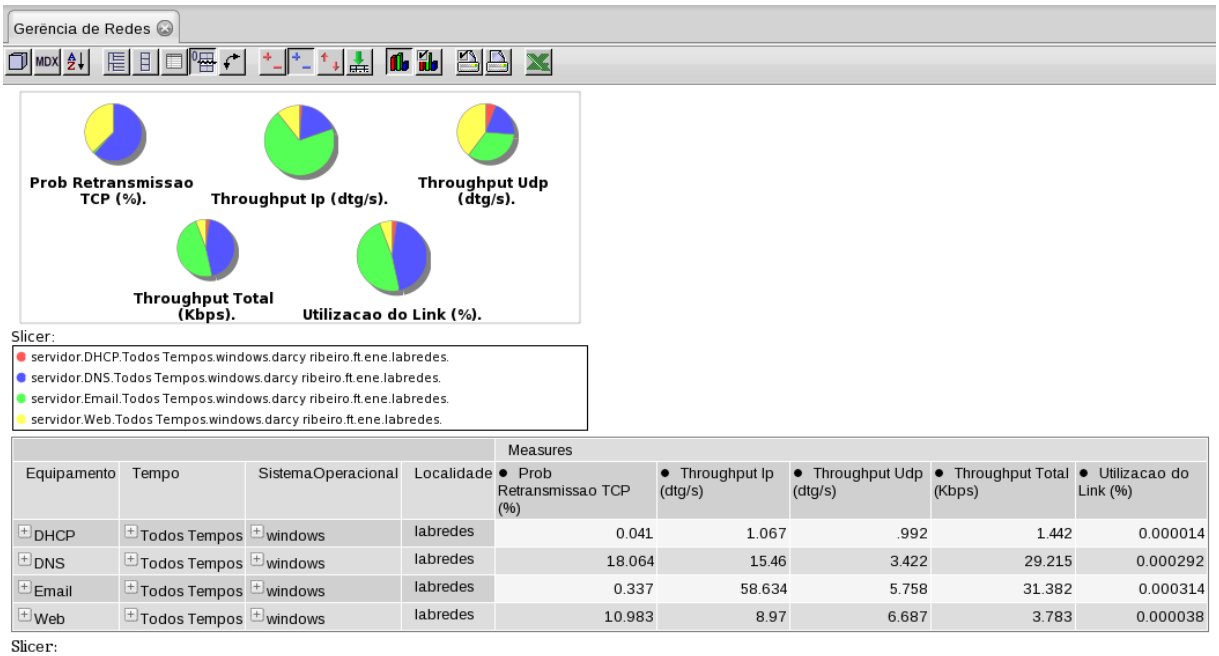



Figura 6.5 – Exemplo de visualização para análise

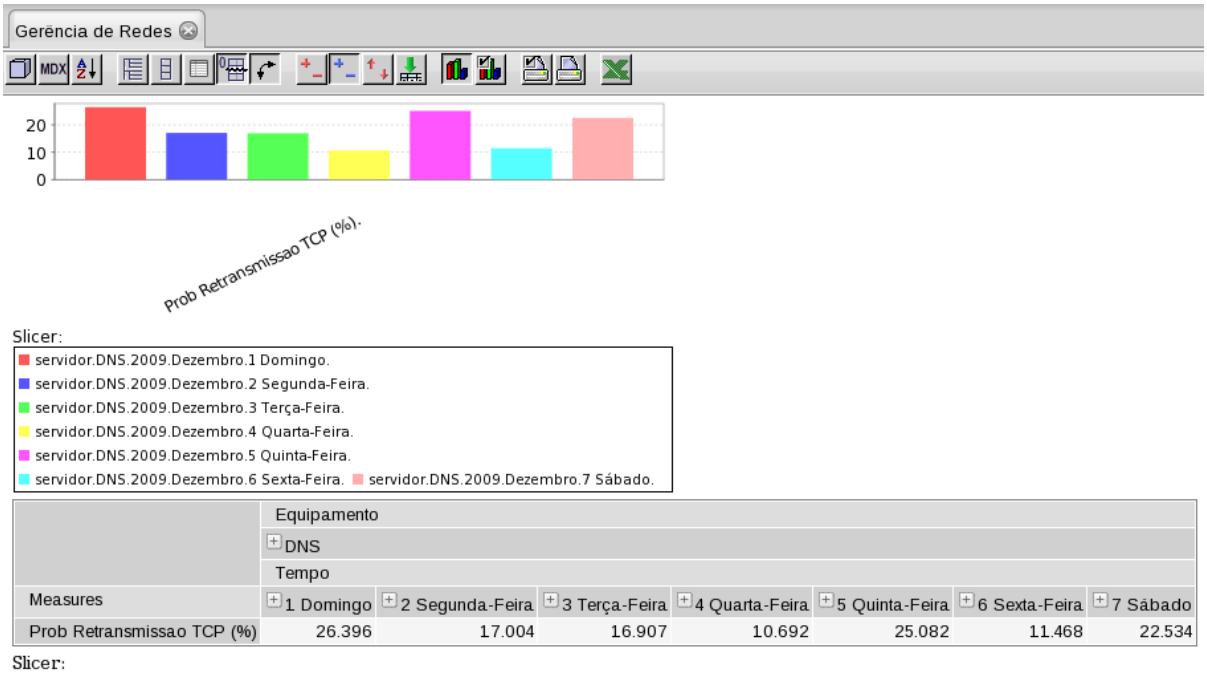



Figura 6.6 – Visualização de suspeita de ataque

O Pentaho Suite possui um módulo que dá suporte à geração de modelos de relatórios, chamado Report Designer. Nele são definidos os campos a serem exibidos, bem como todas as informações que dizem respeito à formatação dos dados e aparência dos relatórios. É possível personalizar os relatórios tanto quanto se queira, de modo a adequá-los às necessidades do usuário final. Concluída a etapa de construção do formato dos relatórios, basta acessar o modelo criado pelo console principal e escolher a extensão do arquivo para qual o relatório será exportado. As extensões suportadas são PDF, XLS, DOC e CSV. Exemplos de relatórios podem ser vistos a seguir.

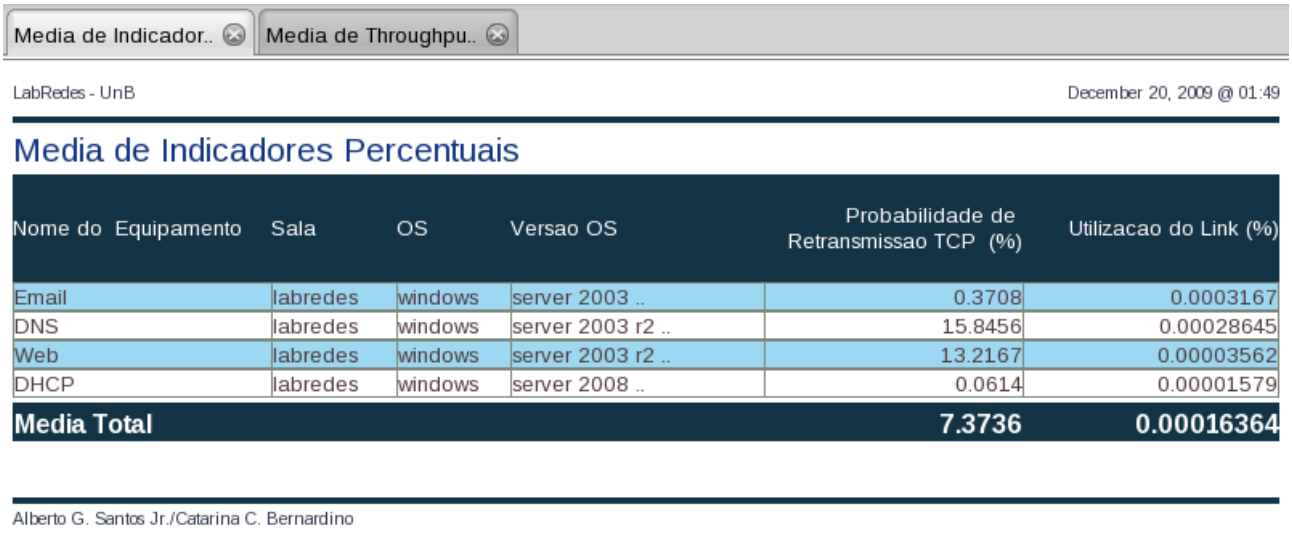



Figura 6.7 – Relatório da média de indicadores percentuais por equipamento

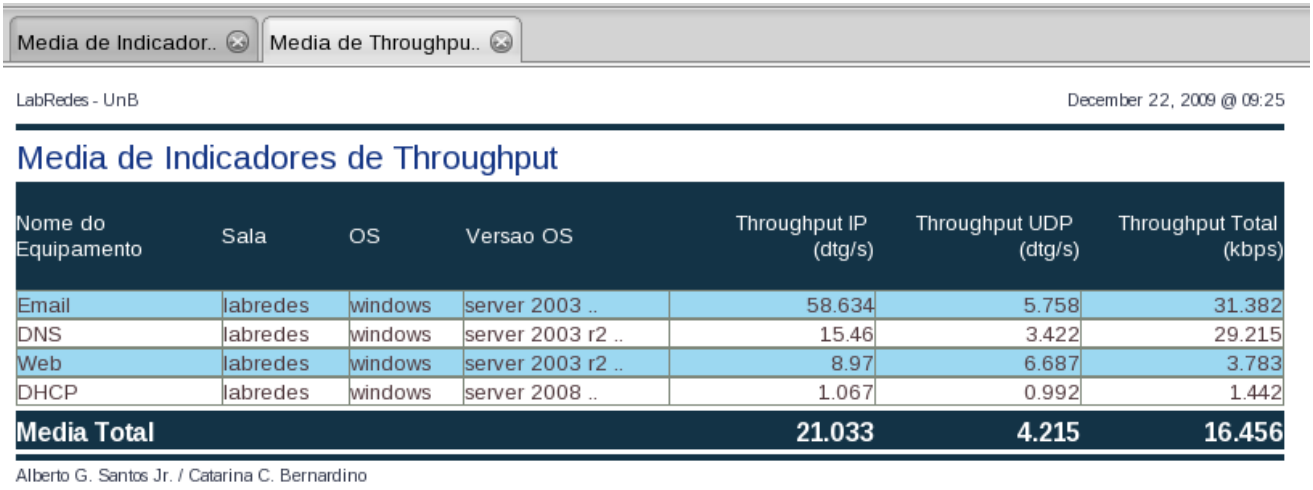



Figura 6.8 – Relatório da média de indicadores de *throughput* por equipamento

Com o apoio das análises por consultas *ad hoc* e dos relatórios é possível acompanhar o desempenho das máquinas analisadas de uma maneira muito útil e proveitosa, proporcionando uma facilidade na resolução de problemas, planejamento e tomada de decisões.

A figura 6.9 ilustra, de forma sintetizada, todo o processo que foi descrito nas seções 6.2 a 6.5.

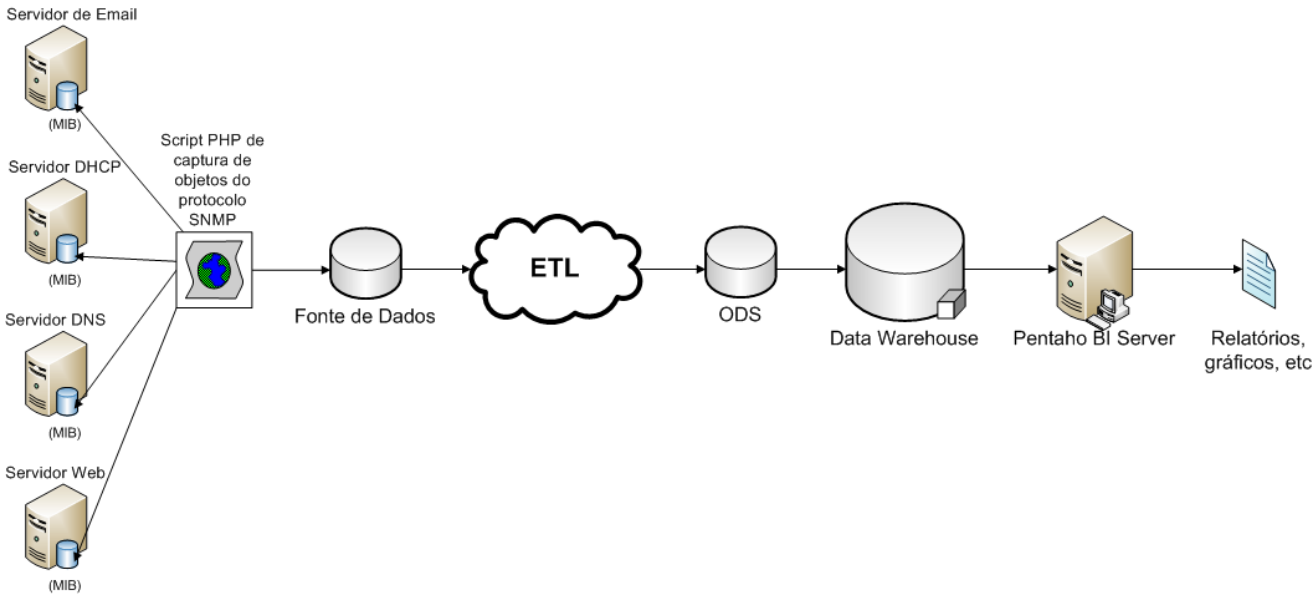



Figura 6.9 – Resumo do processo de *Business Intelligence*

# 7 CONCLUSÕES

Nossa hipótese inicial era mostrar o potencial e os benefícios que se pode obter de um sistema de BI para exploração de indicadores de gerência de redes.

Por essa razão, nós fizemos uma implementação modelo que permitisse averiguar, de fato, a utilidade e os benefícios que um sistema de gerência de redes teria caso tivesse o apoio de uma plataforma de BI.

Os resultados obtidos no projeto confirmam nossa hipótese original nos seguintes aspectos:

- Visualização e análise, de maneira rápida e intuitiva, dos dados históricos do processo de gerência, auxiliando no planejamento, resolução de problemas e na tomada de decisões;
- Associação de métricas operacionais (SNMP) com indicadores gerenciais (BI) pode ser feita de maneira a tornar mais evidente para um gerente a ocorrência de eventos significativos na rede e, portanto, facilitar a tomada de decisão por esse gerente;
- Adequação ótima da plataforma de gerência a cada ambiente gerenciado por meio da escolha dos melhores indicadores para cada cenário;
- Dada a ubiqüidade do SNMP, há uma extrema escalabilidade e portabilidade do sistema, tornando sua possibilidade de expansão praticamente indefinida.

Além disso, tais resultados e o trabalho feito abrem as seguintes perspectivas de trabalhos futuros:

- Coleta e análise de todos os dados presentes nas MIBs dos equipamentos de modo a tornar a gerência mais eficiente e adaptativa a cada cenário de rede;
- Criação de painéis interativos que contenham informações úteis, organizadas em níveis de hierarquia, permitindo ao gerente acesso rápido às informações da rede;
- Desenvolvimento de uma ferramenta completa de gerência de redes com o uso da tecnologia de BI.

- Desenvolver novas associações entre indicadores gerenciais e métricas operacionais de gerência de rede, nas diversas áreas funcionais.

# REFERÊNCIAS BIBLIOGRÁFICAS


[1] BLACK, T. L. Comparação de Ferramentas de Gerenciamento de Redes. 2008. 64 f. Dissertação (Especialização em Tecnologias, Gerência e Segurança de Redes de Computadores) - Instituto de Informática, Universidade Federal do Rio Grande do Sul, Rio Grande do Sul, 2008.

[2] STALLINGS, William. SNMP, SNMPv2, SNMPv3, and RMON 1 and 2. 3. ed. Canadá: Addison-Wesley, 1999. 619 p.

[3] ITU-T. Recommendation M.3010: Principles for a Telecommunications Management Network, M.3010. Melbourne, 1988, revisada em 1996. 75 p.

[4] Cisco Systems. In: Internetworking Technologies Handbook. 2003. 4 ed. p. 56-1 a 56-12.

[5] BIBBS, E., MATT, B. Comparison of SNMP: Version 1, 2 and 3. Xin Tang, 2006.

[6] MAURO, Douglas, SCHMIDT, Kevin. Essential SNMP. 2. ed. EUA: O'Reilly Media, 2005. 460 p.

[7] OETIKER, T. RRDtool Documentation. Disponível em: <http://oss.oetiker.ch/rrdtool/doc/rrdtool.en.html>. Acesso em: 28 de outubro 2009.

[8] BERRY, I., ROMAN, T., ADAMS, L., PASNAK, J., CONNER, J., SCHECK, R. The Cacti Manual. 2007.

[9] BATISTA, Emerson. Sistemas de Informação - O Uso Consciente da Tecnologia para o Gerenciamento. 1. ed. Saraiva, 2004. 282 p.

[10] MARTINS, V. A. Aplicações de BI, Ambientes de DW, Plataforma Pentaho de BI e Ferramentas, Introdução ao OLAP e Modelagem Multidimensional, Desenhando um Esquema Estrela, Utilizando ETL para a construção do DW, Importando um Banco de Dados fonte. 2007. 45 p.

[11] RIBEIRO, E. F. Business Intelligence como Garantia de Diferencial Competitivo. 2005. 70 p. Dissertação (Bacharel em Sistemas de Informação) - Faculdade de Ciências Aplicadas de Minas, União Educacional Minas Gerais, Minas Gerais, 2008.

[12] CHAUDHURI, S., DAYAL, U. An Overview of Data Warehousing and OLAP Technology. EUA: ACM SIGMOD Record, 1997.

[13] KIMBALL, Ralph, ROSS, Margy. The Data Warehouse Toolkit. 2. ed. EUA: John Wiley & Sons, 2002. 421 p.

[14] APOSTOLOPOULOS, T. DASKALOU, V. A Model for SNMP Based Performance Management Services. IEEE Catalogue No. 95TH8061, p. 269-273, 1995.

# ANEXO I

Este anexo contém os scripts php utilizados na captura de dados e no processo de ETL.

- Script que faz a conexão com o banco de dados.

```
<?php
$hostname_localhost = "localhost";
$database_localhost = "projeto";
$username_localhost = "root";
$password_localhost = "******";
$localhost = mysql_pconnect($hostname_localhost, $username_localhost,
  $password_localhost) or trigger_error(mysql_error(),E_USER_ERROR);
?>
```

- Script que captura os objetos do protocolo snmp, faz o tratamento dos dados e armazena-os no banco de dados MySQL.

```
<?php
include('conexao.php');
error_reporting(E_ALL);
ini_set('display_errors', '1');
$database_localhost = "projeto";
mysql_select_db($database_localhost, $localhost);
$query_rsMaquinas = "SELECT * FROM maquinas";
$rsMaquinas = mysql_query($query_rsMaquinas, $localhost) or die(mysql_error());
$row_rsMaquinas = mysql_fetch_assoc($rsMaquinas);
$totalRows_rsMaquinas = mysql_num_rows($rsMaquinas);
$community = "projeto";

    do {
    $maquina_id = $row_rsMaquinas['id'];
    $timestamp = date("Y-m-d H:i:s");

    $sysupTime1 = snmpget($row_rsMaquinas['ip'], $community,
"1.3.6.1.2.1.1.3.0");
    $sysupTime2 = explode("(", $sysupTime1);
    $sysupTime3 = explode(")",$sysupTime2[1]);
    $sysupTime = $sysupTime3[0];
```

```
      $ifSpeed1 = snmpget($row_rsMaquinas['ip'], $community,
"1.3.6.1.2.1.2.2.1.5.".$row_rsMaquinas['interface']);

      $ifSpeed2 = explode(" ",$ifSpeed1);

      $ifSpeed = $ifSpeed2[1];

      $ifInOctets1 = snmpget($row_rsMaquinas['ip'], $community,
"1.3.6.1.2.1.2.2.1.10.".$row_rsMaquinas['interface']);

         $ifInOctets2 = explode(" ",$ifInOctets1);

         $ifInOctets = $ifInOctets2[1];

      $ifInErrors1 = snmpget($row_rsMaquinas['ip'], $community,
"1.3.6.1.2.1.2.2.1.14.".$row_rsMaquinas['interface']);

         $ifInErrors2 = explode(" ",$ifInErrors1);

         $ifInErrors = $ifInErrors2[1];

      $ifOutOctets1 = snmpget($row_rsMaquinas['ip'], $community,
"1.3.6.1.2.1.2.2.1.16.".$row_rsMaquinas['interface']);

         $ifOutOctets2 = explode(" ",$ifOutOctets1);

         $ifOutOctets = $ifOutOctets2[1];

      $ifOutErrors1 = snmpget($row_rsMaquinas['ip'], $community,
"1.3.6.1.2.1.2.2.1.20.".$row_rsMaquinas['interface']);

         $ifOutErrors2 = explode(" ",$ifOutErrors1);

         $ifOutErrors = $ifOutErrors2[1];


      $ipReasmReqds1 = snmpget($row_rsMaquinas['ip'], $community,
"1.3.6.1.2.1.4.14.0");

         $ipReasmReqds2 = explode(" ",$ipReasmReqds1);

         $ipReasmReqds = $ipReasmReqds2[1];

      $ipReasmFails1 = snmpget($row_rsMaquinas['ip'], $community,
"1.3.6.1.2.1.4.16.0");

         $ipReasmFails2 = explode(" ",$ipReasmFails1);

         $ipReasmFails = $ipReasmFails2[1];

      $ipInReceives1 = snmpget($row_rsMaquinas['ip'], $community,
"1.3.6.1.2.1.4.3.0");

         $ipInReceives2 = explode(" ",$ipInReceives1);

         $ipInReceives = $ipInReceives2[1];

      $ipOutRequests1 = snmpget($row_rsMaquinas['ip'], $community,
"1.3.6.1.2.1.4.10.0");

         $ipOutRequests2 = explode(" ",$ipOutRequests1);

         $ipOutRequests = $ipOutRequests2[1];

      $ipFragOKs1 = snmpget($row_rsMaquinas['ip'], $community,
"1.3.6.1.2.1.4.17.0");
```

```
        $ipFragOKs2 = explode(" ",$ipFragOKs1);
        $ipFragOKs = $ipFragOKs2[1];
    $ipFragFails1 = snmpget($row_rsMaquinas['ip'], $community,
"1.3.6.1.2.1.4.18.0");
        $ipFragFails2 = explode(" ",$ipFragFails1);
        $ipFragFails = $ipFragFails2[1];
    $ipForwDatagrams1 = snmpget($row_rsMaquinas['ip'], $community,
"1.3.6.1.2.1.4.6.0");
        $ipForwDatagrams2 = explode(" ",$ipForwDatagrams1);
        $ipForwDatagrams = $ipForwDatagrams2[1];


    $tcpRetransSegs1 = snmpget($row_rsMaquinas['ip'], $community,
"1.3.6.1.2.1.6.12.0");
        $tcpRetransSegs2 = explode(" ",$tcpRetransSegs1);
        $tcpRetransSegs = $tcpRetransSegs2[1];
    $tcpOutSegs1 = snmpget($row_rsMaquinas['ip'], $community,
"1.3.6.1.2.1.6.11.0");
        $tcpOutSegs2 = explode(" ",$tcpOutSegs1);
        $tcpOutSegs = $tcpOutSegs2[1];


    $udpInDatagrams1 = snmpget($row_rsMaquinas['ip'], $community,
"1.3.6.1.2.1.7.1.0");
        $udpInDatagrams2 = explode(" ",$udpInDatagrams1);
        $udpInDatagrams = $udpInDatagrams2[1];
    $udpOutDatagrams1 = snmpget($row_rsMaquinas['ip'], $community,
"1.3.6.1.2.1.7.4.0");
        $udpOutDatagrams2 = explode(" ",$udpOutDatagrams1);
        $udpOutDatagrams = $udpOutDatagrams2[1];
    $udpInErrors1 = snmpget($row_rsMaquinas['ip'], $community,
"1.3.6.1.2.1.7.3.0");
        $udpInErrors2 = explode(" ",$udpInErrors1);
        $udpInErrors = $udpInErrors2[1];
    $udpNoPorts1 = snmpget($row_rsMaquinas['ip'], $community,
"1.3.6.1.2.1.7.2.0");
        $udpNoPorts2 = explode(" ",$udpNoPorts1);
        $udpNoPorts = $udpNoPorts2[1];
```

```
	$query_if = "INSERT INTO interface
(maquinaid,horario,sysUpTime,ifSpeed,ifInOctets,ifInErrors,ifOutOctets,ifOutErrors
)
VALUES('$maquina_id','$timestamp','$sysupTime','$ifSpeed','$ifInOctets','$ifInErro
rs','$ifOutOctets','$ifOutErrors')";

	mysql_query($query_if,$localhost) or die(mysql_error());


	$query_ip = "INSERT INTO ip
(maquinaid,horario,sysUpTime,ipReasmReqds,ipReasmFails,ipInReceives,ipOutRequests,
ipFragOKs,ipFragFails,ipForwDatagrams) VALUES
('$maquina_id','$timestamp','$sysupTime','$ipReasmReqds','$ipReasmFails','$ipInRec
eives','$ipOutRequests','$ipFragOKs','$ipFragFails','$ipForwDatagrams')";

	mysql_query($query_ip,$localhost) or die(mysql_error());


	$query_tcp = "INSERT INTO tcp
(maquinaid,horario,sysUpTime,tcpRetransSegs,tcpOutSegs) VALUES
('$maquina_id','$timestamp','$sysupTime','$tcpRetransSegs','$tcpOutSegs')";

	mysql_query($query_tcp,$localhost) or die(mysql_error());


	$query_udp = "INSERT INTO udp
(maquinaid,horario,sysUpTime,udpInDatagrams,udpOutDatagrams,udpInErrors,udpNoPorts
) VALUES
('$maquina_id','$timestamp','$sysupTime','$udpInDatagrams','$udpOutDatagrams','$ud
pInErrors','$udpNoPorts')";

	mysql_query($query_udp,$localhost) or die(mysql_error());


	} while ($row_rsMaquinas = mysql_fetch_assoc($rsMaquinas));
?>
```

- Script que faz o cálculo dos indicadores e armazena-os no banco de dados MySQL.

```
<?php
include('conexao.php');
error_reporting(E_ALL);
ini_set('display_errors', '1');
mysql_select_db($database_localhost, $localhost);


$query_rsInterface1 = "SELECT * FROM interface where maquinaid=1";
$rsInterface1 = mysql_query($query_rsInterface1, $localhost) or
die(mysql_error());
$query_rsInterface2 = "SELECT * FROM interface where maquinaid=2";
```

```
   $rsInterface2 = mysql_query($query_rsInterface2, $localhost) or
die(mysql_error());
   $query_rsInterface3 = "SELECT * FROM interface where maquinaid=3";
   $rsInterface3 = mysql_query($query_rsInterface3, $localhost) or
die(mysql_error());
   $query_rsInterface4 = "SELECT * FROM interface where maquinaid=4";
   $rsInterface4 = mysql_query($query_rsInterface4, $localhost) or
die(mysql_error());


   $query_rsIp1 = "SELECT * FROM ip where maquinaid=1";
   $rsIp1 = mysql_query($query_rsIp1, $localhost) or die(mysql_error());
   $query_rsIp2 = "SELECT * FROM ip where maquinaid=2";
   $rsIp2 = mysql_query($query_rsIp2, $localhost) or die(mysql_error());
   $query_rsIp3 = "SELECT * FROM ip where maquinaid=3";
   $rsIp3 = mysql_query($query_rsIp3, $localhost) or die(mysql_error());
   $query_rsIp4 = "SELECT * FROM ip where maquinaid=4";
   $rsIp4 = mysql_query($query_rsIp4, $localhost) or die(mysql_error());


   $query_rsTcp1 = "SELECT * FROM tcp where maquinaid=1";
   $rsTcp1 = mysql_query($query_rsTcp1, $localhost) or die(mysql_error());
   $query_rsTcp2 = "SELECT * FROM tcp where maquinaid=2";
   $rsTcp2 = mysql_query($query_rsTcp2, $localhost) or die(mysql_error());
   $query_rsTcp3 = "SELECT * FROM tcp where maquinaid=3";
   $rsTcp3 = mysql_query($query_rsTcp3, $localhost) or die(mysql_error());
   $query_rsTcp4 = "SELECT * FROM tcp where maquinaid=4";
   $rsTcp4 = mysql_query($query_rsTcp4, $localhost) or die(mysql_error());


   $query_rsUdp1 = "SELECT * FROM udp where maquinaid=1";
   $rsUdp1 = mysql_query($query_rsUdp1, $localhost) or die(mysql_error());
   $query_rsUdp2 = "SELECT * FROM udp where maquinaid=2";
   $rsUdp2 = mysql_query($query_rsUdp2, $localhost) or die(mysql_error());
   $query_rsUdp3 = "SELECT * FROM udp where maquinaid=3";
   $rsUdp3 = mysql_query($query_rsUdp3, $localhost) or die(mysql_error());
   $query_rsUdp4 = "SELECT * FROM udp where maquinaid=4";
   $rsUdp4 = mysql_query($query_rsUdp4, $localhost) or die(mysql_error());


   $query_rsAntigoRows = "SELECT prox_coluna FROM aux";
```

```
   $totalAntigoRows = mysql_query($query_rsAntigoRows, $localhost) or
die(mysql_error());


   $totalRows = mysql_num_rows($rsInterface1);
   $ifSpeed1 = mysql_result($rsInterface1,0,'ifSpeed');
   $ifSpeed2 = mysql_result($rsInterface2,0,'ifSpeed');
   $ifSpeed3 = mysql_result($rsInterface3,0,'ifSpeed');
   $ifSpeed4 = mysql_result($rsInterface4,0,'ifSpeed');


      for ($i=($totalAntigoRows+1); $i < $totalRows; $i++)      {
            $j = $i - 1;
            $dTcpRetransSegs1 = mysql_result($rsTcp1,$i,'tcpRetransSegs') -
mysql_result($rsTcp1,$j,'tcpRetransSegs');
            $dTcpRetransSegs2 = mysql_result($rsTcp2,$i,'tcpRetransSegs') -
mysql_result($rsTcp2,$j,'tcpRetransSegs');
            $dTcpRetransSegs3 = mysql_result($rsTcp3,$i,'tcpRetransSegs') -
mysql_result($rsTcp3,$j,'tcpRetransSegs');
            $dTcpRetransSegs4 = mysql_result($rsTcp4,$i,'tcpRetransSegs') -
mysql_result($rsTcp4,$j,'tcpRetransSegs');


            $dTcpOutSegs1 = mysql_result($rsTcp1,$i,'tcpOutSegs') -
mysql_result($rsTcp1,$j,'tcpOutSegs');
            $dTcpOutSegs2 = mysql_result($rsTcp2,$i,'tcpOutSegs') -
mysql_result($rsTcp2,$j,'tcpOutSegs');
            $dTcpOutSegs3 = mysql_result($rsTcp3,$i,'tcpOutSegs') -
mysql_result($rsTcp3,$j,'tcpOutSegs');
            $dTcpOutSegs4 = mysql_result($rsTcp4,$i,'tcpOutSegs') -
mysql_result($rsTcp4,$j,'tcpOutSegs');


            $dIpInReceives1 = mysql_result($rsIp1,$i,'ipInReceives') -
mysql_result($rsIp1,$j,'ipInReceives');
            $dIpInReceives2 = mysql_result($rsIp2,$i,'ipInReceives') -
mysql_result($rsIp2,$j,'ipInReceives');
            $dIpInReceives3 = mysql_result($rsIp3,$i,'ipInReceives') -
mysql_result($rsIp3,$j,'ipInReceives');
            $dIpInReceives4 = mysql_result($rsIp4,$i,'ipInReceives') -
mysql_result($rsIp4,$j,'ipInReceives');


            $dIpOutRequests1 = mysql_result($rsIp1,$i,'ipOutRequests') -
mysql_result($rsIp1,$j,'ipOutRequests');
```

```
            $dIpOutRequests2 = mysql_result($rsIp2,$i,'ipOutRequests') -
mysql_result($rsIp2,$j,'ipOutRequests');

            $dIpOutRequests3 = mysql_result($rsIp3,$i,'ipOutRequests') -
mysql_result($rsIp3,$j,'ipOutRequests');

            $dIpOutRequests4 = mysql_result($rsIp4,$i,'ipOutRequests') -
mysql_result($rsIp4,$j,'ipOutRequests');


            $dIpForwDatagrams1 = mysql_result($rsIp1,$i,'ipForwDatagrams') -
mysql_result($rsIp1,$j,'ipForwDatagrams');

            $dIpForwDatagrams2 = mysql_result($rsIp2,$i,'ipForwDatagrams') -
mysql_result($rsIp2,$j,'ipForwDatagrams');

            $dIpForwDatagrams3 = mysql_result($rsIp3,$i,'ipForwDatagrams') -
mysql_result($rsIp3,$j,'ipForwDatagrams');

            $dIpForwDatagrams4 = mysql_result($rsIp4,$i,'ipForwDatagrams') -
mysql_result($rsIp4,$j,'ipForwDatagrams');


            $dSysUpTime1 = mysql_result($rsIp1,$i,'sysUpTime') -
mysql_result($rsIp1,$j,'sysUpTime');

            $dSysUpTime2 = mysql_result($rsIp2,$i,'sysUpTime') -
mysql_result($rsIp2,$j,'sysUpTime');

            $dSysUpTime3 = mysql_result($rsIp3,$i,'sysUpTime') -
mysql_result($rsIp3,$j,'sysUpTime');

            $dSysUpTime4 = mysql_result($rsIp4,$i,'sysUpTime') -
mysql_result($rsIp4,$j,'sysUpTime');


            $dUdpInDatagrams1 = mysql_result($rsUdp1,$i,'udpInDatagrams') -
mysql_result($rsUdp1,$j,'udpInDatagrams');

            $dUdpInDatagrams2 = mysql_result($rsUdp2,$i,'udpInDatagrams') -
mysql_result($rsUdp2,$j,'udpInDatagrams');

            $dUdpInDatagrams3 = mysql_result($rsUdp3,$i,'udpInDatagrams') -
mysql_result($rsUdp3,$j,'udpInDatagrams');

            $dUdpInDatagrams4 = mysql_result($rsUdp4,$i,'udpInDatagrams') -
mysql_result($rsUdp4,$j,'udpInDatagrams');


            $dUdpNoPorts1 = mysql_result($rsUdp1,$i,'udpNoPorts') -
mysql_result($rsUdp1,$j,'udpNoPorts');

            $dUdpNoPorts2 = mysql_result($rsUdp2,$i,'udpNoPorts') -
mysql_result($rsUdp2,$j,'udpNoPorts');

            $dUdpNoPorts3 = mysql_result($rsUdp3,$i,'udpNoPorts') -
mysql_result($rsUdp3,$j,'udpNoPorts');
```

```
            $dUdpNoPorts4 = mysql_result($rsUdp4,$i,'udpNoPorts') -
mysql_result($rsUdp4,$j,'udpNoPorts');


            $dUdpInErrors1 = mysql_result($rsUdp1,$i,'udpInErrors') -
mysql_result($rsUdp1,$j,'udpInErrors');
            $dUdpInErrors2 = mysql_result($rsUdp2,$i,'udpInErrors') -
mysql_result($rsUdp2,$j,'udpInErrors');
            $dUdpInErrors3 = mysql_result($rsUdp3,$i,'udpInErrors') -
mysql_result($rsUdp3,$j,'udpInErrors');
            $dUdpInErrors4 = mysql_result($rsUdp4,$i,'udpInErrors') -
mysql_result($rsUdp4,$j,'udpInErrors');


            $dUdpOutDatagrams1 = mysql_result($rsUdp1,$i,'udpOutDatagrams') -
mysql_result($rsUdp1,$j,'udpOutDatagrams');
            $dUdpOutDatagrams2 = mysql_result($rsUdp2,$i,'udpOutDatagrams') -
mysql_result($rsUdp2,$j,'udpOutDatagrams');
            $dUdpOutDatagrams3 = mysql_result($rsUdp3,$i,'udpOutDatagrams') -
mysql_result($rsUdp3,$j,'udpOutDatagrams');
            $dUdpOutDatagrams4 = mysql_result($rsUdp4,$i,'udpOutDatagrams') -
mysql_result($rsUdp4,$j,'udpOutDatagrams');


            $dIfInOctets1 = mysql_result($rsInterface1,$i,'ifInOctets') -
mysql_result($rsInterface1,$j,'ifInOctets');
            $dIfInOctets2 = mysql_result($rsInterface2,$i,'ifInOctets') -
mysql_result($rsInterface2,$j,'ifInOctets');
            $dIfInOctets3 = mysql_result($rsInterface3,$i,'ifInOctets') -
mysql_result($rsInterface3,$j,'ifInOctets');
            $dIfInOctets4 = mysql_result($rsInterface4,$i,'ifInOctets') -
mysql_result($rsInterface4,$j,'ifInOctets');


            $dIfOutOctets1 = mysql_result($rsInterface1,$i,'ifOutOctets') -
mysql_result($rsInterface1,$j,'ifOutOctets');
            $dIfOutOctets2 = mysql_result($rsInterface2,$i,'ifOutOctets') -
mysql_result($rsInterface2,$j,'ifOutOctets');
            $dIfOutOctets3 = mysql_result($rsInterface3,$i,'ifOutOctets') -
mysql_result($rsInterface3,$j,'ifOutOctets');
            $dIfOutOctets4 = mysql_result($rsInterface4,$i,'ifOutOctets') -
mysql_result($rsInterface4,$j,'ifOutOctets');


            if ($dTcpOutSegs1 == 0) {
                  $probRetransTcp1 = 0;
```

```
} else {
        $probRetransTcp1 = $dTcpRetransSegs1/$dTcpOutSegs1;
        }
if ($dTcpOutSegs2 == 0) {
        $probRetransTcp2 = 0;
} else {
        $probRetransTcp2 = $dTcpRetransSegs2/$dTcpOutSegs2;
        }
if ($dTcpOutSegs3 == 0) {
        $probRetransTcp3 = 0;
} else {
        $probRetransTcp3 = $dTcpRetransSegs3/$dTcpOutSegs3;
        }
if ($dTcpOutSegs4 == 0) {
        $probRetransTcp4 = 0;
} else {
        $probRetransTcp4 = $dTcpRetransSegs4/$dTcpOutSegs4;
        }

if ($dSysUpTime1 == 0) {
        $throuIp1 = 0;
        $throuUdp1 = 0;
        $throuTotal1 = 0;
} else {
        $throuIp1 =
($dIpInReceives1+$dIpOutRequests1+$dIpForwDatagrams1)/($dSysUpTime1/100);
        $throuUdp1 =
($dUdpInDatagrams1+$dUdpNoPorts1+$dUdpInErrors1+$dUdpOutDatagrams1)/($dSysUpTime1/
100);
        $throuTotal1 =
($dIfInOctets1+$dIfOutOctets1)*(8/1000)/($dSysUpTime1/100);
        }
if ($dSysUpTime2 == 0) {
        $throuIp2 = 0;
        $throuUdp2 = 0;
        $throuTotal2 = 0;
} else {
```

```
                $throuIp2 =
($dIpInReceives2+$dIpOutRequests2+$dIpForwDatagrams2)/($dSysUpTime2/100);
                $throuUdp2 =
($dUdpInDatagrams2+$dUdpNoPorts2+$dUdpInErrors2+$dUdpOutDatagrams2)/($dSysUpTime2/
100);
                $throuTotal2 =
($dIfInOctets2+$dIfOutOctets2)*(8/1000)/($dSysUpTime2/100);
                }
          if ($dSysUpTime3 == 0) {
                $throuIp3 = 0;
                $throuUdp3 = 0;
                $throuTotal3 = 0;
          } else {
                $throuIp3 =
($dIpInReceives3+$dIpOutRequests3+$dIpForwDatagrams3)/($dSysUpTime3/100);
                $throuUdp3 =
($dUdpInDatagrams3+$dUdpNoPorts3+$dUdpInErrors3+$dUdpOutDatagrams3)/($dSysUpTime3/
100);
                $throuTotal3 =
($dIfInOctets3+$dIfOutOctets3)*(8/1000)/($dSysUpTime3/100);
                }
          if ($dSysUpTime4 == 0) {
                $throuIp4 = 0;
                $throuUdp4 = 0;
                $throuTotal4 = 0;
          } else {
                $throuIp4 =
($dIpInReceives4+$dIpOutRequests4+$dIpForwDatagrams4)/($dSysUpTime4/100);
                $throuUdp4 =
($dUdpInDatagrams4+$dUdpNoPorts4+$dUdpInErrors4+$dUdpOutDatagrams4)/($dSysUpTime4/
100);
                $throuTotal4 =
($dIfInOctets4+$dIfOutOctets4)*(8/1000)/($dSysUpTime4/100);
                }

          if ($ifSpeed1 == 0) {
                $utilizLink1 = 0;
          } else {
                $utilizLink1 = ($throuTotal1 * 10)/$ifSpeed1;
```

```
                }
            if ($ifSpeed2 == 0) {
                $utilizLink2 = 0;
            } else {
                $utilizLink2 = ($throuTotal2 * 10)/$ifSpeed2;
                }
            if ($ifSpeed3 == 0) {
                $utilizLink3 = 0;
            } else {
                $utilizLink3 = ($throuTotal3 * 10)/$ifSpeed3;
                }
            if ($ifSpeed4 == 0) {
                $utilizLink4 = 0;
            } else {
                $utilizLink4 = ($throuTotal4 * 10)/$ifSpeed4;
                }
            $horario = mysql_result($rsInterface1,$i,'horario');
            $query1 = "INSERT INTO indicadores
(maquinaid,horario,prob_retrans_tcp,throu_ip,throu_udp,throu_total,utiliz_link)
VALUES
('1','$horario','$probRetransTcp1','$throuIp1','$throuUdp1','$throuTotal1','$utili
zLink1')";
            mysql_query($query1,$localhost) or die(mysql_error());
            $query2 = "INSERT INTO indicadores
(maquinaid,horario,prob_retrans_tcp,throu_ip,throu_udp,throu_total,utiliz_link)
VALUES
('2','$horario','$probRetransTcp2','$throuIp2','$throuUdp2','$throuTotal2','$utili
zLink2')";
            mysql_query($query2,$localhost) or die(mysql_error());
            $query3 = "INSERT INTO indicadores
(maquinaid,horario,prob_retrans_tcp,throu_ip,throu_udp,throu_total,utiliz_link)
VALUES
('3','$horario','$probRetransTcp3','$throuIp3','$throuUdp3','$throuTotal3','$utili
zLink3')";
            mysql_query($query3,$localhost) or die(mysql_error());
            $query4 = "INSERT INTO indicadores
(maquinaid,horario,prob_retrans_tcp,throu_ip,throu_udp,throu_total,utiliz_link)
VALUES
('4','$horario','$probRetransTcp4','$throuIp4','$throuUdp4','$throuTotal4','$utili
zLink4')";
```

```
            mysql_query($query4,$localhost) or die(mysql_error());
        }
        $proxColuna = $totalRows + 1;
        $query_proxColuna = "UPDATE aux SET prox_coluna=$proxColuna";
        mysql_query($query_proxColuna,$localhost) or die(mysql_error());
?>
```

# ANEXO II

Este anexo contém a modelagem dos bancos de dados utilizados na realização do trabalho.

- Bancos de dados que serviram como fonte de dados.

```
-- Tabelas utilizadas para armazenar objetos SNMP.

CREATE TABLE maquinas
(
id mediumint(8) unsigned NOT NULL auto_increment,
nome varchar(200) NOT NULL default '',
ip varchar(200),
interface varchar(200),
tipo varchar(200),
campus varchar(200),
predio varchar(200),
departamento varchar(200),
sala varchar(200),
os varchar(200),
os_versao varchar(200),
PRIMARY KEY (id)
);

INSERT INTO maquinas values (1,'Email','192.168.67.213','65539','servidor','darcy
ribeiro','ft','ene','labredes','windows','server 2003 enterprise x64');
INSERT INTO maquinas values (2,'DNS','192.168.67.201','65540','servidor','darcy
ribeiro','ft','ene','labredes','windows','server 2003 r2 standard');
INSERT INTO maquinas values (3,'DHCP','192.168.67.5','9','servidor','darcy
ribeiro','ft','ene','labredes','windows','server 2008 enterprise');
INSERT INTO maquinas values (4,'Web','192.168.67.204','65540','servidor','darcy
ribeiro','ft','ene','labredes','windows','server 2003 r2 standard');

CREATE TABLE interface
(
id mediumint(8) unsigned NOT NULL auto_increment,
maquinaid int,
horario DATETIME,
sysUpTime int,
ifSpeed int,
ifInOctets int,
ifInErrors int,
ifOutOctets int,
ifOutErrors int,
ifMtu int,
PRIMARY KEY (id)
);
```

```
CREATE TABLE ip
(
id mediumint(8) unsigned NOT NULL auto_increment,
maquinaid int,
horario DATETIME,
sysUpTime int,
ipReasmReqds int,
ipReasmFails int,
ipInReceives int,
ipOutRequests int,
ipFragOKs int,
ipFragFails int,
ipForwDatagrams int,
PRIMARY KEY (id)
);

CREATE TABLE tcp
(
id mediumint(8) unsigned NOT NULL auto_increment,
maquinaid int,
horario DATETIME,
sysUpTime int,
tcpRetransSegs int,
tcpOutSegs int,
PRIMARY KEY (id)
);

CREATE TABLE udp
(
id mediumint(8) unsigned NOT NULL auto_increment,
maquinaid int,
horario DATETIME,
sysUpTime int,
udpInDatagrams int,
udpOutDatagrams int,
udpInErrors int,
udpNoPorts int,
PRIMARY KEY (id)
);
```

- Bancos de dados utilizados no processo de ETL.

```
-- Tabelas utilizadas para o armazenamento dos indicadores de gerência.

CREATE TABLE aux (
  prox_coluna int(11) NOT NULL
);

INSERT INTO aux VALUES (0);
```

```
CREATE TABLE indicadores (
  id int(11) NOT NULL auto_increment,
  maquinaid int(11) NOT NULL,
  horario datetime NOT NULL,
  prob_retrans_tcp float NOT NULL,
  throu_ip float NOT NULL,
  throu_udp float NOT NULL,
  throu_total float NOT NULL,
  utiliz_link float NOT NULL,
  PRIMARY KEY  (id)
);

-- Criação do Operational Data Store

CREATE TABLE ods (
  id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  maquinaid INTEGER UNSIGNED NULL,
  horario DATETIME NULL,
  prob_retrans_tcp FLOAT NULL,
  throu_ip FLOAT NULL,
  throu_udp FLOAT NULL,
  throu_total FLOAT NULL,
  utiliz_link FLOAT NULL,
  os VARCHAR(200) NULL,
  PRIMARY KEY(id)
);

-- Extração, transformação e carga:

INSERT INTO ods
(maquinaid,horario,prob_retrans_tcp,throu_ip,throu_udp,throu_total,utiliz_link,os)
SELECT
i.maquinaid,i.horario,i.prob_retrans_tcp,i.throu_ip,i.throu_udp,i.throu_total,i.ut
iliz_link,i.maquinaid
FROM
indicadores i;

UPDATE ods SET os=2 WHERE maquinaid=4;

INSERT INTO dw_gerencia.dim_tempo
(ano,mes,semana,dia_mes,dia_semana,hora,minuto,horario)
SELECT
year(horario),month(horario),week(horario),dayofmonth(horario),dayofweek(horario),
hour(horario),minute(horario),horario
FROM projeto.indicadores
GROUP BY dayofyear(horario),hour(horario),minute(horario);

UPDATE dw_gerencia.dim_tempo SET dia_semana='1 Domingo' WHERE dia_semana='1';
UPDATE dw_gerencia.dim_tempo SET dia_semana='2 Segunda-Feira' WHERE
dia_semana='2';
UPDATE dw_gerencia.dim_tempo SET dia_semana='3 Terca-Feira' WHERE dia_semana='3';
```

```
UPDATE dw_gerencia.dim_tempo SET dia_semana='4 Quarta-Feira' WHERE dia_semana='4';
UPDATE dw_gerencia.dim_tempo SET dia_semana='5 Quinta-Feira' WHERE dia_semana='5';
UPDATE dw_gerencia.dim_tempo SET dia_semana='6 Sexta-Feira' WHERE dia_semana='6';
UPDATE dw_gerencia.dim_tempo SET dia_semana='7 Sabado' WHERE dia_semana='7';
UPDATE dw_gerencia.dim_tempo SET mes='Dezembro' WHERE mes='12';

INSERT INTO dw_gerencia.dim_equipamento (pk_equipamento, tipo, nome, ip,
interface)
SELECT id, tipo, nome, ip, interface
FROM projeto.maquinas;

INSERT INTO dw_gerencia.dim_localidade (campus,predio,departamento,sala)
SELECT campus,predio,departamento,sala
FROM projeto.maquinas
GROUP BY departamento,sala;

INSERT INTO dw_gerencia.dim_os (nome,versao)
SELECT os, os_versao
FROM projeto.maquinas
GROUP BY os,os_versao;

INSERT INTO dw_gerencia.fat_desempenho
(fk_equipamento,fk_tempo,fk_os,fk_localidade,prob_retrans_tcp,throu_ip,throu_udp,t
hrou_total,utiliz_link)
SELECT od.maquinaid AS "pk_equipamento", e.pk_tempo, od.os AS pk_os, 1 AS
"pk_localidade", od.prob_retrans_tcp*100, od.throu_ip, od.throu_udp,
od.throu_total, od.utiliz_link*100
FROM projeto.ods od, dw_gerencia.dim_tempo e
WHERE od.horario=e.horario;
```

- Bancos de dados utilizados como DW.

```
-- Criação das tabelas dimensão e fato do Data Warehouse.

CREATE TABLE dim_os (
  pk_os INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  nome VARCHAR(200) NULL,
  versao VARCHAR(200) NULL,
  PRIMARY KEY(pk_os)
) ENGINE=InnoDB;

CREATE TABLE dim_tempo (
  pk_tempo INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  ano INTEGER UNSIGNED NULL,
  mes INTEGER UNSIGNED NULL,
  semana INTEGER UNSIGNED NULL,
  dia_mes INTEGER UNSIGNED NULL,
  dia_semana VARCHAR(200) NULL,
  hora INTEGER UNSIGNED NULL,
  minuto INTEGER UNSIGNED NULL,
  horario DATETIME NULL,
```

```
  PRIMARY KEY(pk_tempo)
) ENGINE=InnoDB;

CREATE TABLE dim_equipamento (
  pk_equipamento INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  tipo VARCHAR(200) NULL,
  nome VARCHAR(200) NULL,
  ip VARCHAR(200) NULL,
  interface INTEGER UNSIGNED NULL,
  PRIMARY KEY(pk_equipamento)
) ENGINE=InnoDB;

CREATE TABLE dim_localidade (
  pk_localidade INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  campus VARCHAR(200) NULL,
  predio VARCHAR(200) NULL,
  departamento VARCHAR(200) NULL,
  sala VARCHAR(200) NULL,
  PRIMARY KEY(pk_localidade)
) ENGINE=InnoDB;

CREATE TABLE fat_desempenho (
  pk_desempenho INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  fk_localidade INTEGER UNSIGNED NOT NULL,
  fk_os INTEGER UNSIGNED NOT NULL,
  fk_tempo INTEGER UNSIGNED NOT NULL,
  fk_equipamento INTEGER UNSIGNED NOT NULL,
  prob_retrans_tcp FLOAT NULL,
  throu_ip FLOAT NULL,
  throu_udp FLOAT NULL,
  throu_total FLOAT NULL,
  utiliz_link FLOAT NULL,
  PRIMARY KEY(pk_desempenho),
  FOREIGN KEY(fk_equipamento)
    REFERENCES dim_equipamento(pk_equipamento)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,
  FOREIGN KEY(fk_tempo)
    REFERENCES dim_tempo(pk_tempo)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,
  FOREIGN KEY(fk_os)
    REFERENCES dim_os(pk_os)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,
  FOREIGN KEY(fk_localidade)
    REFERENCES dim_localidade(pk_localidade)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION
) ENGINE=InnoDB;
```

# ANEXO III

Esse anexo contém o arquivo .xml e .xaction que modelam a apresentação dos dados no Pentaho.

- analysis_gerencia.xaction

```
<?xml version="1.0" encoding="UTF-8"?>
<action-sequence>
  <name>analysis_gerencia.xaction</name>
  <title>%title</title>
  <version>1</version>
  <logging-level>ERROR</logging-level>
  <documentation>
    <author>Catarina e Júnior</author>
    <help/>
    <result-type>report</result-type>
    <description>%description</description>
    <icon>analysis_productline.png</icon>
  </documentation>

  <inputs>
    <mode type="string">
      <default-value/>
      <sources>
        <request>mode</request>
      </sources>
    </mode>
  </inputs>

  <outputs>
    <model type="string"/>
    <connection type="string"/>
    <mdx type="string"/>
    <options type="list"/>
    <title type="string"/>
    <url type="string">
      <destinations>
        <response>redirect</response>
      </destinations>
    </url>
    <charttype type="string"/>
    <chartlocation type="string"/>
  </outputs>

  <resources/>

  <actions>
    <action-definition>
```

```
<component-name>PivotViewComponent</component-name>
<action-type>Pivot View</action-type>
<action-inputs>
  <mode type="string"/>
</action-inputs>
<action-outputs>
  <model type="string"/>
  <connection type="string"/>
  <mdx type="string"/>
  <options type="list"/>
  <title type="string"/>
  <url type="string"/>
  <charttype type="string"/>
  <chartlocation type="string"/>
</action-outputs>
<component-definition>
  <title>Drill Down to Pivot Table</title>
  <viewer>Pivot</viewer>
  <model><![CDATA[gerencia/Analises/gerencia.mondrian.xml]]></model>
  <charttype>16</charttype>
  <chartlocation>top</chartlocation>
  <connection>jdbc/gerencia</connection >
  <!--query>default</query-->
  <options>
    <personal/>
    <cube-nav/>
    <mdx-edit/>
    <sort-conf/>
    <spacer/>
    <level-style/>
    <hide-spans/>
    <properties/>
    <non-empty/>
    <swap-axes/>
    <spacer/>
    <drill-member/>
    <drill-position/>
    <drill-replace/>
    <drill-thru/>
    <spacer/>
    <chart/>
    <chart-conf/>
    <spacer/>
    <print-conf/>
    <print-pdf/>
    <spacer/>
    <excel/>
  <jndi>gerencia</jndi>
  <query><![CDATA[
```

```
select NON EMPTY {[Measures].[Prob Retransmissao TCP (%)], [Measures].[Throughput
Ip (dtg/s)], [Measures].[Throughput Udp (dtg/s)], [Measures].[Throughput Total
(Kbps)], [Measures].[Utilizacao do Link (%)]} ON COLUMNS,
  NON EMPTY Crossjoin({[Equipamento].[Todos Equipamentos].[servidor].[DHCP],
[Equipamento].[Todos Equipamentos].[servidor].[DNS], [Equipamento].[Todos
Equipamentos].[servidor].[Email], [Equipamento].[Todos
Equipamentos].[servidor].[Web]}, {([Tempo].[Todos Tempos],
[SistemaOperacional].[Todos SOs].[windows], [Localidade].[Todas
Localidades].[darcy ribeiro].[ft].[ene].[labredes])}) ON ROWS
from [gerencia]

]]></query>
      </component-definition>
      <action-name>Pivot View</action-name>
      <logging-level>DEBUG</logging-level>
    </action-definition>

  </actions>
</action-sequence>
```

- gerencia.mondrian.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Schema name="gerencia">
  <Cube name="gerencia" cache="true" enabled="true">
    <Table name="fat_desempenho">
    </Table>
    <Dimension foreignKey="fk_equipamento" name="Equipamento">
      <Hierarchy hasAll="true" allMemberName="Todos Equipamentos"
primaryKey="pk_equipamento">
        <Table name="dim_equipamento">
        </Table>
        <Level name="Tipo" column="tipo" type="String" uniqueMembers="true"
levelType="Regular" hideMemberIf="Never">
        </Level>
        <Level name="Nome" column="nome" type="String" uniqueMembers="true"
levelType="Regular" hideMemberIf="Never">
        </Level>
        <Level name="Endereco Ip" column="ip" type="String" uniqueMembers="true"
levelType="Regular" hideMemberIf="Never">
        </Level>
        <Level name="Interface" column="interface" type="String"
uniqueMembers="true" levelType="Regular" hideMemberIf="Never">
        </Level>
      </Hierarchy>
    </Dimension>

    <Dimension foreignKey="fk_tempo" name="Tempo">
      <Hierarchy hasAll="true" allMemberName="Todos Tempos" primaryKey="pk_tempo">
        <Table name="dim_tempo">
```

```
        </Table>
        <Level name="Ano" column="ano" type="String" uniqueMembers="true"
levelType="Regular" hideMemberIf="Never">
        </Level>
        <Level name="Mes" column="mes" type="String" uniqueMembers="true"
levelType="Regular" hideMemberIf="Never">
        </Level>
        <!--Level name="Semana" column="semana" type="String" uniqueMembers="true"
levelType="Regular" hideMemberIf="Never">
        </Level-->
        <Level name="Dia da Semana" column="dia_semana" type="String"
uniqueMembers="true" levelType="Regular" hideMemberIf="Never">
        </Level>
        <Level name="Dia do Mes" column="dia_mes" type="String"
uniqueMembers="true" levelType="Regular" hideMemberIf="Never">
        </Level>
        <Level name="Hora" column="hora" type="String" uniqueMembers="true"
levelType="Regular" hideMemberIf="Never">
        </Level>
        <Level name="Minuto" column="minuto" type="String" uniqueMembers="true"
levelType="Regular" hideMemberIf="Never">
        </Level>
      </Hierarchy>
    </Dimension>

    <Dimension foreignKey="fk_os" name="SistemaOperacional">
      <Hierarchy hasAll="true" allMemberName="Todos SOs" primaryKey="pk_os">
        <Table name="dim_os">
        </Table>
        <Level name="Nome" column="nome" type="String" uniqueMembers="true"
levelType="Regular" hideMemberIf="Never">
        </Level>
        <Level name="Versao" column="versao" type="String" uniqueMembers="true"
levelType="Regular" hideMemberIf="Never">
        </Level>
      </Hierarchy>
    </Dimension>

    <Dimension foreignKey="fk_Localidade" name="Localidade">
      <Hierarchy hasAll="true" allMemberName="Todas Localidades"
primaryKey="pk_localidade">
        <Table name="dim_localidade">
        </Table>
        <Level name="Campus" column="campus" type="String" uniqueMembers="true"
levelType="Regular" hideMemberIf="Never">
        </Level>
        <Level name="Predio" column="predio" type="String" uniqueMembers="true"
levelType="Regular" hideMemberIf="Never">
        </Level>
        <Level name="Departamento" column="departamento" type="String"
uniqueMembers="true" levelType="Regular" hideMemberIf="Never">
```

```
        </Level>
        <Level name="Sala" column="sala" type="String" uniqueMembers="true"
levelType="Regular" hideMemberIf="Never">
        </Level>
      </Hierarchy>
    </Dimension>

    <Measure name="Prob Retransmissao TCP (%)" column="prob_retrans_tcp"
formatString="0.###" aggregator="avg">
    </Measure>
    <Measure name="Throughput Ip (dtg/s)" column="throu_ip" formatString="#.###"
aggregator="avg">
    </Measure>
    <Measure name="Throughput Udp (dtg/s)" column="throu_udp" formatString="#.###"
aggregator="avg">
    </Measure>
    <Measure name="Throughput Total (Kbps)" column="throu_total"
formatString="#.###" aggregator="avg">
    </Measure>
    <Measure name="Utilizacao do Link (%)" column="utiliz_link"
formatString="0.######" aggregator="avg">
    </Measure>
  </Cube>
</Schema>
```