

**Pontifícia Universidade Católica do Rio Grande do Sul**  
**Faculdade de Informática**  
**Curso de Bacharelado em Ciência da Computação**

DIONES F. ROSSETTO  
LUIZ H. A. MELLO

**Desview — Visualização de Informações de  
Desempenho em Redes de Computadores**

Orientadora: Profa. Dra. Isabel H. Manssour

**Trabalho de Conclusão de Curso**

Porto Alegre, 10 de julho de 2010.



## **Resumo**

O contínuo crescimento, em número e diversidade, dos componentes de uma rede de computadores e da necessidade de processamento das informações geradas, tem tornado a atividade de gerenciamento de redes de computadores cada vez mais complexa. Em resposta a esta necessidade, pode-se utilizar técnicas de Visualização de Informações que ajudem a explorar e analisar os dados gerados auxiliando o gerenciamento dessas redes. Este Trabalho de Conclusão tem como principal objetivo o projeto e o desenvolvimento de um sistema de Visualização de Informações de desempenho no gerenciamento de Redes de Computadores. O sistema permite a visualização de dados de desempenho de equipamentos de uma rede de computadores por meio de gráficos 2D e 3D utilizando dados coletados da rede em diferentes componentes que compõem a sua infraestrutura.

Palavras-chaves: Visualização de Informações, Gerência de Redes de Computadores.

## **Abstract**

The continuous growth, in number and diversity, of the components of a computer network and the information processing needs turned its management increasingly complex. In response to these needs, Information Visualization techniques may be used to explore and analyze the generated information in the network management context. The main goal of this Final Work is the design and development of a Network Performance Management system using Information Visualization. The system provides 2D and 3D graphics to visualize performance data of a computer network using data collected from the network of different components that make up its infrastructure.

Keywords: Information Visualization, Computer Networks Management.

# Sumário

<b>LISTA DE FIGURAS</b>	<b>3</b>
<b>LISTA DE TABELAS</b>	<b>5</b>
<b>LISTA DE SÍMBOLOS E ABREVIATURAS</b>	<b>6</b>
<b>1 Introdução</b>	<b>8</b>
1.1 Contextualização . . . . .	8
1.2 Motivação . . . . .	9
1.3 Objetivos . . . . .	9
1.4 Organização do texto . . . . .	10
<b>2 Fundamentação Teórica</b>	<b>11</b>
2.1 Visualização de Informações . . . . .	11
2.1.1 Introdução . . . . .	11
2.1.2 Dados e Representações Visuais . . . . .	11
2.1.3 Modelo de Referência de Visualização . . . . .	12
2.1.4 Vantagens . . . . .	13
2.1.5 Aplicações . . . . .	15
2.1.6 Técnicas . . . . .	15
2.2 Gerência de Redes de Computadores . . . . .	19
2.2.1 Introdução . . . . .	19
2.2.2 Arquitetura de Gerenciamento . . . . .	20
2.2.3 MIB . . . . .	21
2.2.4 Protocolo SNMP . . . . .	23
2.2.5 Gerenciamento de Desempenho . . . . .	26
<b>3 Trabalhos Relacionados</b>	<b>28</b>
3.1 Cacti . . . . .	28
3.2 MRTG . . . . .	29
3.3 Nagios . . . . .	30
3.4 OpenNMS . . . . .	32
3.5 Ferramenta <i>Management Traffic Analyzer</i> . . . . .	33

3.6	Ferramenta Prisma . . . . .	34
3.7	Visualização e Análise de Dados de Monitoramento de Sistemas usando Multi-resolução . . . . .	35
<b>4</b>	<b>Descrição do Desview</b>	<b>38</b>
4.1	Introdução . . . . .	38
4.2	Ambiente de desenvolvimento . . . . .	38
4.3	Modelagem do protótipo . . . . .	40
4.4	Interface e funcionalidades . . . . .	44
4.5	Implementação . . . . .	45
4.5.1	Infraestrutura do sistema . . . . .	46
4.5.2	Visualizações . . . . .	46
4.6	Estudo de caso . . . . .	48
<b>5</b>	<b>Conclusões e trabalhos futuros</b>	<b>53</b>

# Listas de Figuras

2.1	Modelo clássico de visualização [10]. . . . .	14
2.2	Modelo de referência de visualização [1]. . . . .	14
2.3	Imagen gerada com a utilização da técnica de <i>treemap</i> , retirada de RDI [23]. . . . .	17
2.4	Imagen gerada utilizando a técnica de dispersão, retirada de RDI [23]. . . . .	17
2.5	Exemplo de gráfico com coordenadas paralelas, retirado de RDI [23]. . . . .	18
2.6	Integração entre as áreas funcionais de Gerência de Redes [3]. . . . .	20
2.7	A MIB-2. . . . .	22
3.1	Exemplo de gráfico gerado pelo Cacti [5]. . . . .	29
3.2	Imagens geradas pelo MRTG, retiradas de Switch [22]. . . . .	31
3.3	Exemplo de gráfico gerado pelo Nagios [6]. . . . .	32
3.4	Exemplo de gráfico gerado pelo OpenNMS. . . . .	33
3.5	Gráfico gerado pela ferramenta <i>Management Traffic Analyzer</i> [7]. . . . .	34
3.6	Gráfico com as três técnicas utilizadas no Prisma ( <i>treemap</i> , dispersão e coordenadas paralelas) para o mesmo conjunto de dados de entrada [8]. . . . .	35
3.7	Exemplo de um Mapa Auto-Organizável retirado de Germano [31]. O mapa bi-dimensional gerado representa as posições dos nodos em relação aos nodos vizinhos. O objetivo é que os nodos que forem topologicamente próximos respondam de maneira semelhante a entradas semelhantes. . . . .	36
3.8	Mapeamento de dados de entrada 6-D em 2-D através de Mapas Auto-Organizáveis, onde os dados de entrada correspondem a 6 diferentes vetores com apenas um número ‘1’ em cada vetor. Figura retirada de Nayak [27]. . . . .	37
4.1	Casos de uso do sistema. . . . .	41
4.2	Modelagem do banco de dados do sistema. . . . .	43
4.3	Tela principal do protótipo. . . . .	44
4.4	Tela de inserção de equipamentos. . . . .	45
4.5	Tela de inserção de tarefas e de variáveis associadas. . . . .	45
4.6	Gráfico de tempo real obtido, informando que a variável em monitoração passou do <i>threshold</i> informado. . . . .	48
4.7	Gráfico polar obtido a partir de leituras efetuadas em uma variável durante um mês. . . . .	49

4.8	Gráfico de barras contendo o valor mínimo, o valor máximo e a média dos valores lidos de três variáveis.	49
4.9	Gráfico de linhas contendo a média de todos os valores lidos durante os meses de determinado ano.	50
4.10	Gráfico OpenGL 2D - linhas.	51
4.11	Gráfico OpenGL 2D - espirais.	51
4.12	Gráfico OpenGL 3D - esferas nível de variáveis distribuídas nos meses do ano.	52
4.13	Gráfico OpenGL 3D - esferas mostrando os dias do mês para a variável selecionada.	52
5.1	Modelo conceitual de classes do sistema	74
5.2	Modelo conceitual de classes: parte 1 - pacote <i>model</i> .	75
5.3	Modelo conceitual de classes: parte 2 - pacote <i>controller</i> .	76
5.4	Modelo conceitual de classes: parte 3 - pacote <i>util</i> .	77
5.5	Modelo conceitual de classes: parte 4 - pacote <i>graphics</i> .	78
5.6	Modelo conceitual de classes: parte 5 - pacote <i>charts</i> .	79
5.7	Modelo conceitual de classes: parte 6 - pacote <i>view</i> .	80
5.8	Tela de <i>login</i> do sistema.	81
5.9	Tela inicial do sistema com tarefas e visualizações.	82
5.10	Inserindo equipamentos e tarefas.	82
5.11	Inserção de equipamentos.	83
5.12	Inserção de tarefas e variáveis.	83
5.13	Definição de <i>thresholds</i> de variáveis.	84
5.14	Edição de equipamentos e tarefas.	84
5.15	Remover de equipamentos e tarefas.	85
5.16	Janela de escolha de equipamento a ser deletado.	85
5.17	Botões de ações para as tarefas.	85
5.18	Tela de escolha de formatos de exportação.	86
5.19	Tela de escolha de visualizações a serem plotadas.	87
5.20	Janela de escolha de variáveis.	87
5.21	Exemplo de gráfico de tempo real gerado.	88
5.22	Janela de seleção de tarefas para gráficos OpenGL.	88
5.23	Exemplo de gráfico de linhas e de espirais gerados.	89
5.24	Exemplo 1 de gráfico em esferas.	89
5.25	Exemplo 2 de gráfico em esferas.	90
5.26	Nível de dias de uma tarefa selecionada.	90

# **Lista de Tabelas**

2.1	Sumário da caracterização dos dados com exemplos de diferentes domínios [9].	12
2.2	Classes de representações visuais [9].	13
2.3	Áreas FCAPS dos grupos da subárvore <i>mib-2</i> .	23
2.4	Exemplos de objetos da <i>mib-2</i> e seus respectivos grupos.	24

# Listas de Símbolos e Abreviaturas

VISINFO	Visualização de Informação	8
CG	Computação Gráfica	8
GRC	Gerência de Redes de Computadores	8
TC	Trabalho de Conclusão	10
ISO	<i>International Organization for Standardization</i>	19
MIB	<i>Management Information Base</i>	19
SMI	<i>Structure Management Information</i>	20
TCP	<i>Transmission Control Protocol</i>	21
UDP	<i>User Datagram Protocol</i>	21
SNMP	<i>Simple Network Management Protocol</i>	21
RFC	<i>Request for Comments</i>	21
IAB	<i>Internet Activities Board</i>	21
ASN.1	<i>Abstract Syntax Notation.1</i>	21
OID	<i>Object IDentifier</i>	21
CCITT	Comitê Consultivo Internacional de Telegrafia e Telefonia	21
DOD	<i>Departament of Defense of USA</i>	22
OSI	<i>Open Systems Interconnection</i>	22
TCP/IP	<i>Transmission Control Protocol/Internet Protocol</i>	23

IP	<i>Internet Protocol</i>	<b>25</b>
CPU	<i>Central Processing Unit</i>	<b>28</b>
RRD	<i>Round Robin Database</i>	<b>28</b>
MRTG	<i>Multi Router Traffic Grapher</i>	<b>29</b>
HTML	<i>HyperText Markup Language</i>	<b>29</b>
GPL	<i>GNU General Public License</i>	<b>30</b>
ICMP	<i>Internet Control Message Protocol</i>	<b>30</b>
HTTP	<i>Hypertext Transfer Protocol</i>	<b>30</b>
UML	<i>Unified Modeling Language</i>	<b>39</b>
MVC	<i>Model-view-controller</i>	<b>40</b>
DAO	<i>Data Access Object</i>	<b>40</b>
XML	<i>Extensible Markup Language</i>	<b>42</b>
PDF	<i>Portable Document Format</i>	<b>46</b>

# 1 Introdução

Neste capítulo é feita uma contextualização do trabalho desenvolvido, além de apresentar a sua motivação e os objetivos do trabalho. Ao final do capítulo é feita uma descrição da organização do texto.

## 1.1 Contextualização

A Visualização de Informação (*VISINFO*) é uma área de aplicação da Computação Gráfica (*CG*) que se utiliza de diferentes técnicas e algoritmos com o objetivo de facilitar a análise de um grande volume de dados, gerados por diversas fontes, através de representações gráficas manipuláveis. A representação visual destes volumes de dados visa facilitar o seu entendimento e tem se mostrado de grande importância nos últimos anos, para diversas organizações que necessitam analisar e extrair informações dos diversos dados gerados. De acordo com Card, Mackinlay e Shneiderman [1], a *VISINFO* consiste na utilização de representações visuais e interativas de dados abstratos, suportadas por computador, para ampliar a cognição e encontrar relações e características presentes em um conjunto de dados.

A *VISINFO* tem por objetivo facilitar o entendimento das informações proporcionando uma representação visual de um conjunto de dados de uma forma mais simples e intuitiva, facilitando o entendimento do significado e das relações entre os dados pertencentes a um conjunto. Torna-se muito mais fácil para uma pessoa entender o significado de uma imagem do que de vários dados isolados [2].

A Gerência de Rede de Computadores (*GRC*) é uma área de redes de computadores que se preocupa em garantir que um determinado conjunto de ações, procedimentos e políticas sejam tomados para manter sempre ativa uma determinada rede, pelo fato da importância que as redes têm desenvolvido nos últimos anos e ao porte e à complexidade que elas têm adquirido para a maioria das organizações. Sem um controle efetivo, os recursos das redes de computadores (que normalmente são heterogêneos), não proporcionariam os retornos desejados para os quais foram concebidos. Dessa forma, alguns dos principais objetivos da área de *GRC* podem ser definidos como: garantir a qualidade da infraestrutura de rede; identificar padrões e tendências de uso; planejar o crescimento da rede e identificar e resolver problemas o mais breve possível [3].

## 1.2 Motivação

A visualização possui um importante papel no entendimento, exploração e análise de um conjunto de dados, porém, no contexto de gerenciamento de infraestrutura de sistemas, a maioria das ferramentas disponíveis de visualização é limitada a informações tabulares e linhas de tempo informando eventos ocorridos. Como o número de dispositivos sendo monitorados tende a crescer e a infraestrutura que os comporta torna-se cada vez mais complexa, é importante que sejam implementadas ferramentas com visualizações mais efetivas e que explorem técnicas de análise de dados no gerenciamento de sistemas.

Sistemas de GRC que utilizam técnicas de VISINFO para representar e organizar as informações de redes têm sido desenvolvidos em áreas como topologia de redes e gerenciamento de falhas ocorridas em elementos de rede, tais como o sistema OpenNMS [37] (apresentado no capítulo 3). Como a VISINFO permite a criação de abordagens abstratas e intuitivas para transmitir as informações de forma que os usuários explorem e compreendam muitas informações de uma só vez, pode-se desenvolver uma ferramenta que seja capaz de prover essa funcionalidade no gerenciamento de desempenho de redes de computadores também. Essa representação visual é desejável porque, por um lado abstrai detalhes do conjunto de informações e, por outro, propicia uma organização desse conjunto de dados segundo algum critério [9]. O uso de uma ferramenta visual que apresente um grande conjunto de informações pode auxiliar os administradores de redes a se anteciparem a possíveis problemas que venham a ocorrer na rede, aumentando o desempenho da rede e, consequentemente, a confiabilidade da mesma.

## 1.3 Objetivos

O objetivo geral deste trabalho é o projeto e o desenvolvimento de um sistema de VISINFO de desempenho no gerenciamento de redes de computadores. Este sistema permite a visualização, através do uso de técnicas de VISINFO, do desempenho de uma rede de computadores por meio de gráficos 2D e 3D. Além disso, o protótipo desenvolvido realiza a coleta e o armazenamento dos dados de variáveis de equipamentos de rede para visualização dos dados em tempo real, ou então, para posterior análise através de gráficos dos valores armazenados das leituras.

Assim, o trabalho une as práticas e os conhecimentos adquiridos ao longo da faculdade nas disciplinas de Redes de Computadores, Engenharia de *Software* e CG, em uma ferramenta para a resolução de inúmeros problemas. Assim, o sistema descrito permite auxiliar o administrador da rede a antecipar-se a problemas que venham a ocorrer ou estejam ocorrendo em uma rede, além da análise dos dados para, então, com o auxílio de representações gráficas simples e/ou mais elaboradas, encontrar relações entre os dados e, assim, tomar atitudes que possam auxiliar o desenvolvimento de uma rede de computadores e evitar a degradação do sistema.

## **1.4 Organização do texto**

O presente trabalho de conclusão (*TC*) está dividido em cinco capítulos estruturados da seguinte forma: o Capítulo 1 apresenta uma introdução, com a contextualização das áreas de VISINFO e GRC, motivação e objetivos do trabalho; o Capítulo 2 apresenta a fundamentação teórica sobre a área de VISINFO e GRC; o Capítulo 3 descreve trabalhos relacionados com este trabalho; o Capítulo 4 , por sua vez, apresenta maiores detalhes relativos ao trabalho implementado, como ambiente de desenvolvimento, funcionalidades, escopo e modelagem do sistema e o Capítulo 5 apresenta as conclusões obtidas com o trabalho desenvolvido e trabalhos futuros.

# **2 Fundamentação Teórica**

Neste capítulo é apresentado o referencial teórico relativo às áreas relacionadas ao presente trabalho. É feita uma introdução e são abordadas características das áreas de VISINFO e de GRC. São mencionadas aplicações, modelos de referência, vantagens e técnicas utilizadas para o desenvolvimento de aplicações com VISINFO. Também são apresentadas as áreas funcionais de GRC, a arquitetura de gerenciamento, além de uma breve introdução ao protocolo SNMP e ao gerenciamento de desempenho.

## **2.1 Visualização de Informações**

### **2.1.1 Introdução**

A VISINFO é uma área emergente da Ciência da Computação que estuda técnicas para apresentar dados e informações de forma visual, de tal modo que as relações entre esses dados sejam mais bem compreendidas ou informações novas sejam descobertas. O principal objetivo do uso de técnicas de VISINFO é auxiliar no entendimento de um determinado assunto e a partir de um conjunto de dados alfanuméricos gerar representações gráficas. Sem a utilização de técnicas de visualização para a análise de certo conjunto de informações, seria necessário um esforço muito maior para compreender e relacionar dados entre si [4].

Dessa forma, a área de VISINFO se apresenta como um campo de estudo de grande utilidade, uma vez que agrupa técnicas que facilitam o entendimento de informações a partir da representação visual de vários conjuntos de dados [4].

### **2.1.2 Dados e Representações Visuais**

Dados descrevem fenômenos (ou processos) que são objeto de estudo ou análise. Dessa forma, dados podem ser caracterizados de acordo com diferentes critérios e a identificação dessas características é a consideração inicial a ser feita na escolha da técnica de visualização a ser utilizada [9]. Um primeiro critério para caracterizar atributos é a classe (ou tipo) de informação que representam. Atributos podem representar propriedades ou relações entre os dados. Um segundo critério de caracterização dos atributos, dependente do primeiro, é quanto ao tipo de dado que representa a identificação de entidade relacionada. Finalmente, um terceiro critério para a caracterização é de acordo com a dimensão e a natureza do

domínio onde estão representados. Um sumário da caracterização dos dados e exemplos de domínios podem ser observados na Tabela 2.1, retirada de Freitas [9].

Tabela 2.1: Sumário da caracterização dos dados com exemplos de diferentes domínios [9].

Critério	Classe	Exemplo
Classe de Informação	Escalar Vetorial Tensorial Relacionamento	Temperatura. Grandezas físicas associadas à dinâmica de fluidos. <i>Links</i> em um hiperdocumento.
Tipos dos valores	Alfanumérico Numérico (inteiro, real) Simbólico	Valor de identificação. Temperatura. <i>Links</i> em um hiperdocumento.
Natureza do domínio	Discreto Contínuo Contínuo-discretizado	Marcas de automóveis. Superfície de um terreno. Anos (tempo discretizado).
Natureza do domínio	1D 2D 3D $n$ -D	Fenômeno ocorrendo no tempo. Superfície de um terreno. Volume de dados médicos. Dados de uma população.

Segundo [9], “representações gráficas ou visuais correspondem às ‘figuras’ ou ‘imagens’ empregadas para representar o conjunto de dados em análise. Além dos tradicionais gráficos de pontos, de linhas, de tortas, de barras e histogramas de frequência, os quais nos permitem observar relações entre atributos, uma série de representações gráficas mais ou menos complexas são empregadas para codificar através de elementos visuais tanto valores como relacionamentos entre entidades ou elementos de dados”. Um resumo dessas representações pode ser observado na Tabela 2.2, retirada de Freitas [9].

### 2.1.3 Modelo de Referência de Visualização

Segundo [9], “um modelo de referência de visualização permite a identificação dos componentes essenciais a serem considerados na utilização de uma determinada técnica ou no desenvolvimento de uma nova. O modelo clássico de visualização, proposto por Harber e McNabb ([9] *apud* [10]), mostrado na Figura 2.1 é um *pipeline* simples, onde dados sofrem filtragem e mapeamento para alguma representação geométrica, a qual, finalmente passa por um processo de geração de imagem (*rendering*). Outro modelo, proposto por Card, Mackinlay e Shneiderman ([9] *apud* [1]), descreve visualização como uma sequência de mapeamentos ‘ajustáveis’ de dados para uma representação visual de modo a possibilitar interação do usuário com o espaço de informação. A sequência de mapeamento encontra-se reproduzida na Figura 2.2. Neste modelo, os ‘dados brutos’ (coletados ou gerados por algum processo) são transformados em tabelas, ou seja, descrições relacionais que incluem

Tabela 2.2: Classes de representações visuais [9].

Classe	Tipo	Utilização
Gráficos 2D, 3D	Pontos	Representação da distribuição dos elementos no espaço domínio,
	Circulares	representação da dependência/correlação entre os atributos.
	Linhas	
	Barras	
Superfícies (para 3D)		
Ícones, glifos e objetos geométricos	Elementos geométricos 2D ou 3D diversos	Representação de entidades em um contexto, representação de grupos de atributos de diversos tipos.
Mapas	de pseudo-cores	Representação de campos escalares ou de categorias.
	de linhas	Representação de linhas de contorno de regiões ou isovalores.
	de superfícies	Idem, no espaço 3D.
Diagramas	de ícones e símbolos diversos	Representação de grupos de atributos (categorias, escalares, vetoriais, tensoriais).
	Nodos e arestas	Representação de relacionamentos diversos: É-um, É-parte-de, comunicação, sequência referência, etc.

metadados. O uso de tabelas é uma simplificação desnecessária, pois os dados podem ser representados em outros tipos quaisquer de estruturas de dados, dependendo da aplicação.

Além da transformação inicial dos dados brutos em descrições relacionais, novas transformações podem ser aplicadas, seja para agregar novos dados ao conjunto inicial (por exemplo, calcular grandezas estatísticas), quer para converter os dados originais para outros tipos ou reorganizar o conjunto de dados (por exemplo, classificação).

As tabelas relacionais são, então, mapeadas para estruturas ou representações visuais. Estas estruturas visuais são, por sua vez, exibidas em imagens. Operações sobre essas representações visuais correspondem a transformações visuais e objetivam, genericamente, mostrar informações adicionais sobre elementos do conjunto de dados através de mudança do ponto de observação, manipulação geométrica ou indicação de região/subconjunto de interesse”.

#### 2.1.4 Vantagens

Dentre as vantagens que podem ser associadas ao uso de VISINFO podem ser citadas as seguintes [4]:

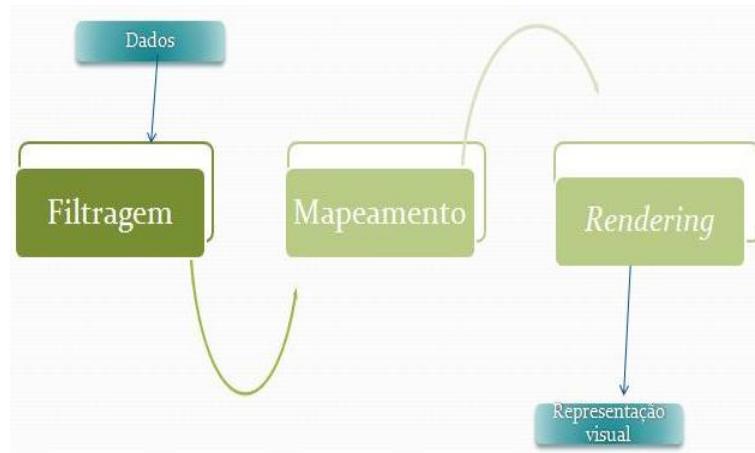


Figura 2.1: Modelo clássico de visualização [10].

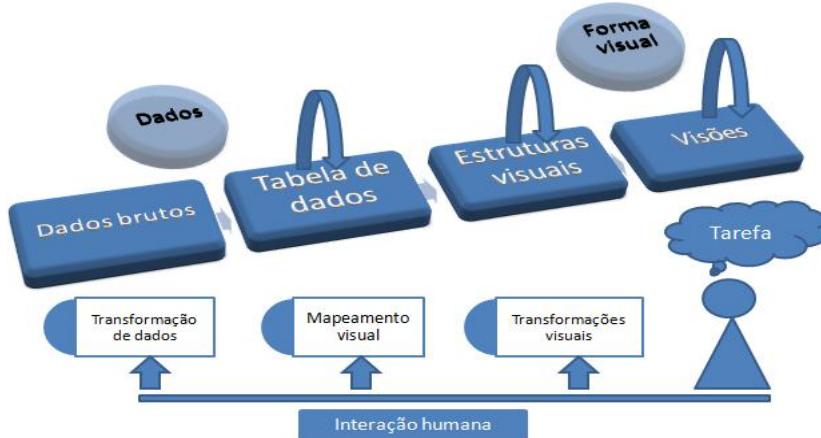


Figura 2.2: Modelo de referência de visualização [1].

- Uma grande quantidade de informações pode ser condensada em apenas uma única e simples imagem que representa essa informação de forma a ser facilmente entendível;
- O processo de visualização trabalha com o sentido humano da visão, que é o mais capaz de perceber informações por unidade de tempo. O sentido da visão é paralelo e rápido, sendo possível prestar atenção em determinado objeto sem perder de vista (obviamente, com menos detalhes) o que está acontecendo ao redor;
- O sentido humano da visão é treinado para reconhecer padrões, sendo possível identificar e lidar com diversos novos problemas do cotidiano com a utilização de padrões já vistos e que envolvam o processamento visual de informações. A constatação de padrões ou características visuais presentes em imagens contribui de forma significativa para o processo de compreensão, mais do que a simples observação dos dados em sua forma bruta.

A construção de uma visualização que possibilite a observação desses padrões pode ser obtida organizando os dados segundo algum critério definido e apresentando-os de modo visual. Isto se torna vantajoso, pois une uma grande quantidade de informação relevante que estava dispersa em uma ou várias formas visuais, facilitando a análise desses dados e consequente uso da maior quantidade de informação possível.

### 2.1.5 Aplicações

As técnicas de VISINFO têm sido utilizadas em diversas áreas tais como: monitoramento de bolsas de valores e estudo de mercados; consultas a bancos de dados através de técnicas de *data mining*, envolvendo áreas como jogos, tendências de consumo de produtos e sistemas de informações geográficos e engenharia de *software*, através da utilização de visualizações que auxiliem o programador a analisar e entender o funcionamento de um programa, em um nível maior de abstração quando comparado a uma simples análise do código fonte.

As técnicas de VISINFO têm sido usadas também na área de redes de computadores com a finalidade de realizar análises da grande quantidade de dados gerados, aplicando filtros sobre esses dados a fim de perceber padrões que essas redes apresentam, e apresentando várias informações simultaneamente em uma única imagem.

### 2.1.6 Técnicas

As técnicas de VISINFO procuram representar graficamente dados de um determinado domínio de aplicação de forma que, a partir das relações espaciais exibidas, o ser humano interprete e compreenda as informações apresentadas e, finalmente, deduza novos conhecimentos. Elas têm o potencial de auxiliar na análise visual e exploração de grandes conjuntos de dados, através do emprego de mecanismos que buscam tanto representar visualmente os dados, quanto permitir ao usuário a interação com estas representações.

Em uma VISINFO, os dados são abstratos e não há necessariamente uma representação geométrica inerente aos mesmos. Uma imagem é gerada com base nos relacionamentos e informações que podem ser inferidos acerca dos dados. O processo de se gerar uma visualização, basicamente, consiste em mapear os atributos dos dados abstratos em atributos visuais de uma imagem. Os dados abstratos a serem visualizados podem ser classificados nas seguintes categorias principais [1]:

- Nominal — conjunto de elementos distintos, sem uma relação de ordem entre eles. Exemplo: melão, maçã, melancia, morango;
- Ordinal — conjunto de elementos distintos, mas com uma relação de ordem entre os mesmos. Exemplo: primeiro, segundo, terceiro, ..., ou segunda, terça, quarta, ... ;
- Quantitativo — faixa de valores numéricos. Essa categoria pode ser dividida em intervalos (com valores discretos) e razão (representando uma faixa contínua de valores).

Para o desenvolvimento de sistemas que utilizem técnicas de VISINFO deve-se considerar a melhor forma de se mapear informações para uma representação gráfica que facilite a sua interpretação pelos usuários. Deve-se, também, fornecer meios para limitar ou filtrar a quantidade de informação a ser interpretada, mas mantendo o usuário ciente do espaço total de informações.

É necessário, também, possibilitar formas de manipulação do conjunto de dados, tanto geométrica (rotações e *zoom* na representação gráfica, por exemplo) como analiticamente (redução ou expansão do conjunto de dados exibido de acordo com algum critério determinado pelo usuário).

De acordo com Card, Mackinlay e Shneiderman [1], as estruturas dimensionais de representação são as seguintes: as estruturas unidimensionais, que são tipicamente empregadas para a apresentação de documentos de texto ou de linhas do tempo, podendo ser combinadas com um segundo ou terceiro eixo para mostrar comparação entre valores; as estruturas visuais bidimensionais, como gráficos de linhas, barras, pizza e mapas, que são mais apropriadas para apresentar dados estatísticos, descrever funções matemáticas ou visualizar informações geográficas; e para dados com  $n$  dimensões ( $n$  maior que 3), quando técnicas mais elaboradas são necessárias, tais como, *Multidimensional Scalling*, Coordenadas Paralelas, *treemap*, dispersão e *Glyphs*.

Para ilustrar, a seguir são apresentadas brevemente três técnicas de VISINFO [23]:

1. *Treemap* — técnica baseada no preenchimento de espaços. Os itens são ordenados por um valor de atributo que é representado por um retângulo de área proporcional ao valor deste atributo. É uma técnica ideal para representar dados hierárquicos e fazer correlações entre dados e grupos de dados. A Figura 2.3, retirada de RDI [23], mostra uma imagem gerada com a utilização da técnica de *treemap*.
2. Dispersão — técnica baseada na ideia de gráfico X-Y, ou seja, os eixos são configuráveis por atributo. Cada ponto ou coordenada representa um conjunto de valores de atributos. É uma técnica ideal para identificação de grupos de dados por afinidade de características. A Figura 2.4, retirada de RDI [23], mostra uma imagem gerada com a utilização da técnica de dispersão.
3. Coordenadas paralelas — técnica proposta por Inselberg [20], mapeia um espaço  $n$ -dimensional em uma estrutura bidimensional que usa  $n$  eixos paralelos verticais equidistantes, denominados coordenadas. Os eixos verticais representam as dimensões ou atributos dos dados e uma linha representando cada item de dado conecta os eixos nos seus respectivos valores, permitindo a observação de padrões e tendências nos dados [21]. A Figura 2.5, retirada de RDI [23], mostra um gráfico utilizando coordenadas paralelas.

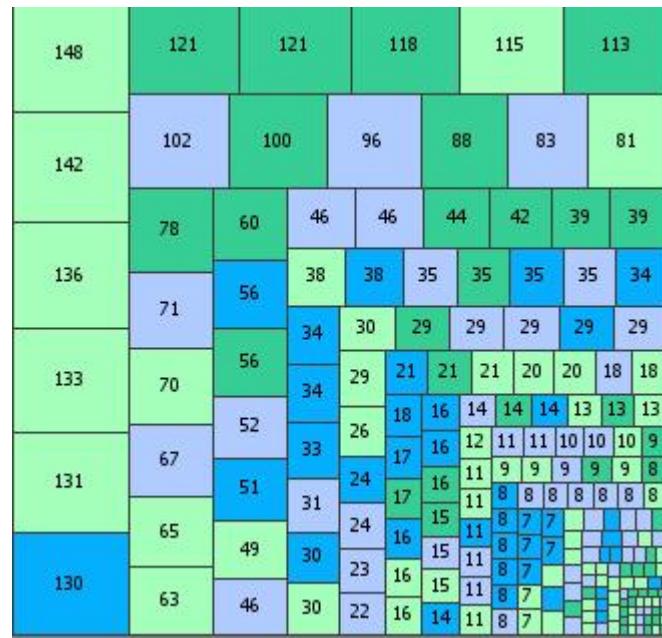


Figura 2.3: Imagem gerada com a utilização da técnica de *treemap*, retirada de RDI [23].

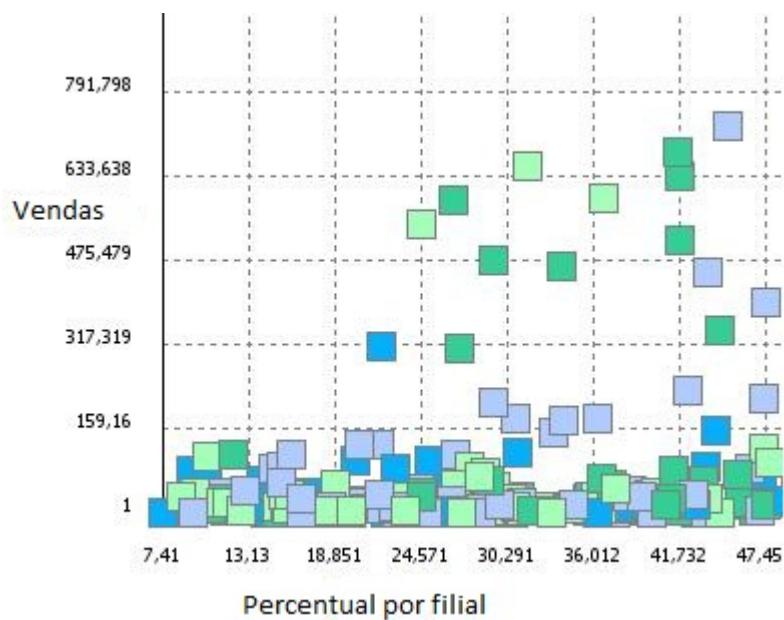


Figura 2.4: Imagem gerada utilizando a técnica de dispersão, retirada de RDI [23].

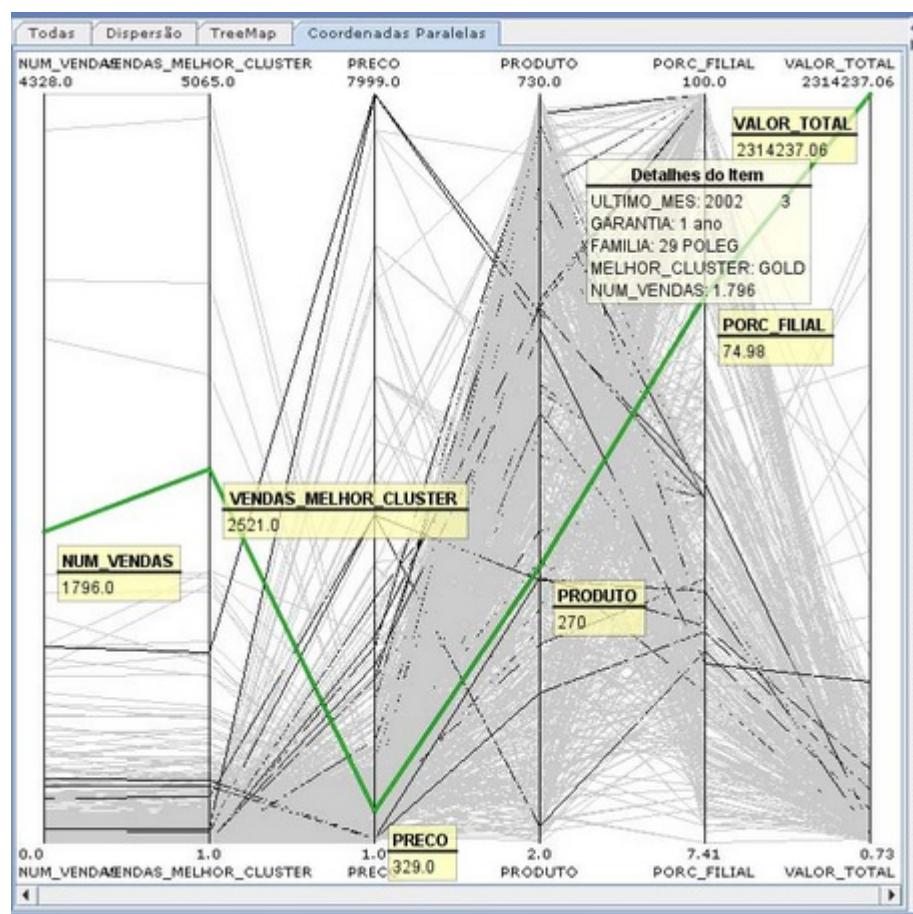


Figura 2.5: Exemplo de gráfico com coordenadas paralelas, retirado de RDI [23].

## 2.2 Gerência de Redes de Computadores

### 2.2.1 Introdução

A área de GRC foi inicialmente impulsionada pela necessidade de monitoração e controle do universo de dispositivos que compõem as redes de comunicação. O aumento do grau de complexidade das redes e do seu tamanho exige o emprego de um sistema de gerenciamento que proporcione qualidade de serviço, pró-atividade, integração com processos de serviços e de negócios. O contínuo crescimento em número e diversidade dos componentes de redes de computadores tem tornado a atividade de gerenciamento cada vez mais complexa, e o uso dos serviços das redes é afetado pela disponibilidade e eficiência do gerenciamento feito nas redes de computadores [11].

Atualmente, as redes de computadores e os seus recursos associados têm se tornado fundamentais e de tal importância para uma organização, que elas “não podem falhar”, sob o risco da organização parar se a rede não estiver funcional. Isto significa que o nível de falhas e de degradação de desempenho considerados aceitáveis está cada vez mais diminuindo, sendo o nível desejado igual ou muito próximo a zero, dependendo da importância da rede para a instituição [11].

Para resolver os problemas associados da GRC a ISO (*International Organization for Standardization*) propôs três modelos para o gerenciamento de redes [11]:

1. Modelo Organizacional — estabelece a hierarquia entre sistemas de GRC em um domínio de gerência, dividindo o ambiente a ser gerenciado em vários domínios.
2. Modelo Informacional — define os objetos de gerência, as relações e as operações sobre esses objetos. Uma MIB (*Management Information Base*) é necessária para armazenar os objetos gerenciados.
3. Modelo Funcional — descreve as funcionalidades de gerência: Gerenciamento de Falhas, Gerenciamento de Configuração, Gerenciamento de Contabilização, Gerenciamento de Desempenho e Gerenciamento de Segurança.

Uma forma simples de caracterizar as funções de gerenciamento é através das funcionalidades que envolvem o modelo FCAPS, sigla formada a partir das iniciais de cada área de gerenciamento em inglês (*Fault, Configuration, Accounting, Performance e Security*). Este modelo serve de base para os demais por definir as áreas funcionais da GRC [11].

O Gerenciamento de Falhas é o responsável pelo desenvolvimento de aplicações que identifiquem as falhas e informem a um gerente a ocorrência dessas, para que decisões e atitudes sejam tomadas para resolver o problema ocorrido. Com as informações sobre falhas ocorridas, a qualidade de serviço acertada com um usuário tende a ser mantida, uma vez que o setor responsável pela operação e/ou manutenção da rede antecipa-se aos usuários na solução de problemas de rede [40].

O Gerenciamento de Configuração é a área responsável pela análise, estudo e gerenciamento da topologia e do fluxo de dados da rede através de ferramentas gráficas.

O Gerenciamento de Contabilização é importante para a identificação e registro de custos e do volume de recursos utilizados pelos usuários de uma rede.

O Gerenciamento de Desempenho é importante não para garantir apenas a qualidade de serviço acordada com os usuários, mas também para assegurar que seja atingida com os menores custos possíveis. Pode-se por meio do Gerenciamento de Desempenho, adequar os meios de comunicação utilizados pelos usuários às suas reais necessidades, auxiliando o setor responsável pela administração da rede, e se antecipado aos usuários na manutenção dos níveis de desempenho dos serviços oferecidos, como, por exemplo, o tempo de resposta.

O Gerenciamento de Segurança é o responsável pela monitoração e controle dos mecanismos de segurança de uma rede. Estes mecanismos envolvem desde os mecanismos de controle de acesso aos sistemas computacionais até o controle de informações sigilosas que trafegam na rede. Exemplos destes mecanismos incluem desde proteção de senhas, com uso de criptografia até a obrigatoriedade de troca de senha periodicamente.

A Figura 2.6 apresenta uma forma de integração entre as 5 áreas funcionais de GRC [3]. Se as áreas de Segurança, de Contabilização, de Falhas e de Configuração funcionarem de forma adequada, a área de Desempenho será beneficiada, pois o desempenho de uma rede é diretamente influenciado por problemas ocasionados em uma ou mais das quatro áreas mencionadas.



Figura 2.6: Integração entre as áreas funcionais de Gerência de Redes [3].

### 2.2.2 Arquitetura de Gerenciamento

Os componentes essenciais de uma arquitetura de gerenciamento de redes são [11]:

- Descrição da Estrutura de Gerenciamento — relacionada com as informações que podem ser efetuadas sobre os objetos gerenciados (SMI - *Structure Management Information*).
- Definição da Base de Informação de Gerenciamento — lista inicial dos objetos gerenciados (MIB).

- Protocolos e serviços — utilização de protocolos e serviços de gerenciamento sobre os protocolos de transporte TCP (*Transmission Control Protocol*) ou UDP (*User Datagram Protocol*).

Uma SMI define como são identificadas e descritas as informações gerenciadas. A MIB é especificada utilizando a SMI e contém os objetos a serem gerenciados.

Um exemplo de SMI é a SMI Internet, projetada para ser uma estrutura de informação dependente de um protocolo, por exemplo, o protocolo SNMP (*Simple Network Management Protocol*) [11].

### 2.2.3 MIB

Um objeto gerenciado é a visão abstrata de um recurso real do sistema. Assim, todos os recursos da rede que devem ser gerenciados são modelados, e as estruturas de dados resultantes são os objetos gerenciados. Os objetos gerenciados podem ter permissões para serem lidos ou alterados, sendo que cada leitura representará o estado real do recurso e, cada alteração também será refletida no próprio recurso [25].

Dessa forma, a MIB é o conjunto dos objetos gerenciados, que procura abranger todas as informações necessárias para a gerência da rede, possibilitando assim, a automatização de grande parte das tarefas de gerência.

O RFC (*Request for Comments*) 1066 [26] apresentou a primeira versão da MIB para uso com o protocolo TCP/IP, a MIB-1. Este padrão explicou e definiu a base de informação necessária para monitorar e controlar redes baseadas no protocolo TCP/IP. O RFC 1066 foi aceito pela IAB (*Internet Activities Board*) como padrão no RFC 1156. O RFC 1158 propôs uma segunda MIB, a MIB-2, para uso com o protocolo TCP/IP, sendo aceita e formalizada como padrão no RFC 1213. A MIB-2 expandiu a base de informações definidas na MIB-1.

A MIB-2 usa uma arquitetura de árvore, definida na ISO ASN.1 (*Abstract Syntax Notation.1*), para organizar todas as suas informações. Cada parte da informação da árvore é um nó rotulado que contém um OID (*Object Identifier*), ou seja, uma sequência de números separados por pontos e uma pequena descrição textual [25]. Um exemplo de objeto está descrito a seguir:

```
directory(1)
  identificador de objetos: 1.3.6.1.1
  descrição textual: {internet 1}
```

Um nó rotulado pode ter subárvores contendo outros nós rotulados. Caso não tenha subárvores, ele é um nó folha e será um objeto que contém um valor. Na Figura 2.7 é apresentada a árvore que representa a MIB-2.

O nó raiz da árvore MIB-2 não tem nome ou número, mas tem três subárvores:

1. *ccitt(0)*, administrada pelo CCITT (Comitê Consultivo Internacional de Telegrafia e Telefonia).
2. *iso(1)*, administrada pela ISO.

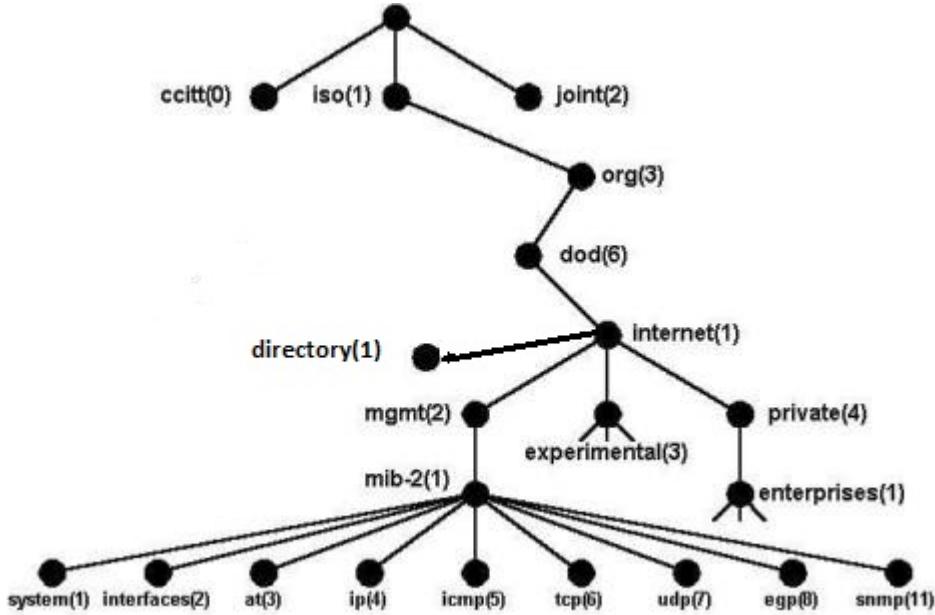


Figura 2.7: A MIB-2.

3. *joint-iso-ccitt(2)*, administrada pela ISO juntamente com o CCITT.

Sob o nó *iso(1)*, estão outras subárvore, como é o caso da subárvore *org(3)*, definida pela ISO para conter outras organizações. Uma das organizações que está sob a subárvore *org(3)* é o DOD (*Departament of Defense of USA*), no nó *dod(6)*. A *Internet(1)* está sob o *dod(6)*, e possui quatro subárvore:

1. *directory(1)*: contém informações sobre o serviço de diretórios OSI (*Open Systems Interconnection*).
2. *mgmt(2)*: contém informações de gerenciamento, é sob esta subárvore que está o nó da *mib-2*, com o identificador de objeto 1.3.6.1.2.1 ou { *mgmt* 1 }.
3. *experimental(3)*: contém os objetos que ainda estão sendo pesquisados pela IAB.
4. *private(4)*: contém objetos definidos por outras organizações.

Abaixo da subárvore *mib-2* estão os objetos usados para obter informações específicas dos dispositivos da rede. Esses objetos são divididos em 11 grupos, que são apresentados na Tabela 2.3 e detalhados a seguir, com as respectivas áreas FCAPS que cada grupo abrange [25]. Na Tabela 2.4 são apresentados alguns exemplos de objetos pertencentes a cada um dos grupos.

Cada objeto contido nos grupos apresentados na Tabela 2.4 é descrito no RFC 1213. A descrição dos objetos é dividida em cinco partes: o nome do objeto, a sintaxe abstrata do objeto, a descrição textual do significado do objeto (nome completo, versão, etc.), o tipo de

Tabela 2.3: Áreas FCAPS dos grupos da subárvore *mib-2*.

MIB-2						
Grupos	Informação	F	C	A	P	S
<i>system(1)</i>	Sistema de operação	•	•			
<i>interfaces(2)</i>	Interface da rede	•	•	•	•	
<i>address translation(3)</i>	Mapeamento de endereços IP					
<i>ip(4)</i>	Protocolo IP	•	•	•	•	
<i>icmp(5)</i>	Protocolo ICMP				•	
<i>tcp(6)</i>	Protocolo TCP		•	•	•	•
<i>udp(7)</i>	Protocolo UDP		•	•	•	
<i>egp(8)</i>	Protocolo EGP	•	•	•	•	
<i>cmot(9)</i>	Protocolo CMOT					
<i>transmission(10)</i>	Meios de transmissão					
<i>snmp(11)</i>	Protocolo SNMP	•	•	•	•	•

acesso permitido ao objeto (*read-only*, *read-write*, *write-only* ou não acessível) e o estado do objeto (obrigatório, opcional ou obsoleto). O exemplo a seguir apresenta a descrição do objeto `sysDescr { Internet 1 }`.

```
OBJECT
  sysDescr {Internet 1}
  Syntax: DisplayString(SIZE(0..255))
  Description: "A textual description of the entity.
This value should include the full name and version identification of the
system's hardware type, software operating-system and networking software.
It is mandatory that this only contain printable ASCII characters."
  Access: read-only
  Status: mandatory
```

## 2.2.4 Protocolo SNMP

Com a crescente necessidade de Gerenciamento de Redes de Computadores, fez-se necessário que padrões para o desenvolvimento de protocolos fossem estabelecidos. O SNMP é um protocolo de gerência de redes TCP/IP (*Transmission Control Protocol/ Internet Protocol*), da camada de aplicação que facilita o intercâmbio de informações entre os dispositivos de rede, como placas e *switches*, possibilitando aos administradores de rede encontrar e resolver eventuais problemas na rede e prover informações sobre os equipamentos gerenciados [3]. O SNMP é usado para administrar o desempenho de uma rede, encontrar e resolver problemas de rede e planejar o crescimento da mesma. É utilizado em servidores, roteadores, *firewalls*, *switches*, *hubs*, impressoras, *modems*, equipamentos remotos, multiplexadores, etc.

Tabela 2.4: Exemplos de objetos da *mib-2* e seus respectivos grupos.

Grupo	Objeto — Informação (descrição) utilizada no gerenciamento	
1	<i>sysDescr</i> <i>sysLocation</i> <i>sysUpTime</i>	Descrição do sistema Localização física do sistema Quanto tempo o sistema está operacional
2	<i>ifDescr</i> <i>ifSpeed</i>	Nome da interface Largura de banda da interface
4	<i>ipInReceives</i> <i>ipInAddrErrors</i> <i>ipForwDatagrams</i>	Taxa de datagramas de recebidos Taxa de erros de endereço de entrada Taxa de datagramas repassados
5	<i>icmpInMsgs</i> <i>icmpInErrors</i>	Taxa de recebimento de mensagens Taxa de erros de entrada
6	<i>tcpInErrs</i> <i>tcpInSegs</i> <i>tcpOutSegs</i>	Número de pacotes recebidos com erro Taxa de segmentos TCP recebidos Taxa de segmentos TCP enviados
7	<i>udpInDatagrams</i> <i>udpInErrors</i>	Taxa de datagramas recebidos Taxa de datagramas UDP recebidos com erro
8	<i>egpInMsgs</i> <i>egpInErrors</i> <i>egpOutMsgs</i>	Taxa de mensagens recebidas Taxa de mensagens recebidas com erro Taxa de mensagens enviadas
11	<i>snmpInPkts</i> <i>snmpOutPkts</i>	Taxa de pacotes SNMP recebidos Taxa de pacotes SNMP enviados

Os componentes chaves de uma rede gerenciada pelo protocolo SNMP são os seguintes:

- Estação de Gerenciamento;
- Agente de Gerenciamento;
- MIB;
- Protocolo de Gerenciamento de Redes.

A estação de gerenciamento serve como interface para o gerente humano em um sistema de gerenciamento de rede. O agente de gerenciamento responde às solicitações de informações e de ações da estação de gerenciamento e deve também prover assincronamente informações importantes que não foram solicitadas por esta estação. Os recursos a serem gerenciados são representados como objetos, e a coleção de objetos é referenciada como uma MIB. A forma de comunicação entre a estação de gerenciamento e o agente é definida por um protocolo de gerenciamento de rede, por exemplo, o SNMP.

No escopo deste protocolo, definem-se os elementos de rede classificados como cliente (ou gerente) responsável pela monitoração e controle dos *gateways* e *hosts*, que correspondem aos servidores (ou agentes).

O protocolo SNMP é baseado no paradigma conhecido como “busca-armazenamento” (*fetch-store*), isto é, todas as operações previstas por este protocolo são derivadas de operações básicas de busca e armazenamento. As mensagens desse protocolo não possuem campos fixos e são especificadas na notação ASN.1. O SNMP sugere que a lógica do servidor de objetos de um equipamento seja colocada no agente e este execute operações elementares, como estabelecer e obter valores das variáveis. O programa que analisa, manipula, combina ou aplica algum algoritmo sobre os dados deve residir no gerente [11].

Geralmente, o SNMP é executado sobre o UDP e o IP (*Internet Protocol*), e não sobre o TCP, mas é importante ressaltar que o gerenciamento de redes não é semelhante às outras aplicações de rede. Quando tudo falha, o gerenciamento da rede deve continuar funcionando e deve ser capaz de usar a parte mais simples do protocolo de rede (um protocolo de datagrama não-confirmado) para chegar ao equipamento gerenciado. Os projetistas do SNMP optaram pelo uso de protocolos não orientados a conexão por medo, em caso de falha, de ser impossível estabelecer uma conexão [11], além de considerarem o desempenho do protocolo UDP *versus* o protocolo TCP.

O gerenciamento de rede através do SNMP permite o acompanhamento simples e fácil do estado, em tempo real, da rede, podendo ser usado para gerenciar diferentes tipos de sistemas. Cada máquina gerenciada pelo SNMP deve possuir um agente e uma base de informações MIB, que é um conjunto de variáveis que representam informações referentes ao seu estado atual, estas informações ficam disponíveis ao gerente através de consulta e podem ser alteradas por ele.

No protocolo SNMPv1 existem as operações: *set*, *get*, *get-next* e *trap*. Já o protocolo SNMPv2 inseriu melhorias nas operações de *trap* e inseriu duas novas operações: *get-bulk* e *inform*.

- A operação *set* é utilizada para alterar o valor da instância de um objeto em um cliente;
- A operação *get* é utilizada para ler o valor de uma instância de um objeto; o gerente solicita que o agente obtenha e retorne o valor;
- A operação de *get-next* é utilizada para ler o valor da próxima instância de um objeto; o gerente fornece o nome de uma variável e o cliente obtém o valor e o nome da próxima instância de um objeto; também é utilizado para obter valores e nomes de objetos de uma tabela de tamanho desconhecido;
- A operação *trap* é utilizada para comunicar um evento; o agente comunica ao gerente o acontecimento de um evento, previamente determinado. No SNMPv2, a operação de *trap* possui a mesma função que no SNMPv1, porém, usa diferentes formatos de mensagens e foi implementada com o objetivo de substituir a primeira versão.
- A operação *get-bulk* é usada para a busca de uma grande quantidade de dados, como informações presentes em uma tabela. Esta operação é uma otimização em relação à operação de *get*.

- A operação *inform* no SNMPv2 permite a comunicação entre os gerentes em uma rede.

### 2.2.5 Gerenciamento de Desempenho

O Gerenciamento de Desempenho consiste em um conjunto de funções responsáveis pela manutenção e exame dos registros que contém o histórico dos estados de um sistema. Seu objetivo é analisar as tendências do uso dos componentes e definir um planejamento através do dimensionamento dos recursos que devem ser alocados para o sistema. Isto é feito através da monitoração das atividades da rede e do controle dos recursos através de ajustes e trocas. Assim, são atendidos os requisitos dos usuários deste sistema, satisfazendo as suas demandas, ou seja, garantindo que não ocorram insuficiências de recursos quando sua utilização se aproximar da capacidade total do sistema.

As funções de Gerenciamento de Desempenho podem ser divididas em dois grupos [11]:

- Medidas de Tráfego: estas funções possibilitam ao usuário definir e controlar a entrega de relatórios de medidas de tráfego.
- Monitoração de Desempenho: são informações que permitem ao usuário obter, avaliar e relatar parâmetros de desempenho da rede. Tais informações podem ser utilizadas como apoio ao diagnóstico de falhas, ao planejamento da rede e à qualidade do serviço.

Para atingir os objetivos de desempenho desejáveis, deve-se monitorar a taxa de utilização dos recursos, a taxa em que estes recursos são pedidos e a taxa em que os pedidos a um recurso são rejeitados. Para cada tipo de monitoração, é definido um valor máximo aceitável (*threshold*), um valor de alerta e um valor em que se remove a situação de alerta.

Na Gerência de Desempenho tem-se a possibilidade de avaliar o comportamento dos recursos em um ambiente de gerenciamento para verificar se este comportamento é eficiente, ou seja, preocupa-se com o desempenho corrente da rede, através de parâmetros estatísticos como atrasos, vazão, disponibilidade e o número de retransmissões realizadas.

Algumas das questões relativas ao gerenciamento do desempenho são [24]:

- Qual é o nível de capacidade de utilização?
- O *throughput* foi reduzido para níveis aceitáveis?
- O tráfego é excessivo?
- Existem gargalos?
- O tempo de resposta está aumentando?

Para tratar estas questões, o gerente de uma rede de computadores deve focar em um conjunto inicial de recursos a serem monitorados, a fim de estabelecer níveis de desempenho. Isto inclui associar métricas e valores apropriados aos recursos de rede que possam fornecer indicadores de diferentes níveis de desempenho. Muitos recursos devem ser monitorados

para se obter informações sobre o nível de operação da rede. Colecionando e analisando estas informações, o gerente da rede pode ficar mais e mais capacitado no reconhecimento de situações indicativas de degradação de desempenho. Estatísticas de desempenho podem ajudar no planejamento, administração e manutenção de grandes redes. Estas informações podem ser utilizadas para reconhecer situações de gargalo antes que elas causem problemas para o usuário final. Ações corretivas podem ser executadas, tais como, trocar tabelas de roteamento para balancear ou redistribuir a carga de tráfego durante horários de pico, ou ainda, a longo prazo, indicar a necessidade de expansão de linhas para uma determinada área [24].

# 3 Trabalhos Relacionados

Dentre os trabalhos relacionados com o Desview (apresentado no Capítulo 4) são apresentadas neste capítulo quatro ferramentas utilizadas largamente na indústria e três trabalhos acadêmicos:

- Cacti;
- Nagios;
- MRTG;
- OpenNMS;
- *Management Traffic Analyzer*;
- Prisma;
- Visualização e Análise de Dados de Monitoramento de Sistemas usando Multi-resolução.

## 3.1 Cacti

O Cacti [5] é uma ferramenta industrial que coleta informações sobre o estado de uma rede de computadores e as exibe através de gráficos. Foi desenvolvido para ser flexível, de modo a se adaptar facilmente a diversas necessidades, bem como ser robusto e fácil de usar. Dentre as funcionalidades do Cacti estão a monitoração do estado de elementos de rede e programas, bem como largura de banda utilizada e uso de CPU (*Central Processing Unit*).

O Cacti utiliza a ferramenta *RRDTool* [16], ou seja, uma ferramenta para criar bases de dados responsáveis por armazenar os dados recolhidos e por gerar os gráficos. O RRD (*Round Robin Database*) foi desenvolvido para armazenar séries de dados numéricos sobre o estado de uma rede de computadores, porém pode ser empregado no armazenamento de qualquer outra série de dados como temperatura, uso de CPU, etc. As informações são repassadas para a ferramenta através de *scripts* ou outros programas escolhidos pelo usuário que devem se encarregar de obter os dados. Pode-se utilizar também o protocolo SNMP para consultar informações em elementos de redes e/ou programas que suportam tal protocolo.

O Cacti é uma solução gráfica de rede projetada para aproveitar as funcionalidades do *RRDTool* como armazenamento de dados e gráficos. O Cacti proporciona modelos gráficos avançados, vários métodos de aquisição de dados, gerenciamento de usuários e de recursos. Todas essas funcionalidades são acessadas através de uma interface gráfica *web* podendo ser utilizado desde redes pequenas até redes complexas, com centenas de dispositivos.

Um exemplo de gráfico gerado pelo Cacti pode ser observado na Figura 3.1. Nessa figura é apresentada a quantidade de memória utilizada, em *kilobytes*, por um elemento de rede, em um intervalo de tempo (no caso 5 em 5 minutos, padrão no Cacti).

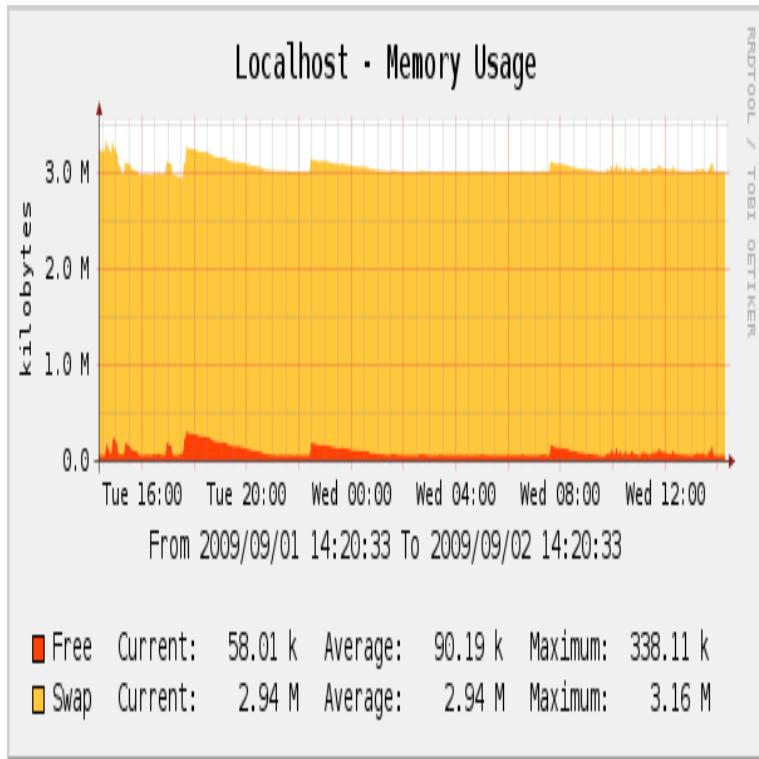


Figura 3.1: Exemplo de gráfico gerado pelo Cacti [5].

## 3.2 MRTG

O MRTG (*Multi Router Traffic Grapher*) [15] é uma ferramenta de monitoração que gera páginas com gráficos de dados coletados a partir de SNMP ou *scripts* externos. É conhecido principalmente pelo seu uso na monitoração de tráfego de rede, utilizando dados solicitados via comandos SNMP a um determinado *host*.

Foi desenvolvido por Tobias Oetiker e Dave Rand em Perl utilizando um módulo em C para gerar os gráficos. O MRTG é uma ferramenta comumente utilizada para monitorar tráfego em interfaces de rede, mas pode monitorar muitas outras variáveis, tais como utilização de disco rígido, temperatura de *hardware* e uso de processador, podendo gerar

alertas a partir de *thresholds*, facilitando, assim, o gerenciamento da rede. O princípio de funcionamento do MRTG é relativamente simples: o programa é executado periodicamente para consultar as informações do banco de dados (MIB) dos equipamentos de acordo com um arquivo de configuração. O resultado desta consulta é usado para construir gráficos que são armazenadas em uma página do diretório *web*. Feito isto, o MRTG permite analisar graficamente através de um navegador, a tendência do tráfego de entrada e saída de um *host*.

O MRTG tem como propósito apresentar na forma de gráficos 2D medidas estatísticas dos dados de dispositivos de rede (*switches*, roteadores e *hubs*), utilizando o protocolo de captura de dados de rede SNMP.

São algumas características do MRTG [15]:

- Possuir opções que permitem personalizar a forma como ela opera;
- Coleta dados a cada 5 minutos por padrão, mas este tempo pode ser alterado;
- Cria uma página HTML com 4 gráficos (diário, semanal, mensal e anual). Se algum deles não for necessário pode ser eliminado;
- Pode avisar caso o valor do gráfico atinja um valor pré-estabelecido. Por exemplo: se determinado servidor atinge 95% do espaço do disco, o MRTG pode mandar um *email* para o administrador informando o ocorrido;
- Possui uma ferramenta para gerar os arquivos de configuração: o CFGMAKER e outra ferramenta para gerar uma página de índice para os casos em que muitos itens são monitorados: o INDEXMAKER.

Exemplos de gráficos diário, semanal, mensal e anual gerados pelo MRTG podem ser observados na Figura 3.2, retirados de Switch [22]. Nos gráficos são mostrados a média de *bits* recebidos e enviados em gráficos com os valores diários, semanais, mensais e anuais coletados da rede.

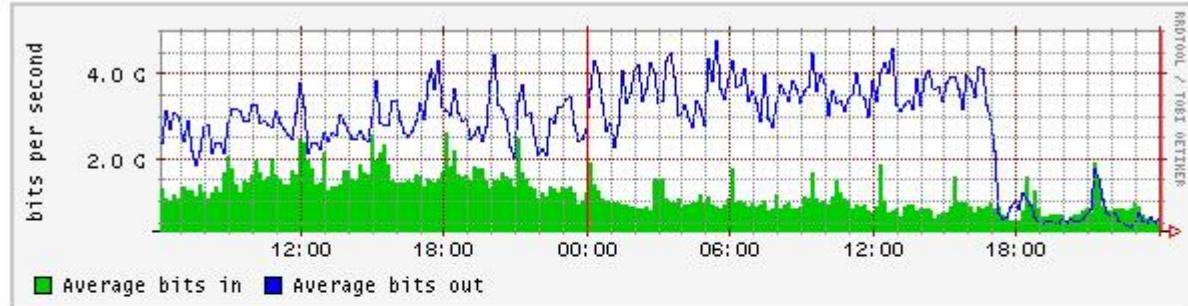
### 3.3 Nagios

O Nagios [6] é uma aplicação, também industrial, de monitoração de rede de código aberto e licenciado pelo sistema GPL (*GNU General Public License*). Ele pode monitorar tanto *hosts* quanto serviços, gerando alertas quando ocorrerem problemas e também quando os problemas forem resolvidos. O Nagios foi originalmente desenvolvido com o nome de Netsaint, foi escrito e é atualmente mantido por Ethan Galstad, juntamente com um conjunto de desenvolvedores que ativamente disponibilizam *plugins* oficiais e não-oficiais.

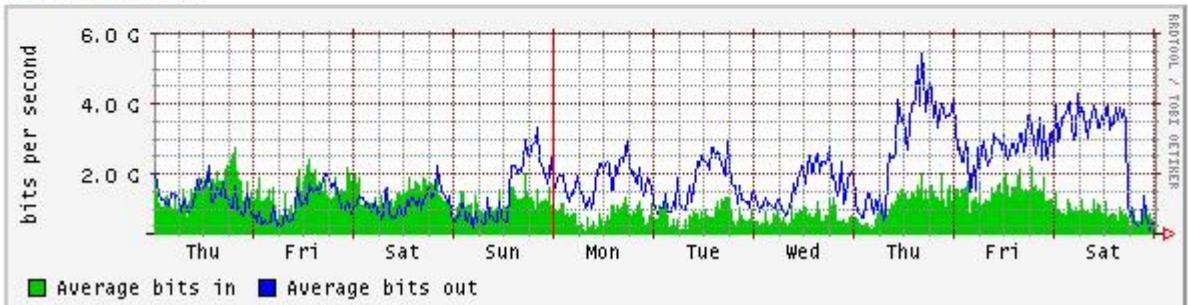
As principais funcionalidades que o Nagios oferece são:

- Monitoração de diversos serviços de rede (SNMP, ICMP, TCP, HTTP e outros);

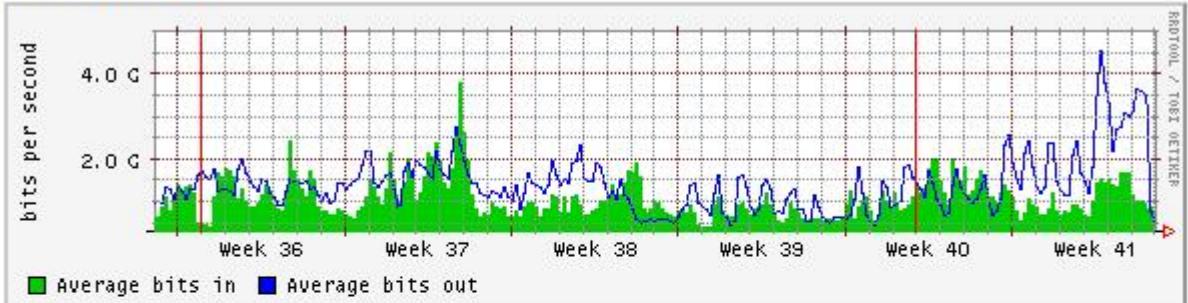
### Daily graph



### Weekly graph



### Monthly graph



### Yearly graph

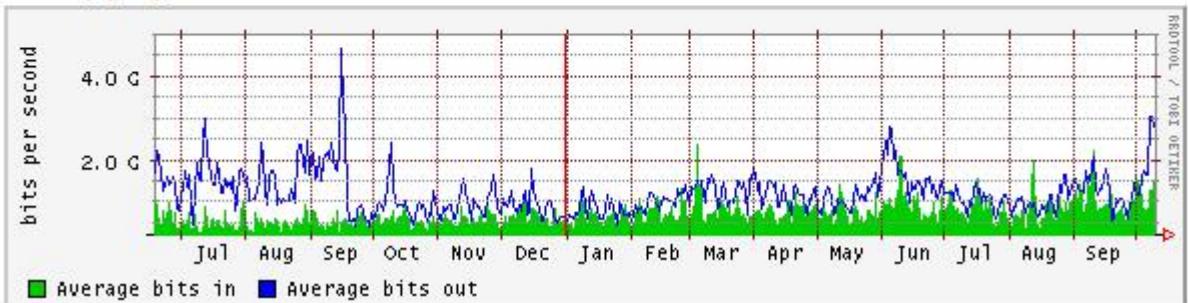


Figura 3.2: Imagens geradas pelo MRTG, retiradas de Switch [22].

- Monitoração de recursos de computadores ou equipamentos de rede (carga do processador, uso de disco e *logs* do sistema);
- Capacidade de notificar quando um serviço ou equipamento apresenta problemas e quando o problema é resolvido (via *email*, mensagem de texto no celular ou qualquer outro meio definido pelo usuário através de um *plugin*);
- Interface *web* para visualização do atual *status* da rede, notificações, histórico de problemas, arquivos de *log*, etc.

Um exemplo de gráfico gerado pela ferramenta Nagios pode ser observado na Figura 3.3. Nessa figura é apresentado o uso de CPU, com valores máximos, média e último valor lido pelos diversos processos de um elemento de rede em um intervalo de tempo.

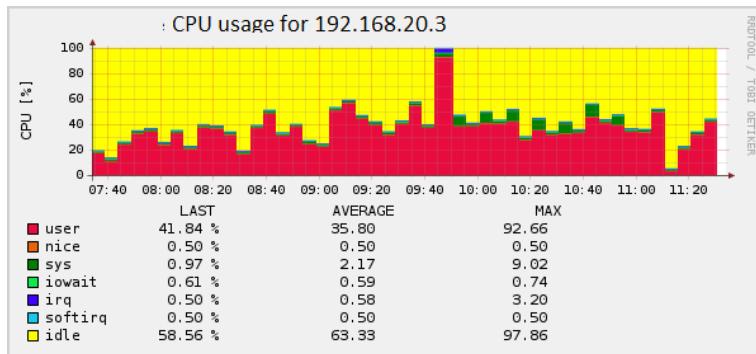


Figura 3.3: Exemplo de gráfico gerado pelo Nagios [6].

## 3.4 OpenNMS

O OpenNMS [37] é um projeto *open-source* desenvolvido em Java, dedicado à criação de uma plataforma de gerência de rede voltada principalmente para a camada de aplicação. Nele é possível visualizar as informações do tipo [38]:

- estado dos serviços e interfaces de rede;
- disponibilidade geral dos serviços;
- eventos gerados;
- gráficos de desempenho;
- informações sobre equipamentos.

Dentre as funcionalidades do OpenNMS pode-se citar a descoberta de recursos e serviços, o *polling* configurável dos equipamentos, calendário de manutenção onde se pode definir períodos em que haverá manutenções que possam afetar a disponibilidade de serviço, coleta de dados através de leituras SNMP, execução de comandos quando determinados *thresholds* forem violados, gráficos estatísticos de utilização, de falhas de *buffer*, de erros *in/out*, de descartes, dentre outros [38].

Na Figura 3.4, tem-se um exemplo de gráfico que a ferramenta OpenNMS gera, onde é possível verificar 4 variáveis da MIB-2 sendo tendo seus valores monitorados e plotados em um gráfico.

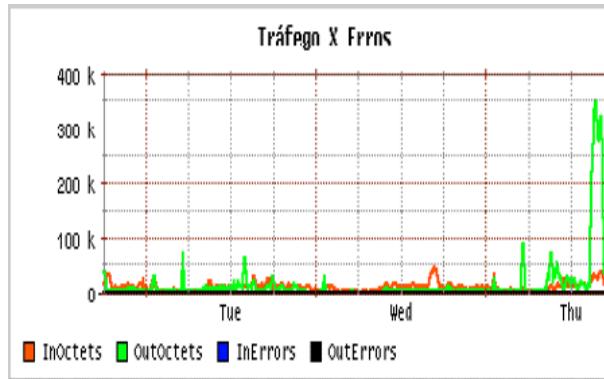


Figura 3.4: Exemplo de gráfico gerado pelo OpenNMS.

### 3.5 Ferramenta *Management Traffic Analyzer*

O trabalho realizado por Salvador e Granville [7] tem por objetivo apresentar através de técnicas de VISINFO, os resultados de análises obtidos com medições do tráfego SNMP, utilizando a ferramenta *Management Traffic Analyzer*, desenvolvida para automatizar este processo. A ferramenta foi implementada sobre SNMPDUMP (programa que analisa tráfego SNMP), invocando as funções de SNMP quando necessário. As análises são realizadas por *scripts* escritos em Perl, e o resultado dessas análises são armazenadas em um banco de dados MySQL. A seguir, técnicas de visualização implementadas como aplicativos usando *Macromedia Flash ActionScript* permitem a visualização gráfica das informações armazenadas. De acordo com Salvador e Granville [7], o uso de técnicas de visualizações foi importante porque elas permitiram explorar e obter visões do tráfego SNMP, facilitando a descoberta de padrões de tráfego, anomalias e disparidades. No trabalho foram apresentadas visualizações baseadas em técnicas visuais, como a apresentada na Figura 3.5, na qual é demonstrado através de um histograma o número de mensagens SNMP que trafegaram pela rede em cada hora durante determinado dia.

A ferramenta apresentada propõe um conjunto limitado de técnicas para visualizar informações específicas relacionadas com o SNMP através de medições de tráfego. Estas visualizações são capazes de gerar gráficos, porém seria necessário aumentar o conjunto de

técnicas de visualização disponíveis na ferramenta, a fim de melhorar o potencial de visão sobre o tráfego SNMP, já que a enorme quantidade de informações geradas por vários dispositivos e sistemas que compõem as redes modernas demanda cada vez mais esforços para o entendimento dessas informações.

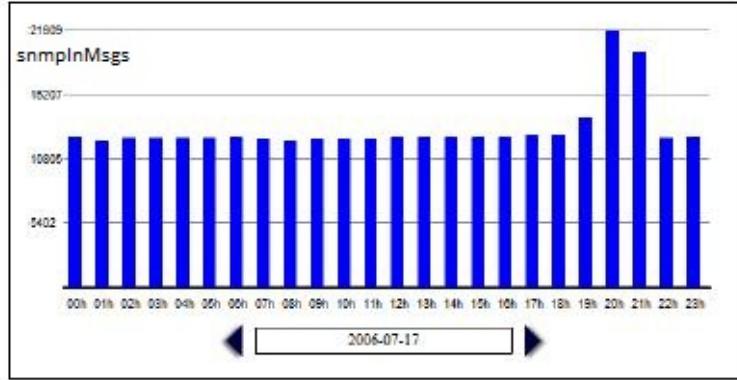


Figura 3.5: Gráfico gerado pela ferramenta *Management Traffic Analyzer* [7].

## 3.6 Ferramenta Prisma

A ferramenta Prisma, desenvolvida por Kauer [8], é uma aplicação que utiliza VISINFO através de técnicas de múltiplas coordenadas para explorar conjuntos de dados de uma rede de computadores. Três técnicas conhecidas de visualização são usadas: *treemap*, dispersão e coordenadas paralelas [10]. A ferramenta utiliza um conjunto de dados proveniente de *logs* de elementos de rede contendo informações como endereço IP, porta e protocolo utilizado, quantidade de *bytes* enviados e recebidos e estado da operação (sucesso, redirecionamento, falha, etc.).

O Prisma foi desenvolvido para ajudar na análise de situações em que grandes volumes de dados são envolvidos. A ferramenta é composta por várias formas de visualização gráfica que permitem aos usuários a análise dos dados e das relações entre eles.

A Figura 3.6 apresenta um exemplo de gráfico gerado pela ferramenta que, de acordo com Kauer [8], possui as seguintes características:

- Extensível, portátil e de fácil manutenção uma vez que foi desenvolvida em Java utilizando *design patterns*;
- A interface gráfica é automaticamente personalizada de acordo com o tipo e o intervalo de valores de dados de cada conjunto;
- Possui filtros que possibilitam ao usuário analisar informações que ele deseja em determinado momento;

- Oferece relatórios de barras, linhas e outras formas gráficas que são gerados automaticamente pela ferramenta.

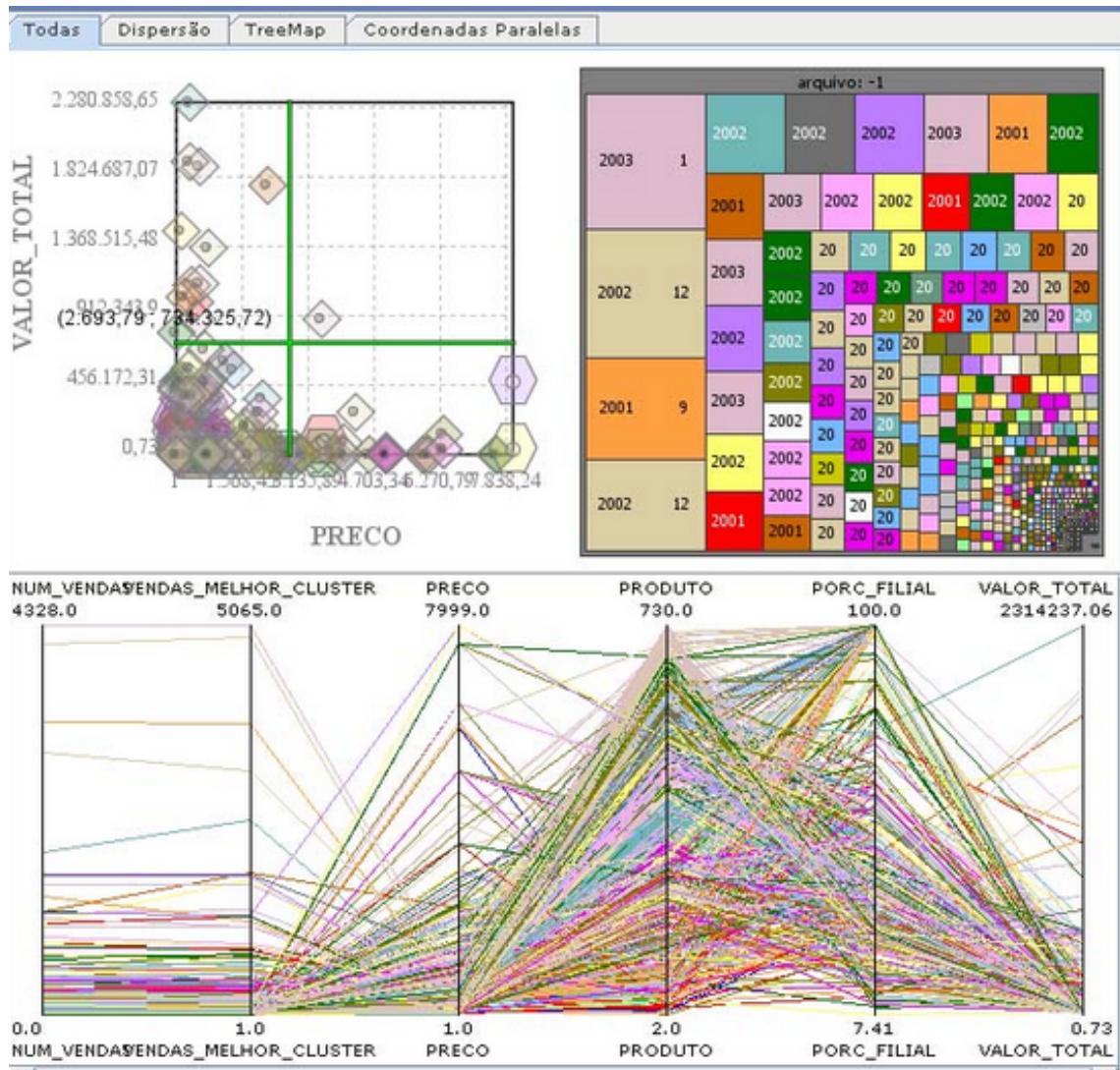


Figura 3.6: Gráfico com as três técnicas utilizadas no Prisma (*treemap*, dispersão e coordenadas paralelas) para o mesmo conjunto de dados de entrada [8].

### 3.7 Visualização e Análise de Dados de Monitoramento de Sistemas usando Multi-resolução

O trabalho realizado por Nayak, Neogi e Kothari [27] explora a natureza multi-resolução do monitoramento de dados e representações de eventos de dados para a construção de uma

topologia que preserve visões para o monitoramento de informações. De acordo com Nayak, Neogi e Kothari [27], o sistema desenvolvido é um *framework* que representa eventos de dados ocorridos. Eventos são normalmente produzidos em certas operações de monitoramento de dados, por exemplo, um evento pode corresponder ao fato de o espaço livre em um disco estar abaixo de 10%. Ao invés de serem observados os pontos máximos ocorridos de eventos, o que normalmente ocorre, é desejável que seja visualizado e entendido o comportamento das complexas infraestruturas que compõem os sistemas atuais. O *framework* proposto é baseado em arranjos e cada fonte de dados monitorada é organizada de forma hierárquica para refletir as inter-relações entre as diferentes fontes de dados monitoradas. No trabalho apresentado são utilizadas redes neurais [29] baseadas em Mapas Auto-Organizáveis [31] para gerar as visualizações a partir dos eventos gerados pelos elementos monitorados, assim, os Mapas Auto-Organizáveis produzem uma topologia preservando o mapeamento em diversas dimensões.

A Figura 3.7 mostra um exemplo de Mapa Auto-Organizável apresentado no trabalho, no qual se mostra a construção de um mapa topológico onde os nodos mais próximos respondem de forma semelhante a padrões de entrada semelhantes a eles, já a Figura 3.8 apresenta uma representação da organização baseada em arranjos, retirada de Nayak [27]. Nela é demonstrado o resultado obtido do mapeamento de dados em 6D para 2D no *framework* desenvolvido.



Figura 3.7: Exemplo de um Mapa Auto-Organizável retirado de Germano [31]. O mapa bi-dimensional gerado representa as posições dos nodos em relação aos nodos vizinhos. O objetivo é que os nodos que forem topologicamente próximos respondam de maneira semelhante a entradas semelhantes.

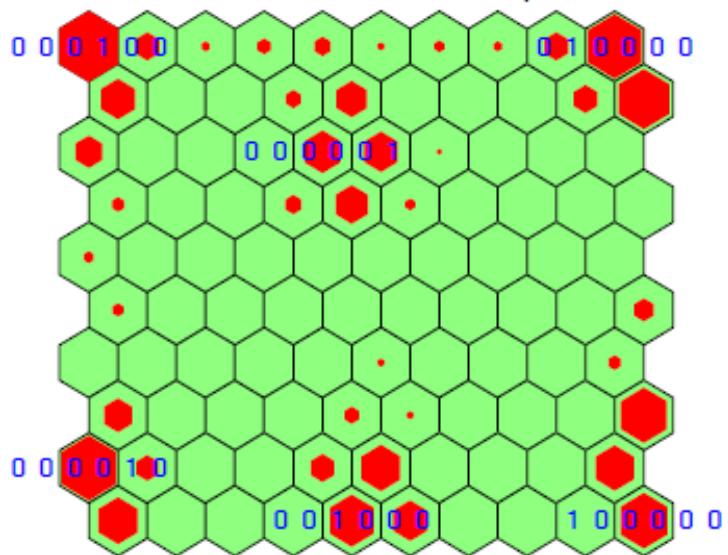


Figura 3.8: Mapeamento de dados de entrada 6-D em 2-D através de Mapas Auto-Organizáveis, onde os dados de entrada correspondem a 6 diferentes vetores com apenas um número ‘1’ em cada vetor. Figura retirada de Nayak [27].

# 4 Descrição do Desview

Neste capítulo são apresentados os objetivos, o ambiente de desenvolvimento utilizado, a modelagem, a interface e as funcionalidades, a implementação do protótipo e um estudo de caso do trabalho que foi desenvolvido.

## 4.1 Introdução

O objetivo deste trabalho é o projeto e o desenvolvimento de um sistema de monitoração de desempenho de elementos em uma rede de computadores que a partir de dados coletados por monitorações via protocolo SNMP, utilizando técnicas de VISINFO sobre os mesmos, de modo que o usuário verifique os estados e dados de variáveis em equipamentos com base nas visualizações que são apresentadas. Assim, caso os valores de variáveis que estão em monitoramento passem de valores pré-definidos, as visualizações informem de maneira fácil ao usuário que a variável em monitoração saiu do intervalo do qual é considerado dentro do desempenho considerado normal.

São requisitos do protótipo que foi desenvolvido:

- Permitir a definição de quais parâmetros serão monitorados;
- Estabelecer parâmetros das medidas de desempenho a serem realizadas;
- Agendar o monitoramento dessas medidas;
- Permitir a consulta, a alteração, a inclusão e a exclusão de medidas;
- Armazenar os resultados obtidos;
- Apresentar os dados obtidos durante as medições através de diferentes representações visuais.

## 4.2 Ambiente de desenvolvimento

Para a implementação do sistema proposto foi utilizada a linguagem de programação Java, a biblioteca OpenGL para a geração das representações visuais, e a biblioteca SNMP4J para a comunicação com o protocolo SNMP dos elementos de rede monitorados.

A biblioteca OpenGL é composta por um conjunto de funções, que fornecem acesso a praticamente todos os recursos do *hardware* de vídeo. Internamente, ela age como uma máquina de estados, que de maneira bem específica diz à placa de vídeo o que deve ser feito. Usando as funções da biblioteca, pode-se especificar, ligar ou desligar vários aspectos dessa máquina, tais como a cor atual, a transparência que será usada, os parâmetros de iluminação, efeitos de neblina, entre outros [30].

Para a implementação de gráficos onde é usada seleção de objetos em OpenGL foi utilizada a biblioteca *Graphic Engine API* [39], que possui métodos que facilitam a seleção e manipulação de formas gráficas em OpenGL. Escolheu-se utilizar esta API devido a problemas encontrados durante o desenvolvimento de gráficos que necessitavam de seleção de objetos, e esta mostrou-se ser uma biblioteca simples com maior facilidade de implementação da funcionalidade desejada.

A linguagem Java foi utilizada para o desenvolvimento por ser uma linguagem livre utilizada em larga escala na indústria, possuir uma grande variedade de *frameworks* que facilitam a programação além de ser portável e orientada a objeto. Foi utilizada a biblioteca OpenGL para Java (JOGL [12]), devido ao fato de ela ter implementação para *hardware* de vídeo, permitindo desenhar, criar superfícies e imagens 3D em tempo real. Já a biblioteca SNMP4J [14] é uma implementação *open source* das primitivas SNMP para a linguagem Java, fornecendo várias funcionalidades para facilitar a comunicação entre uma aplicação e um agente SNMP de determinado elemento de rede.

Também foi utilizado o PostgreSQL [13] como banco de dados para o armazenamento das informações, por ser de código aberto, de alta performance, estável e que possui inúmeros recursos como integridade transacional, controle de concorrência e suporte para chaves estrangeiras e gatilhos. Além disso, possui capacidade de lidar com grandes volumes de dados, o que é normalmente gerado pelas redes de computadores atuais. Os gráficos 2D são implementados utilizando as bibliotecas JFreeChart [32] e ChartDirector [33].

Como ferramentas de desenvolvimento foram utilizados os seguintes *softwares*: plataforma de programação *NetBeans* [17], ferramentas de modelagem *UML Jude Community* [19] e de banco de dados *Power\*Architect* [18].

Foram utilizados, também, a biblioteca Swingx [34], a biblioteca Quartz [35] e o *framework* Hibernate [36] no desenvolvimento das interfaces gráficas, escalonamento e persistência de dados, respectivamente. A Swingx é uma extensão da biblioteca Swing do Java que contém componentes aprimorados com inúmeras funcionalidades no desenvolvimento de aplicações, contendo componentes como calendários, painéis “dobráveis”, componentes que informam que o sistema está processando alguma requisição, dentre outros. A biblioteca Quartz é um escalonador de tarefas, no qual se pode indicar a data e hora exata na qual se deseja executar um trabalho. O Hibernate é um *framework* para o mapeamento objeto-relacional escrito na linguagem Java que facilita o mapeamento dos atributos entre uma base tradicional de dados relacionais e o modelo objeto de uma aplicação. O objetivo do Hibernate é diminuir a complexidade entre os programas Java, baseado no modelo orientado a objeto, que precisam trabalhar com um banco de dados do modelo relacional (presente na maioria dos sistemas de bancos de dados), principalmente no desenvolvimento de consultas e atualizações dos dados [36].

## 4.3 Modelagem do protótipo

Nesta seção são apresentados os diagramas UML que foram modelados para a implementação do protótipo, utilizando a ferramenta *Jude Community* [19]. A Figura 4.1 apresenta os casos de uso, ou seja, uma análise dos requisitos levantados para o sistema, que são os seguintes:

- Criar, excluir e atualizar tarefas e equipamentos;
- Inserir, excluir e atualizar variáveis de tarefas;
- Parar e iniciar tarefas;
- Exportar dados estatísticos;
- Consultar dados persistidos em banco;
- Estabelecer e atualizar parâmetros de desempenho para as variáveis monitoradas;
- Gerar gráficos e visualizações sobre os dados coletados.

Os diagramas de atividades que detalham os casos de uso apresentados encontram-se no Anexo A e modelo conceitual de classes do protótipo encontra-se no Anexo B.

O sistema foi implementado utilizando o padrão de arquitetura de *software MVC* (*model-view-controller*), onde os dados (*model*) são separados da interface (*view*) do sistema. Assim, alterações feitas na interface não interferem na manipulação dos dados, assim como estes também podem ser reorganizados sem alterar a interface. O MVC resolve este problema através da separação das tarefas de acesso aos dados e lógica de negócio, lógica de apresentação e de interação com o usuário, introduzindo um componente entre os dois: o *controller*. Outros padrões de projetos utilizados na implementação foram: o padrão *facade* para o acesso SNMP e o padrão *DAO* (*Data Access Object*) para acesso ao banco de dados.

O protótipo possui os seguintes pacotes principais: *model*, *controller*, *util*, *graphics*, *charts* e *view*. Os pacotes são apresentados brevemente a seguir e os diagramas de classes destes encontram-se no Anexo B.

O pacote *model* contém as entidades do sistema, respectivas classes DAO de cada entidade e classes de utilidades usadas nas entidades.

São características das entidades:

- *Equipment*: entidade que representa um equipamento a ser monitorado na rede, possui atributos como endereço IP, nome, comunidade de escrita e de leitura, *timeout* nas chamadas SNMP, quantidades de *retries* de acesso ao equipamento.
- *Variable*: entidade que representa uma variável da MIB, como atributos possui *oid*, nome, *thresholds*, tipo da variável e tipo de acesso.

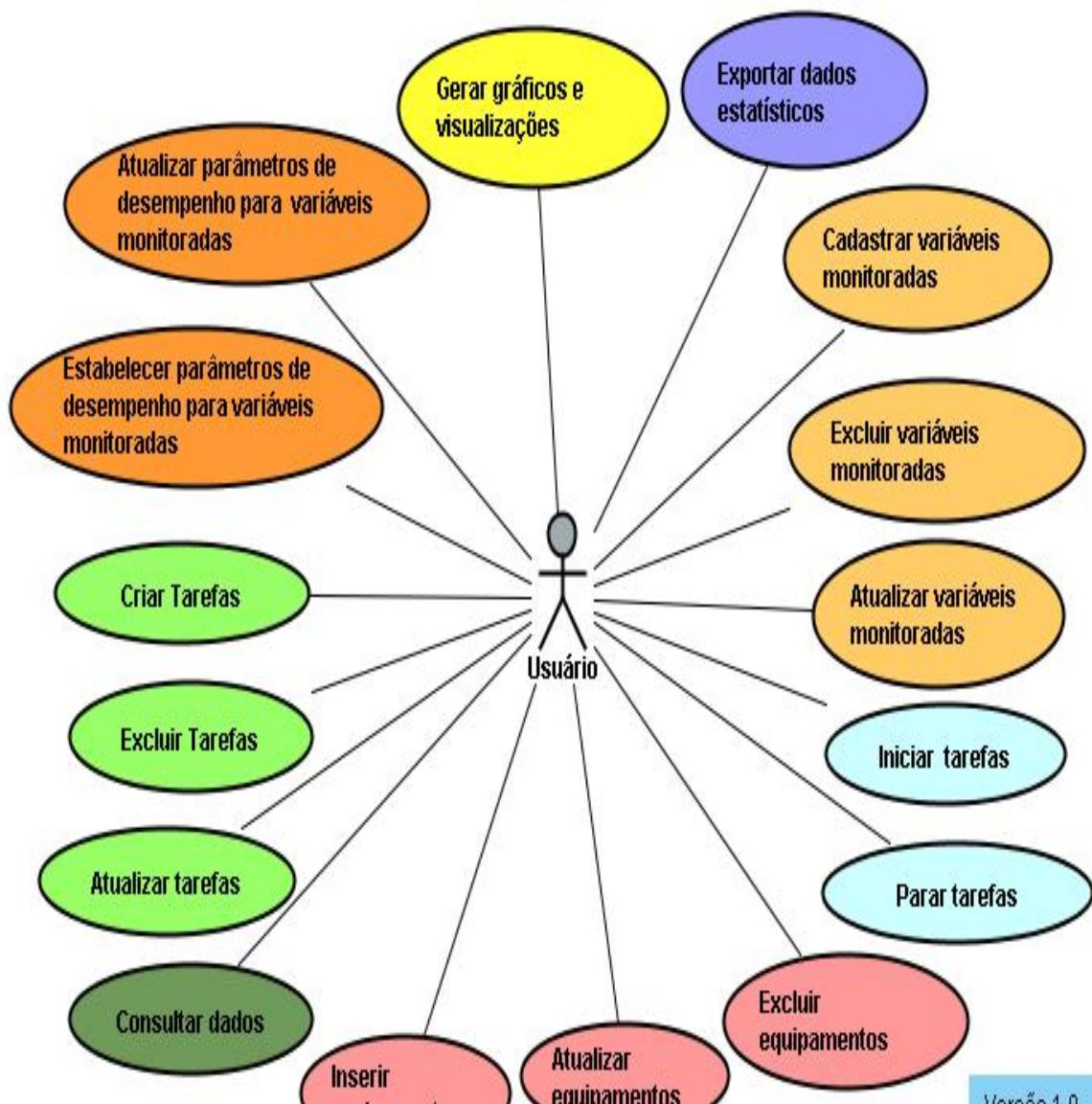


Figura 4.1: Casos de uso do sistema.

- *Task*: entidade que representa uma tarefa que é executada. Cada tarefa possui variáveis associadas, um equipamento, data de início, data de fim e frequência de leitura.
- *Reading*: entidade que representa as leituras efetuadas pelas tarefas. Cada leitura contém data em que foi realizada, valor lido, tarefa e variável associada.
- *Historic*: é uma entidade que representa o histórico de leituras já efetuadas.
- *Search*: é uma entidade que guarda todas as pesquisas efetuadas no sistema.
- *Users*: é uma entidade que contém os usuários do sistema. Somente usuários cadastrados podem acessar o sistema.

O pacote *model* contém, também, as classes DAO do sistema e enumerações utilizadas nas entidades. O modelo DAO é um modelo para persistir dados em banco de dados relacional que permite separar regras de negócio das regras de acesso a banco de dados. As classes de utilidades são enumerações para tipos de dados das entidades, como estado de uma tarefa, tipos de usuário, etc.

As classes do pacote *controller*, são as classes de controle a cada uma das entidades e às classes DAO. O pacote SNMP contém fachadas de acesso aos métodos SNMP (*get*, *set*, *getNext* e *walk*) e um leitor que lê variáveis de uma MIB em formato XML.

O pacote *util* do sistema contém as classes de escalonamento de leituras do sistema, de exportação de dados e classes de mensagens de erros e de avisos.

O pacote *graphics* contém as visualizações em OpenGL do sistema, são apresentadas janelas de escolha dos dados a serem visualizados e a seguir os dados são renderizados nas classes *Renderers* de cada pacote; são apresentadas visualizações em linhas, em espirais e em esferas.

O pacote *charts* contém os gráficos implementados utilizando as ferramentas Chart-Director e JFreeChart. São apresentados gráficos convencionais de tempo real e gráficos polar, de linhas e de barras estáticos com dados armazenados no banco de dados.

E o pacote *view* contém as janelas e interfaces gráficas do sistema, como inserção de equipamentos e de tarefas, edição de dados, consulta, alteração de senha, exportação de dados coletados de leituras SNMP e demais operações do sistema.

Na Figura 4.2 é apresentado o diagrama do banco de dados relacional gerado na ferramenta *Power\*Architect* [18], mostrando as tabelas e relacionamentos. As tabelas apresentadas ilustram como foram representadas as tabelas do banco de dados através do mapeamento das entidades (classes) do Hibernate. Cada entidade gerou uma tabela do banco de dados, além de uma tabela de relacionamento entre as entidades *Task* e *Variable*.

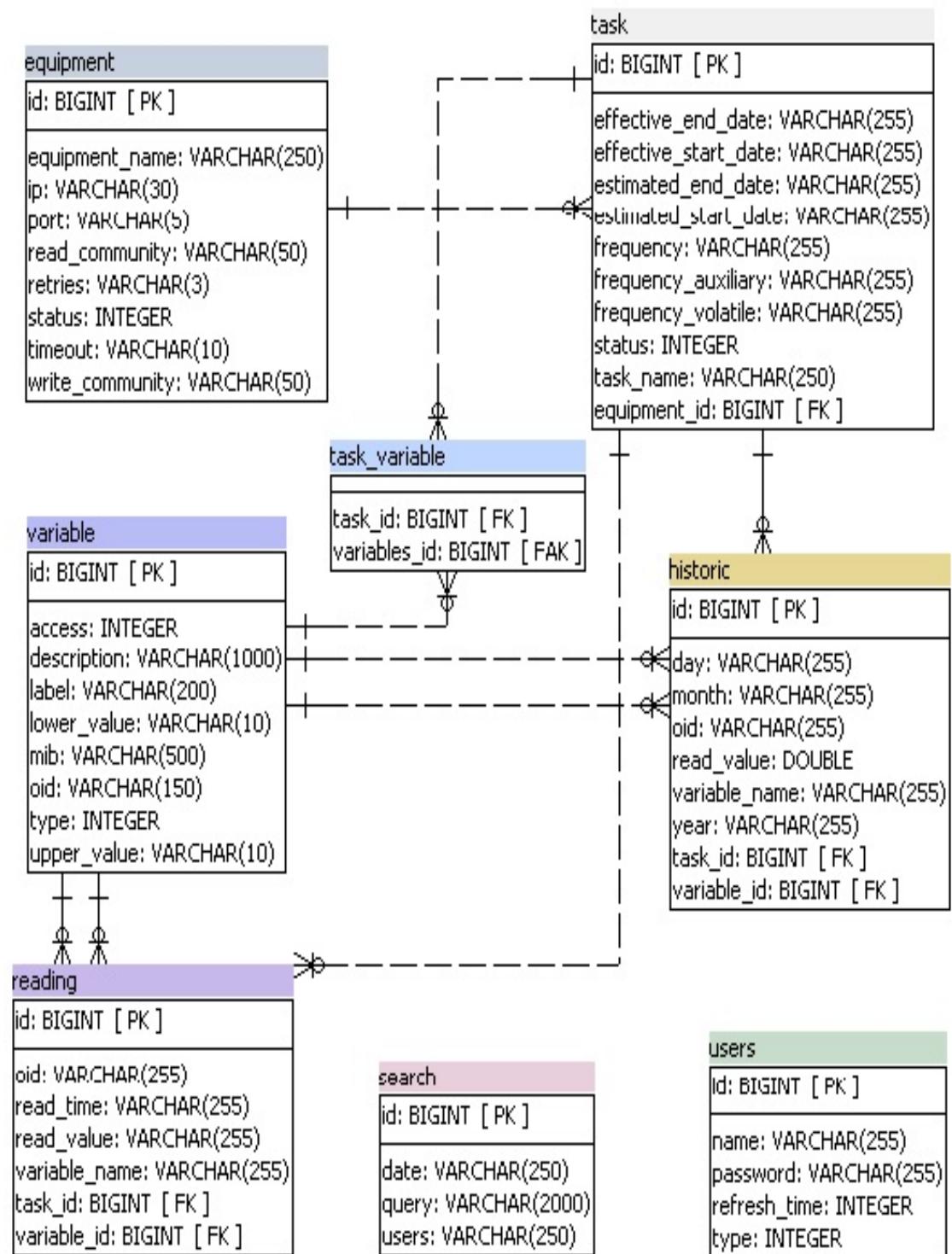


Figura 4.2: Modelagem do banco de dados do sistema.

## 4.4 Interface e funcionalidades

Nesta seção são apresentadas as interfaces gráficas e funcionalidades do sistema desenvolvido.

A Figura 4.3 mostra a janela principal do protótipo, a qual se tem acesso às tarefas que estão sendo monitoradas, criar, editar, parar e iniciar tarefas. Também se tem acesso as visualizações do sistema: visualizações OpenGL e gráficos convencionais. Na Figura 4.5, é possível visualizar a tela de inserção de tarefas e de variáveis associadas que serão lidas pelo escalonador de tarefas. Cada tarefa possui uma data de início, uma data de fim e uma frequência na qual será efetuada a leitura de todas as variáveis que forem associadas a ela.

Outras funcionalidades do protótipo são: a alteração e exclusão de tarefas, de variáveis e de equipamentos monitorados, a definição e alteração de valores de *thresholds* de cada variável, busca de dados no sistema, inserção e atualização de usuários, além da exportação de dados de leituras.

A Figura 4.4 apresenta a tela de inserção de equipamentos e a Figura 4.5 mostra a tela de inserção de tarefas e variáveis a serem monitoradas.

O anexo C apresenta o manual de utilização do protótipo e as visualizações são explicadas em detalhes na seção 4.5.2.

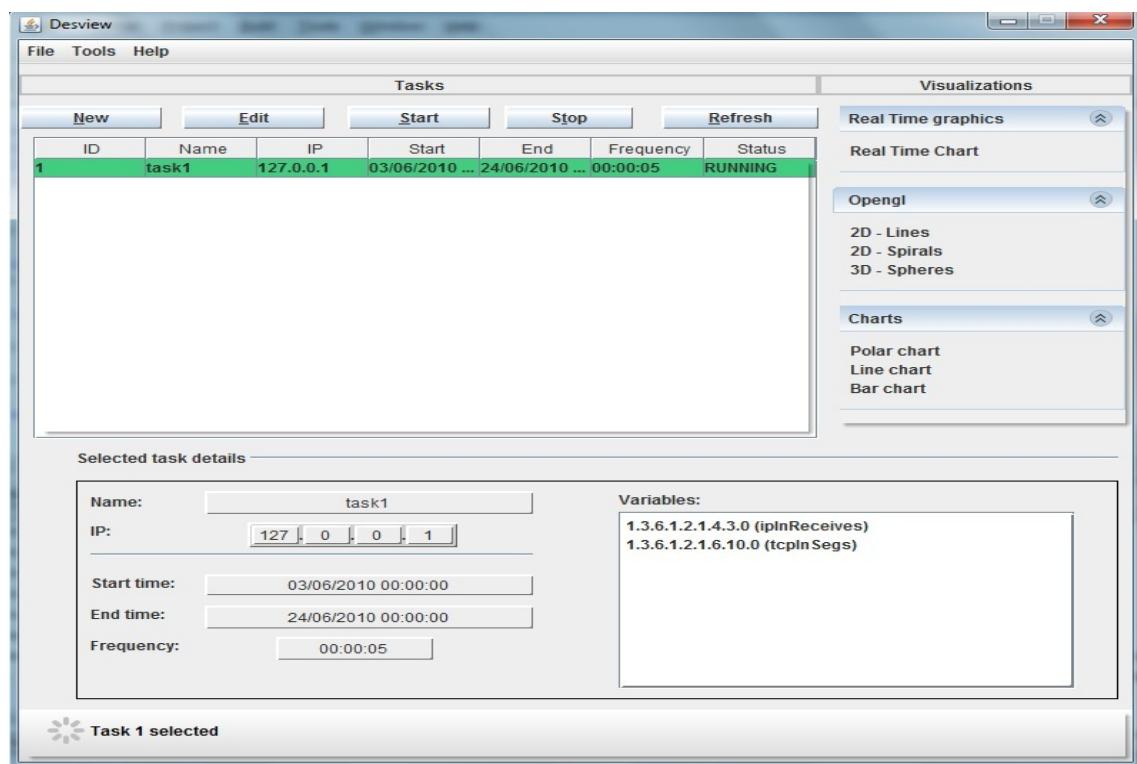


Figura 4.3: Tela principal do protótipo.

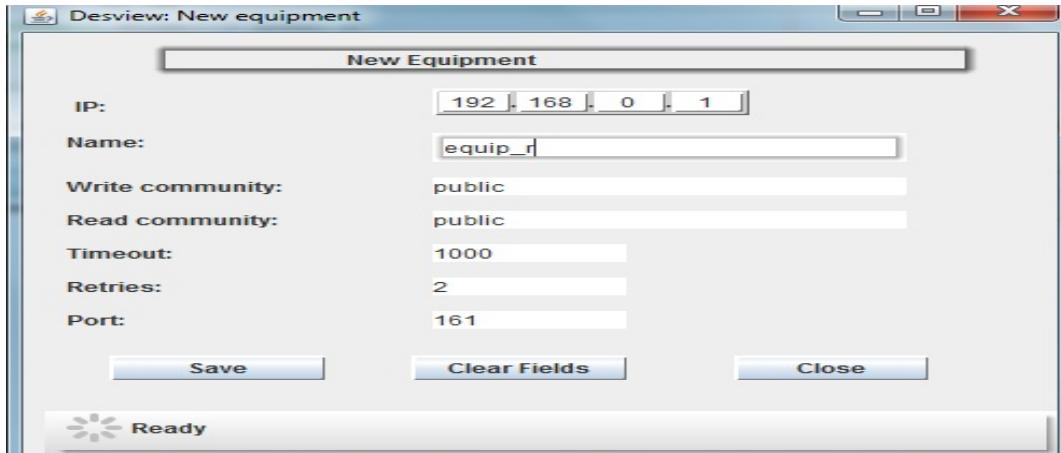


Figura 4.4: Tela de inserção de equipamentos.

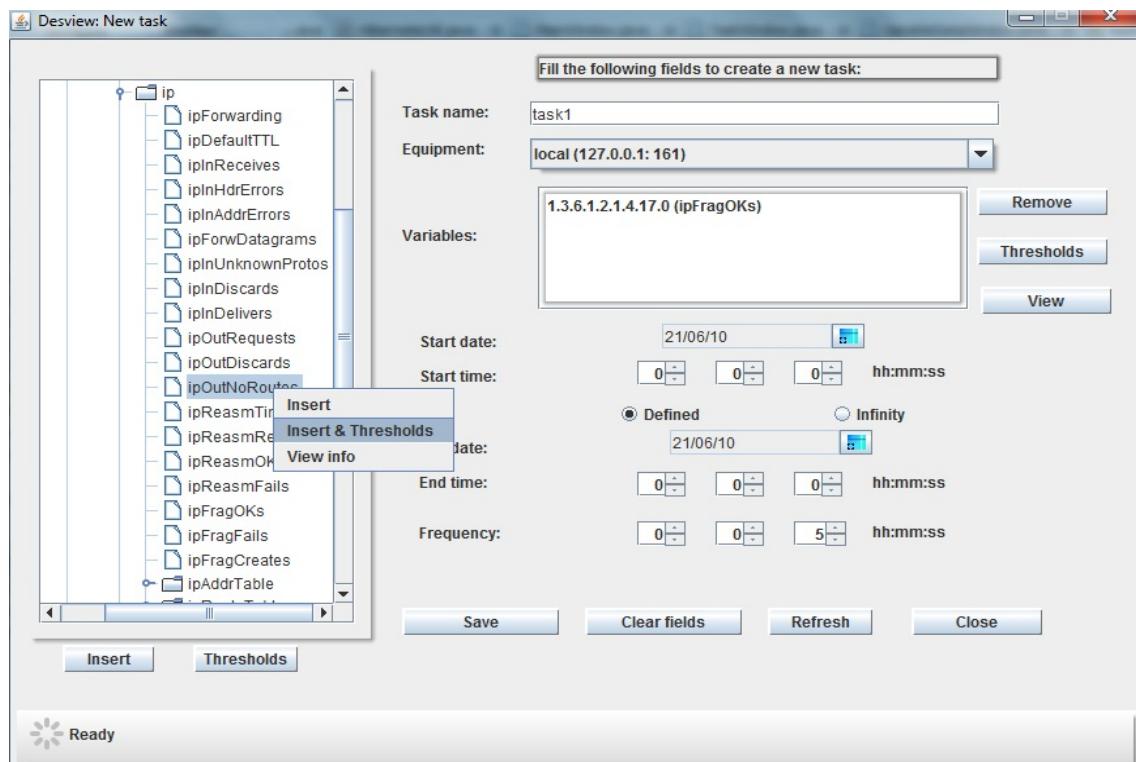


Figura 4.5: Tela de inserção de tarefas e de variáveis associadas.

## 4.5 Implementação

Nesta seção é apresentado como foi implementado o protótipo, ou seja, desde como foi implementada a infraestrutura de banco de dados, as fachadas de obtenção de dados SNMP, o escalonador de leituras e as visualizações implementadas no protótipo.

#### **4.5.1 Infraestrutura do sistema**

A infraestrutura do sistema implementado é composta dos seguintes mecanismos principais:

- Entidades que correspondem aos objetos armazenados no banco de dados. As classes são mapeadas utilizando o Hibernate que controla a inserção, a atualização e remoção de dados do banco de dados de cada uma das entidades.
- Fachada SNMP com os métodos de acesso às variáveis de MIB. O acesso às variáveis de uma MIB é feito através da biblioteca SNMP4J;
- Gerenciador de tarefas, que realiza o escalonamento, a leitura do valor das variáveis e posterior inserção no banco de dados. Como alternativa a este gerenciador de tarefas poderia ter sido utilizado um sistema RRDTool já implementado, tal como o Cacti, por exemplo, porém, seria necessário um estudo da forma de busca dos dados que são armazenados pelo Cacti, o que poderia ser inviável para o escopo apresentado. Além de utilizar como base de dados o MySQL, também foi observado que o armazenamento que o Cacti realiza das leituras ocorre em inúmeras tabelas com grande número de tabelas de relacionamentos. Por isso, optou-se por implementar uma base de dados mais simples e que se adequasse melhor ao trabalho desenvolvido. Como vantagem do uso de um RRDTool, pode-se citar o controle sobre o “estouro” da quantidade de dados e informações armazenadas no banco de dados, característica a qual o sistema implementado não possui;
- Janelas que permitem a inserção, a alteração e a exclusão de equipamentos, de tarefas e de variáveis, além da especificação e da atualização dos valores de *thresholds* das variáveis em monitoração. Também, a pesquisa dos dados armazenados, além de janelas que filtram dados para a geração das visualizações;
- Exportação de dados lidos em formato texto e em formato PDF (*Portable Document Format*);
- Sistema de *login*, com alterações de dados de usuários.

#### **4.5.2 Visualizações**

Nesta seção são apresentadas as visualizações que foram implementadas sobre os dados que são lidos pelas tarefas do protótipo implementado.

##### **Gráfico convencionais**

Os gráficos convencionais implementados no protótipo são detalhados a seguir e exemplos são apresentados na seção 4.6 - estudo de caso.

- Gráfico de barras: exibe as séries como conjuntos de barras horizontais. No protótipo, os dados que podem ser visualizados são a média, o valor mínimo e o valor máximo de cada uma das variáveis que estão em monitoração.
- Gráfico polar: exibe uma série como um conjunto de pontos agrupados por categoria em um círculo de 360°. É um gráfico ideal para representar séries temporais cíclicas, isto é, séries temporais que apresentam em seu desenvolvimento determinada periodicidade. Quanto mais distante o segmento de reta está do centro, maior é o seu valor. No protótipo, os dados que podem ser visualizados através de gráfico polar são a média dos valores lidos de uma variável durante determinado mês de um ano.
- Gráfico de linhas: um gráfico de linha é usado para representar grandes quantidades de dados que variam em um período de tempo contínuo. No protótipo, os dados que podem ser visualizados através de gráfico de linhas são a média dos valores lidos de uma variável em um ano de monitoração.

## Gráfico 2D em linhas

O gráfico em OpenGL em linhas tem o seguinte objetivo: dado um conjunto de variáveis sendo monitoradas por uma tarefa, deseja-se a partir dos *thresholds* inferior e superior de cada uma, verificar através de linhas coloridas quando uma das variáveis passa dos intervalos definidos. Assim, a cor da linha informa se o valor atual lido está abaixo, no intervalo, ou acima dos valores definidos como *thresholds*.

## Gráfico 2D em espiral

Durante o desenvolvimento do gráfico 2D em linhas verificou-se que o desenho de linhas não se adequava à grande quantidade de dados que precisam ser plotados, dessa forma, desenvolveu-se o gráfico OpenGL em espirais que tem o mesmo objetivo do gráfico 2D em formato de linhas, porém, o gráfico em espiral se propõe a mostrar grande quantidade de dados num espaço relativamente pequeno. Assim, caso se deseje acompanhar o andamento do gráfico em linhas é necessário acompanhar o desenho movimentando a câmera ou então redesenhando novamente, o gráfico em espiral facilita essa atualização, pois ao terminar o espaço de desenho do gráfico, apenas inicia-se novamente a espiral.

## Gráfico 3D em esferas

O gráfico 3D de esferas implementado no protótipo tem por objetivo, a partir de uma tarefa que se deseja visualizar, plotar um gráfico 3D informando as médias das variáveis nos meses do ano e o estado da média em relação aos *thresholds*, ao clicar em uma das variáveis em determinado mês do ano é aberto um novo gráfico com média dos dias do mês informando a média em relação aos *thresholds*.

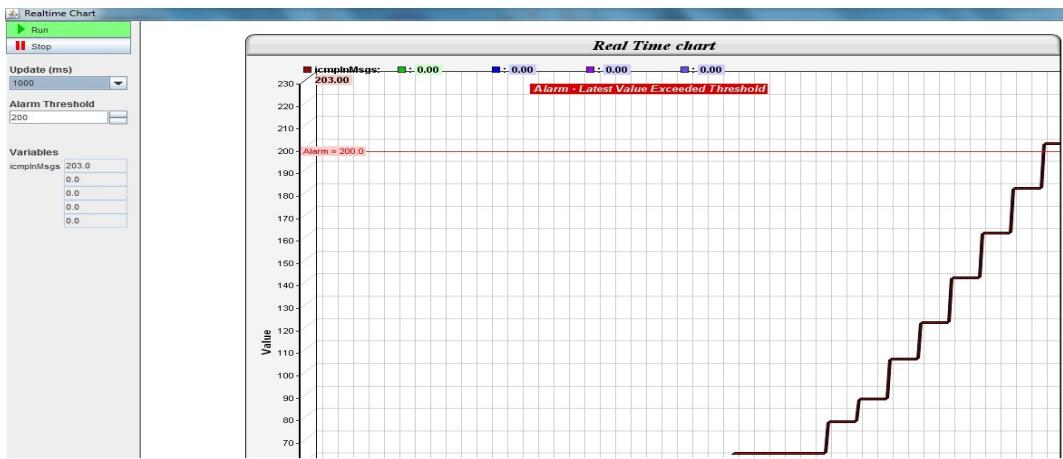


Figura 4.6: Gráfico de tempo real obtido, informando que a variável em monitoração passou do *threshold* informado.

## 4.6 Estudo de caso

O sistema implementado poderia ser utilizado em alguns casos como: controle de taxas de erros (BERT - *bit error rate test*) em equipamentos de redes, que possui um *threshold* superior e se deseja observar se o valor corrente ultrapassa o valor máximo desejado durante a realização do teste. Outro caso útil é a monitoração de sensores de temperaturas, por exemplo. Cada sensor ao conter um agente SNMP embutido, torna-se apto a ser monitorado por um sistema de desempenho que controle a temperatura do mesmo.

Assim, nesta seção são apresentados alguns estudos de caso que poderiam ser utilizados como o sistema de gerenciamento de desempenho desenvolvido e que mostrem a utilidade do protótipo. São utilizadas variáveis da MIB-2 para ilustrar a sua monitoração, mas seria interessante usar variáveis de MIBs proprietárias, tal como variáveis de BERT, ou então temperaturas e outros sensores.

Dada a seguinte situação: utilizando-se um equipamento de rede, deseja-se monitorar o desempenho da variável *icmpInMsgs* da MIB-2. Assim, define-se um *threshold* inferior de 100 e um *threshold* superior de 200 pacotes. Dessa forma será utilizado o gráfico de tempo real para monitorar esta variável, e caso o valor lido ultrapasse o *threshold* estabelecido, uma espécie de alarme é mostrado no gráfico. Dessa forma, um exemplo do gráfico obtido monitorando essa variável é apresentado na Figura 4.6, tornando fácil verificar que determinada variável passou o *threshold* superior definido, ou ainda está abaixo do valor definido. Neste trabalho, foi utilizado o desempenho com métricas de *thresholds*, mas seria interessante poder monitorar, também, métricas através de sequências de cálculos os quais gerariam taxas e frequências como métricas, por exemplo, taxa de *bytes* que entram em determinada interface de um equipamento de rede, mas por limitação de tempo e de escopo, optou-se por utilizar métricas mais simples com o objetivo de validar o trabalho, mesmo não sendo tão significativas.

Utilizando-se a mesma variável apresentada se pode verificar agora qual foi a média

de valores lidos durante um mês de determinado ano. Assim, foi gerado o gráfico polar da Figura 4.7, contendo a média de cada dia do mês, no qual é possível verificar em quais dias do mês o sistema esteve mais sobrecarregado respondendo a requisições. Na Figura 4.8 tem-se a média, o valor máximo e o valor mínimo dentre as leituras efetuadas. Já a Figura 4.9, apresenta durante os meses de determinado ano, qual foi a média dos valores lidos. Dessa forma, consegue-se por meio dos gráficos convencionais apresentados verificar quais foram os pontos em que determinado equipamento foi muito requisitado durante os meses de um ano, e a seguir verificar dentro deste mês qual foram os dias mais críticos.

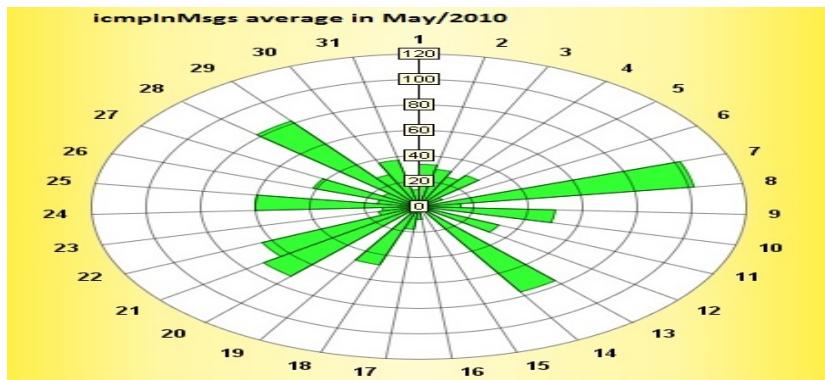


Figura 4.7: Gráfico polar obtido a partir de leituras efetuadas em uma variável durante um mês.

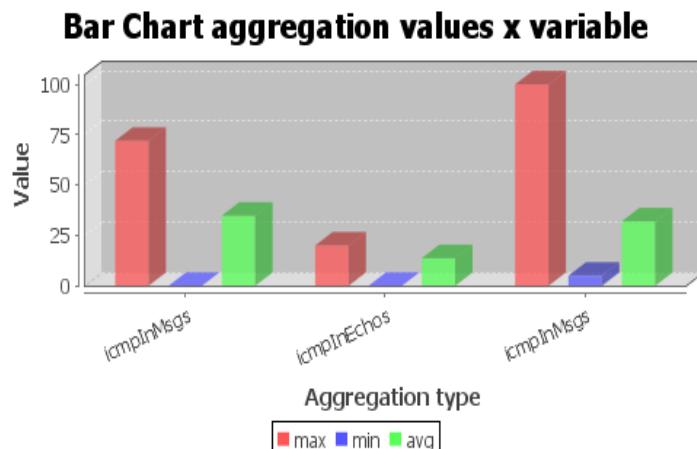


Figura 4.8: Gráfico de barras contendo o valor mínimo, o valor máximo e a média dos valores lidos de três variáveis.

Com os gráficos em OpenGL implementados deseja-se verificar como eles podem auxiliar na monitoração do desempenho da variável em relação aos gráficos convencionais. Os gráficos em OpenGL implementados monitoram variáveis a partir do seu valor atual com os *thresholds* superior e inferior definidos para cada uma em cada tarefa, ou seja, uma



Figura 4.9: Gráfico de linhas contendo a média de todos os valores lidos durante os meses de determinado ano.

variável pode ter diferentes valores de *thresholds* em diferentes tarefas. As cores presentes nos gráficos tem o seguinte significado:

- Se o valor atual lido for menor que o *threshold* inferior, então desenha o gráfico com a cor amarela.
- Se o valor atual lido for maior que o *threshold* superior, então desenha o gráfico com a cor vermelha.
- Se o valor atual lido for maior ou igual que o *threshold* inferior e menor ou igual que o *threshold* superior então desenha o gráfico com a cor verde.

Assim, inicialmente temos o gráfico 2D de linhas (Figura 4.10), no qual é possível verificar em um gráfico que determinada variável saiu de um intervalo de *threshold* e entrou em outro. O mesmo exemplo foi monitorado utilizando o gráfico em espiral e pode-se observar que a espiral se comportou de forma muito mais satisfatória frente a essas mudanças de estado, principalmente devido ao fato de ela comportar muito mais dados em um espaço menor (Figura 4.11).

O objetivo do gráfico de 3D é a partir de uma tarefa mostrar todas as variáveis da tarefa em uma linha de tempo contendo os 12 meses do ano. A seguir é analisada a média dos dados coletados em cada um dos meses e a cor da esfera relativa à variável no mês é colorida de acordo com os critérios apresentados anteriormente. Assim, como no exemplo mostrado na Figura 4.12, na qual se tem um mês com a média acima dos valores desejados, pode-se clicar na esfera correspondente e passar a verificar o estado da variável nos dias do mês (Figura 4.13). Dessa forma, é verificado em que dia a média dos valores foi superior ou inferior aos valores de *thresholds*. Este tipo de gráfico não utiliza dados de leituras em tempo real como os outros gráficos OpenGL, ele utiliza a média de todos os valores presentes no banco de dados para as variáveis que estão sendo visualizadas. No exemplo



Figura 4.10: Gráfico OpenGL 2D - linhas.

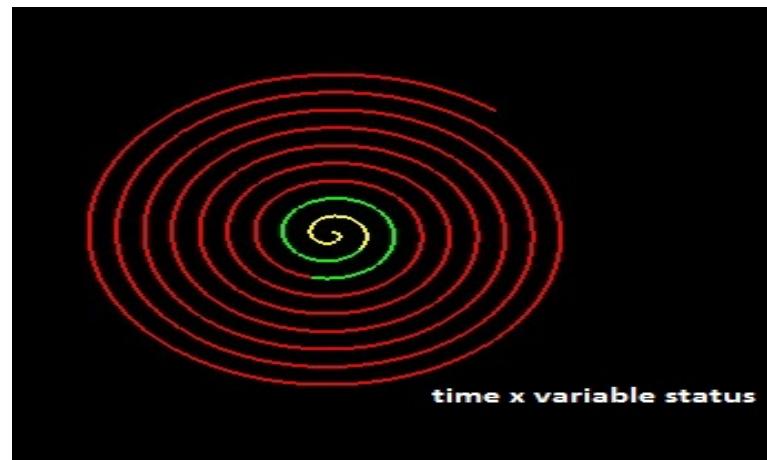


Figura 4.11: Gráfico OpenGL 2D - espirais.

apresentado, foram adicionadas mais duas variáveis de forma a visualizar como ficaria o gráfico com mais de uma variável.

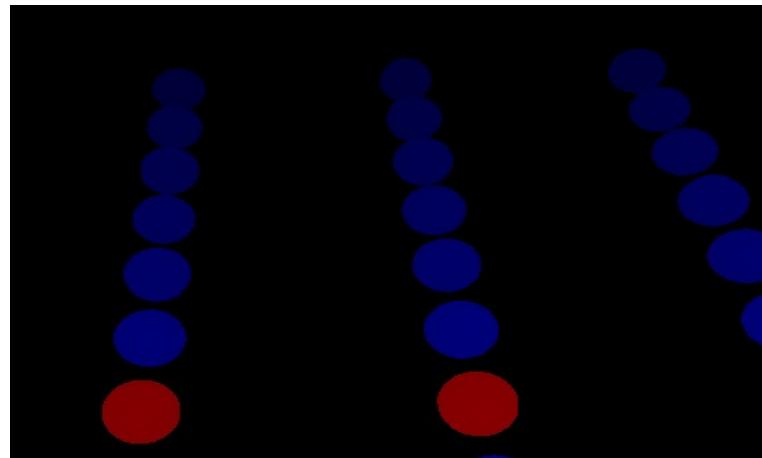


Figura 4.12: Gráfico OpenGL 3D - esferas nível de varíaveis distribuídas nos meses do ano.

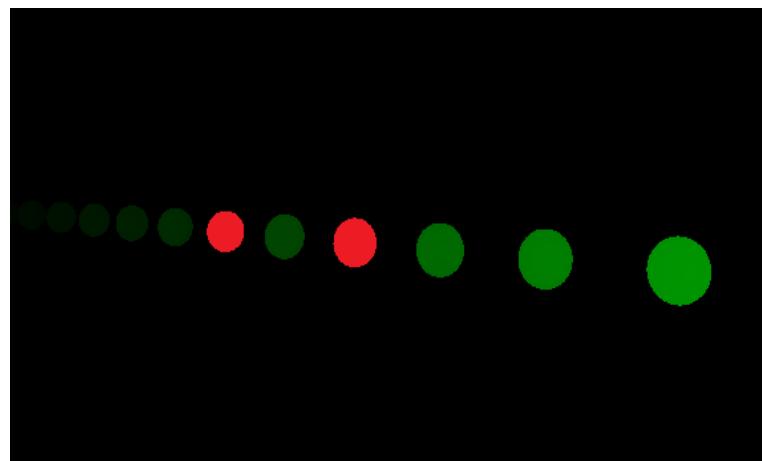


Figura 4.13: Gráfico OpenGL 3D - esferas mostrando os dias do mês para a variável selecionada.

## 5 Conclusões e trabalhos futuros

Ao finalizar este Trabalho de Conclusão pode-se concluir que os conhecimentos adquiridos a partir deste trabalho, foram muito importantes para o desenvolvimento profissional e acadêmico, pois foi possível explorar bibliotecas e tecnologias que não haviam sido exploradas em sala de aula. Assim, com a implementação deste protótipo, iniciou-se um estudo mais aprofundado de uma área da Gerência de redes de computadores pouco explorada frente a outras áreas, que são consideradas, algumas vezes, mais importantes no gerenciamento de redes, tais como gerenciamento de falhas e de configuração.

Com base no problema apresentado, as visualizações implementadas têm por objetivo ilustrar formas de tornar o gerenciamento de desempenho mais fácil de ser “gerenciado”, ou seja, sem a necessidade de visualizar grandes quantidades de dados tabulados, mas sim, visualizar os mesmos dados através de gráficos e visualizações 2D e 3D que facilitem e auxiliem a verificação de que determinada variável está fora dos padrões de desempenho considerados aceitáveis.

Como trabalhos futuros, pode-se apontar como tarefas, aprofundar o tema em relação às visualizações implementadas, além de utilizar variáveis de MIBs que retornem dados relevantes ao gerenciamento de desempenho, presentes na maioria de vezes em MIBs proprietárias. Além disso, pode-se indicar, a implementação de novas formas de visualizações e o processamento de informações vindas da MIB e através desse processamento obter dados estatísticos sem utilizar variáveis específicas de desempenho.

Pode-se citar como semelhanças do protótipo implementado com os trabalhos relacionados apresentados: possuir um escalonador de leituras, utilizar visualizações através de gráficos convencionais. Como diferencial neste sistema, apresentamos algumas ideias de gráficos utilizando a biblioteca OpenGL como apoio para o desenvolvimento de gráficos mais elaborados e com possibilidade de mais interação, além disso, foram exploradas diversas formas geométricas proporcionadas pela VISINFO, na busca por melhores formas de representar os dados coletados da rede e que são plotados em diversas formas geométricas apresentadas. Como desvantagens da protótipo desenvolvido, pode-se citar a falta de um agente de descoberta automático, ou seja, toda rede tem que ser adicionada manualmente e também o fato de não ser um sistema Web, ou seja, por ser um aplicativo Java, dificulta o acesso por múltiplos usuários.

# Referências Bibliográficas

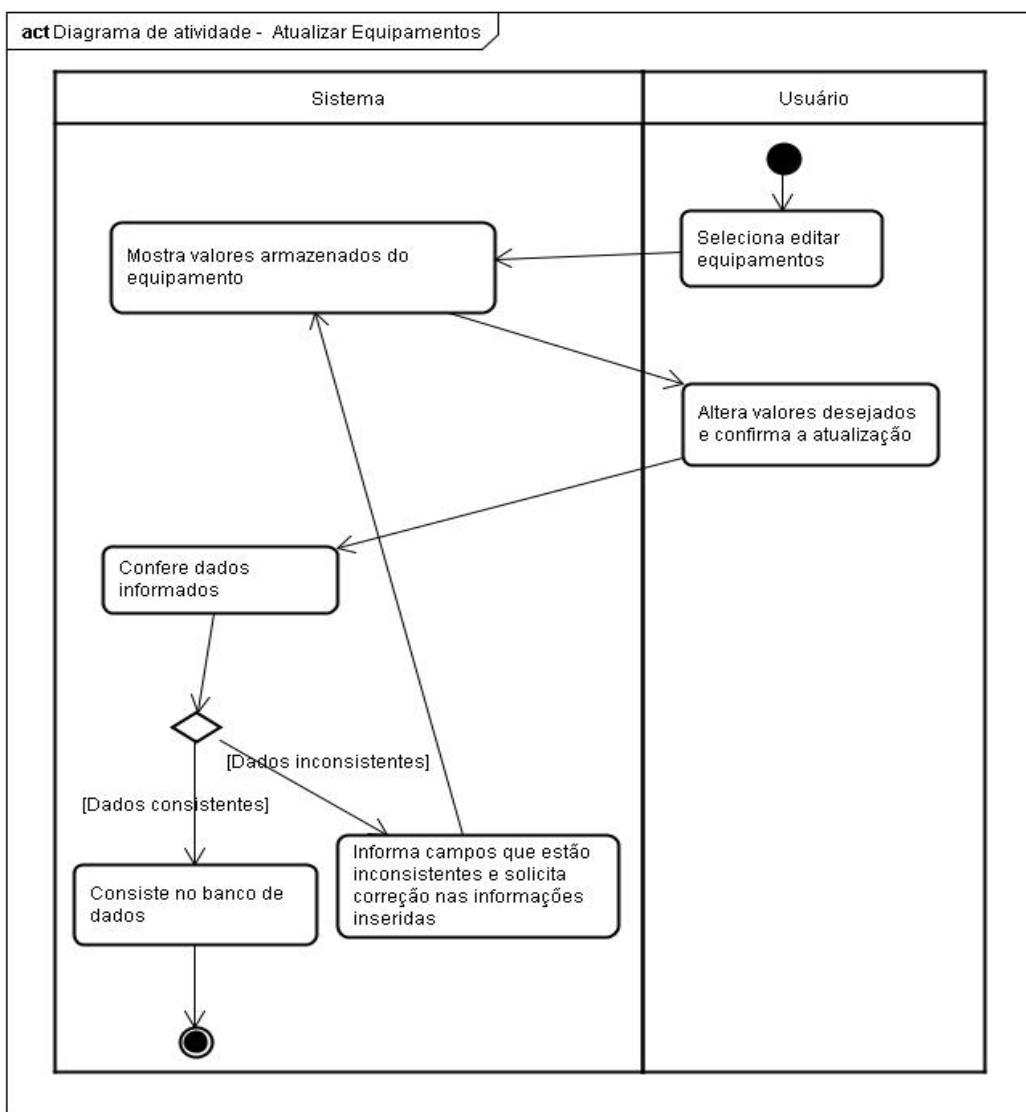
- [1] CARD, S.K.; MACKINLAY, J.D.; SHNEIDERMAN, B. **Readings in Information Visualization - Using Vision to Think.** San Francisco: Morgan Kaufmann Publ., 1999.
- [2] ESTIVALET, Luiz Fernando. **O Uso de Ícones na Visualização de Informações.** PPGC da UFRGS, 2000.
- [3] Wikipédia, a enclopédia livre (2009). **Gerência de redes.** Acesso em 09 de setembro de 2009, em: [http://pt.wikipedia.org/wiki/Ger%C3%Aancia\\_de\\_redes](http://pt.wikipedia.org/wiki/Ger%C3%Aancia_de_redes).
- [4] NASCIMENTO, Hugo; FERREIRA, Cristiane. **Visualização de Informações - Uma Abordagem Prática,** XXV Congresso da Sociedade Brasileira de Computação, cap. 2, 2005.
- [5] Cacti (2009). **Cacti: The Complete RRDTool-based Graphing Solution.** Acesso em 09 de setembro de 2009, em: <http://www.cacti.net>.
- [6] Nagios (2009). **Nagios - The Industry Standard in IT Infrastructure Monitoring.** Acesso em 09 de setembro de 2009, em: <http://www.nagios.org>.
- [7] SALVADOR, Ewerton Monteiro; GRANVILLE, Lisandro Zambenedetti. **An Investigation of Visualization Techniques for SNMP Traffic Traces.** In: IEEE/IFIP Network Operations and Management Symposium (NOMS 2008), 2008, Salvador. Proceedings of the 2008 IEEE/IFIP Network Operations and Management Symposium, 2008.
- [8] KAUER, A. Da Silva; MEIGUINS, B. S. ; CARMO, R. M. C. Do; GARCIA, M. B.; MEIGUINS, A. S. G. **An Information Visualization Tool with Multiple Coordinated Views for Network Traffic Analysis.** In: Information Visualization, 2008. IV '08. 12th International Conference, 2008, London. IV '08. 12th International Conference, 2008. p. 151-156.
- [9] FREITAS, Carla M. D. S. et alii. **Introdução à Visualização de Informações.** PPGC da UFRGS, 2001.

- [10] HABER, R.B. e McNABB, D. A. **Visualization Idioms: A conceptual model for scientific visualization systems**. Visualization in Scientific Computing, 1990. pp. 74-93.
- [11] BRISA. **Gerenciamento de Redes: Uma Abordagem de Sistemas Abertos**. São Paulo: Makron Books, 1<sup>a</sup> ed., 1993.
- [12] Java OpenGL (2009). **JOGL: Java OpenGL**. Acesso em 18 de setembro de 2009, em: <https://jogl.dev.java.net/>.
- [13] PostgreSQL (2009). **PostgreSQL: The world's most advanced open source database**. Acesso em 18 de setembro de 2009, em: <http://www.postgresql.org>.
- [14] SNMP4J (2009). **SNMP4J - Free Open Source SNMP API for Java**. Acesso em 18 de setembro de 2009, em: <http://www.snmp4j.org>.
- [15] MRTG (2009). **MRTG - The Multi Router Traffic Grapher**. Acesso em 09 de outubro de 2009, em: <http://oss.oetiker.ch/mrtg>.
- [16] RRDTool, Logging & Graphing (2009). **About RRDTool**. Acesso em 09 de outubro de 2009, em: <http://oss.oetiker.ch/rrdtool>.
- [17] NetBeans (2009). **NetBeans**. Acesso em 10 de outubro de 2009, em: <http://www.netbeans.org>.
- [18] Power\*Architect (2009). **SQL Power - Power\*Architect Data Modeling Tool**. Acesso em 18 de outubro de 2009, em: <http://www.sqlpower.ca/page/architect>.
- [19] Jude Community (2009). **JUDE/Community - Free UML Modeling Tool**. Acesso em 10 de outubro de 2009, em: <http://jude.change-vision.com/jude-web/product/community.html>.
- [20] INSELBERG, A. **Multidimensional detective**. In IEEE Symposium on Information Visualization (InfoVis '97), pages 100-107, Washington - Brussels - Tokyo. IEEE. October, 1997.
- [21] SILVA, F. C. (2009). **Visualização de Informações**. Acesso em 10 de outubro de 2009, em: [http://graphs.ucpel.tche.br/luzzardi/VisInfo\\_Fred.ppt](http://graphs.ucpel.tche.br/luzzardi/VisInfo_Fred.ppt).
- [22] Switch (2009). **Serving Swiss Universities**. Acesso em 10 de outubro de 2009, em: <http://www.switch.ch/network/operation/statistics/geant2.html>.
- [23] RDI - Rede de Informática (2009). **Prisma - Visual Business Intelligence**. Acesso em 11 de outubro de 2009, em: <http://www.redeinformatica.com.br/downloads/PRISMA.pdf>.
- [24] SPECIALSKY, E. S. **Gerência de redes de computadores**. Acesso em 11 de outubro de 2009, em: [http://www.malima.com.br/article\\_read.asp?id=279](http://www.malima.com.br/article_read.asp?id=279).

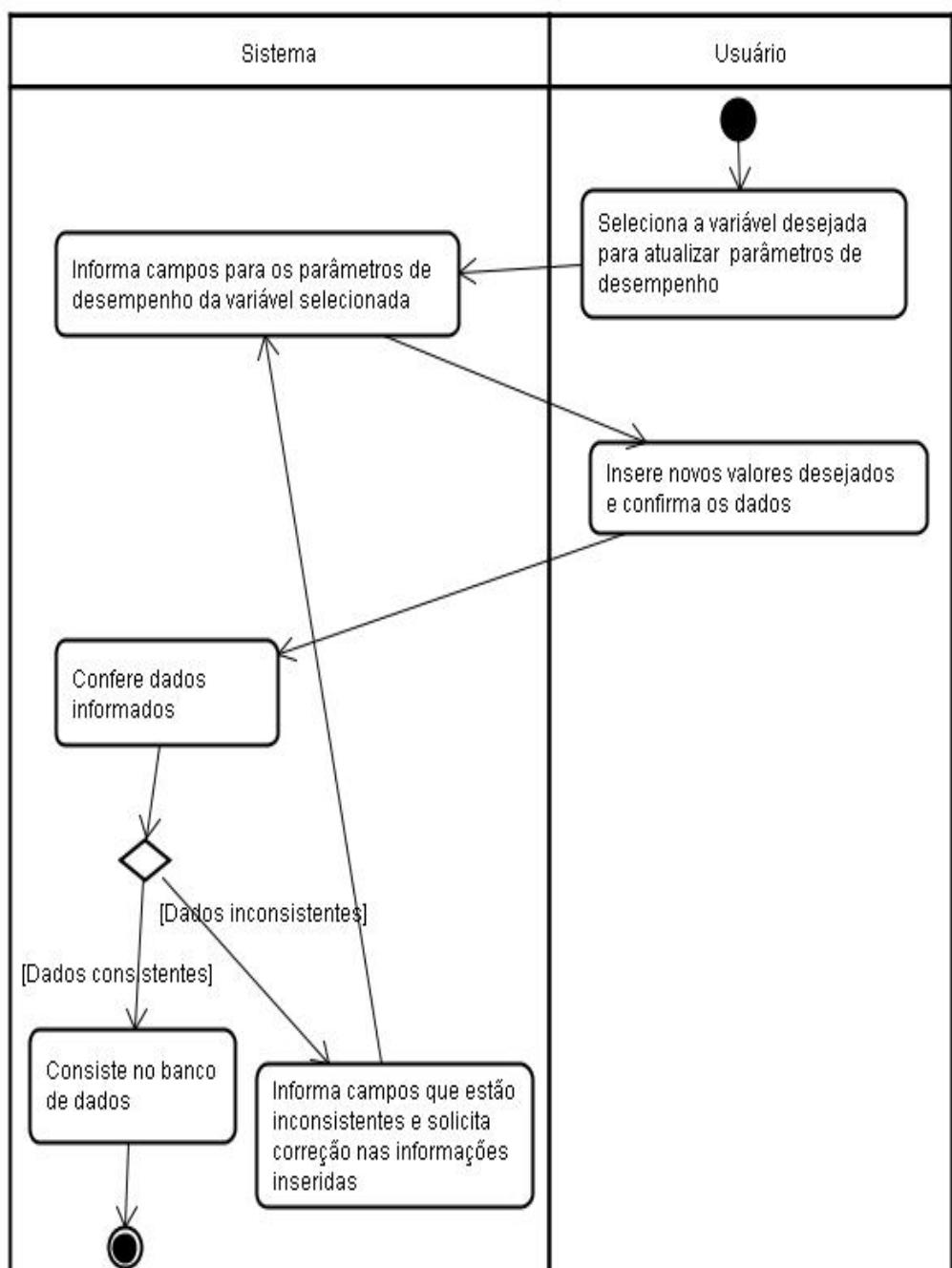
- [25] ARTOLA, E. S.; TAROUCO, L. M. R. **Olho Vivo - Sistema especialista para gerência pró-ativa remota**. PPGC da UFRGS, 1996.
- [26] RFC (2009). **Request for Comments (RFC)**. Acesso em 10 de outubro de 2009, em: <http://www.ietf.org/rfc.html>.
- [27] NAYAK, T.; NEOGI, A.; KOTHARI, R. **Visualization and Analysis of System Monitoring Data using Multi-resolution Context Information**. IEEE Transactions on Network and Service Management. Volume 5. Issue 3. September, 2008.
- [28] MAURO, D.; SCHMIDT, K. **Essential SNMP**. 2.ed. Campus O'Reilly, 2001.
- [29] Wikipédia, l'encyclopédie libre (2009). **Réseau de neurones artificiel**. Acesso em 15 de outubro de 2009, em: [http://fr.wikipedia.org/wiki/Réseau\\_de\\_neurones](http://fr.wikipedia.org/wiki/Réseau_de_neurones).
- [30] COHEN, M.; MANSSOUR, I. **OpenGL: uma abordagem prática e objetiva**. São Paulo: Novatec, 2006.
- [31] GERMANO, T. **Self-organizing map**. Acesso em 15 de outubro de 2009, em: <http://davis.wpi.edu/~matt/courses/soms>.
- [32] JFreeChart. **JFreeChart: A free Java chart library**. Acesso em 20 de novembro de 2009, em: <http://www.jfree.org/jfreechart>.
- [33] ChartDirector. **ChartDirector: Professional chart component for windows and web applications**. Acesso em 20 de novembro de 2009, em: <http://www.advsofteng.com>.
- [34] Swingx. **Swingx: Swing components extensions**. Acesso em 16 de junho de 2010, em: <https://swingx.dev.java.net>.
- [35] Quartz. **Quartz - Job scheduler**. Acesso em 16 de junho de 2010, em: <http://www.quartz-scheduler.org>.
- [36] Hibernate. **Hibernate - Relational Persistence for Java**. Acesso em 16 de junho de 2010, em: <http://www.hibernate.org>.
- [37] OpenNMS. **OpenNMS - Enterprise-grade open-source network management**. Acesso em 25 de junho de 2010, em: <http://www.opennms.org>.
- [38] OYAMA, Cibele. **OpenNMS: uma visão geral**. RNP, dezembro de 2006. Acesso em 25 de junho de 2010, em: [http://www.rnp.br/\\_arquivo/sci/2002/openNMS.pdf](http://www.rnp.br/_arquivo/sci/2002/openNMS.pdf).
- [39] JOUVIE, Jérôme. **Graphic Engine API**. Acesso em 26 de junho de 2010, em: <http://jerome.jouvie.free.fr>.
- [40] SANTOS, Aldri Luiz. **Gerência de redes de computadores**. Acesso em 26 de junho de 2010, em: <http://www.inf.ufpr.br/aldri>.

# Anexo A — Diagramas de Atividades

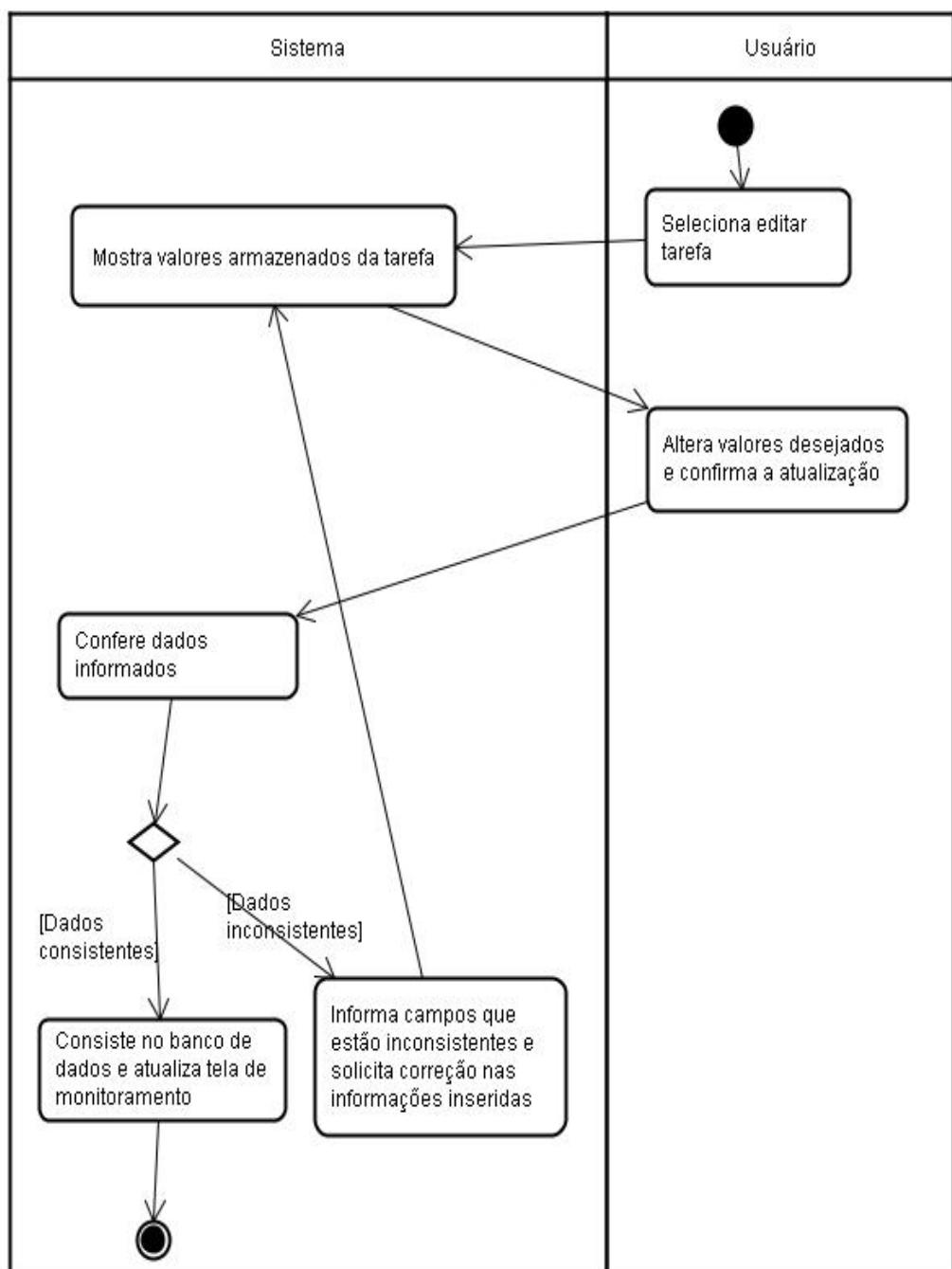
Neste anexo são apresentados os diagramas de atividades UML modelados para a implementação do sistema. São apresentados os diagrama de atividade detalhando, assim, cada caso de uso do sistema.



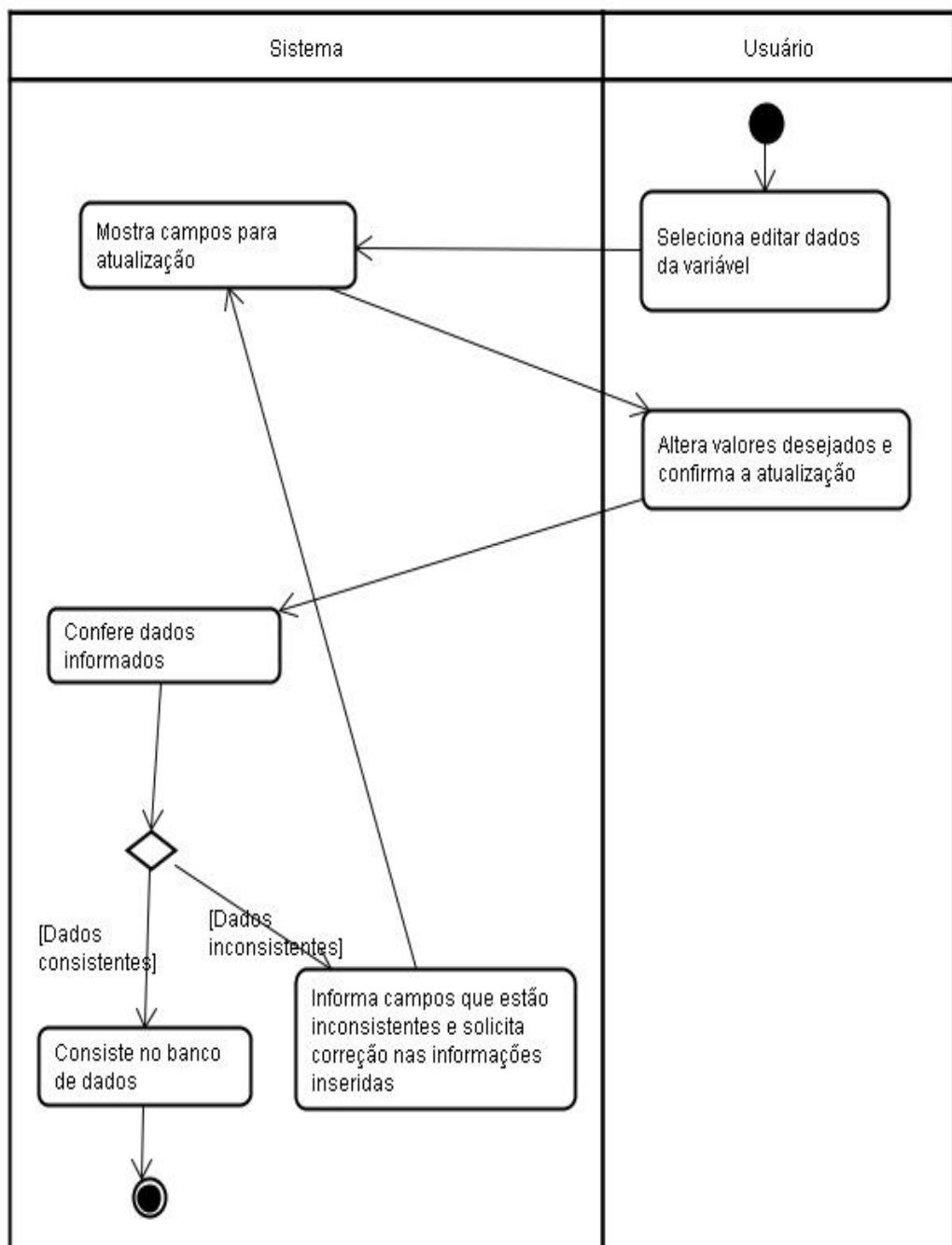
actDiagrama de atividade - Atualizar parâmetros de desempenho para variáveis monitoradas



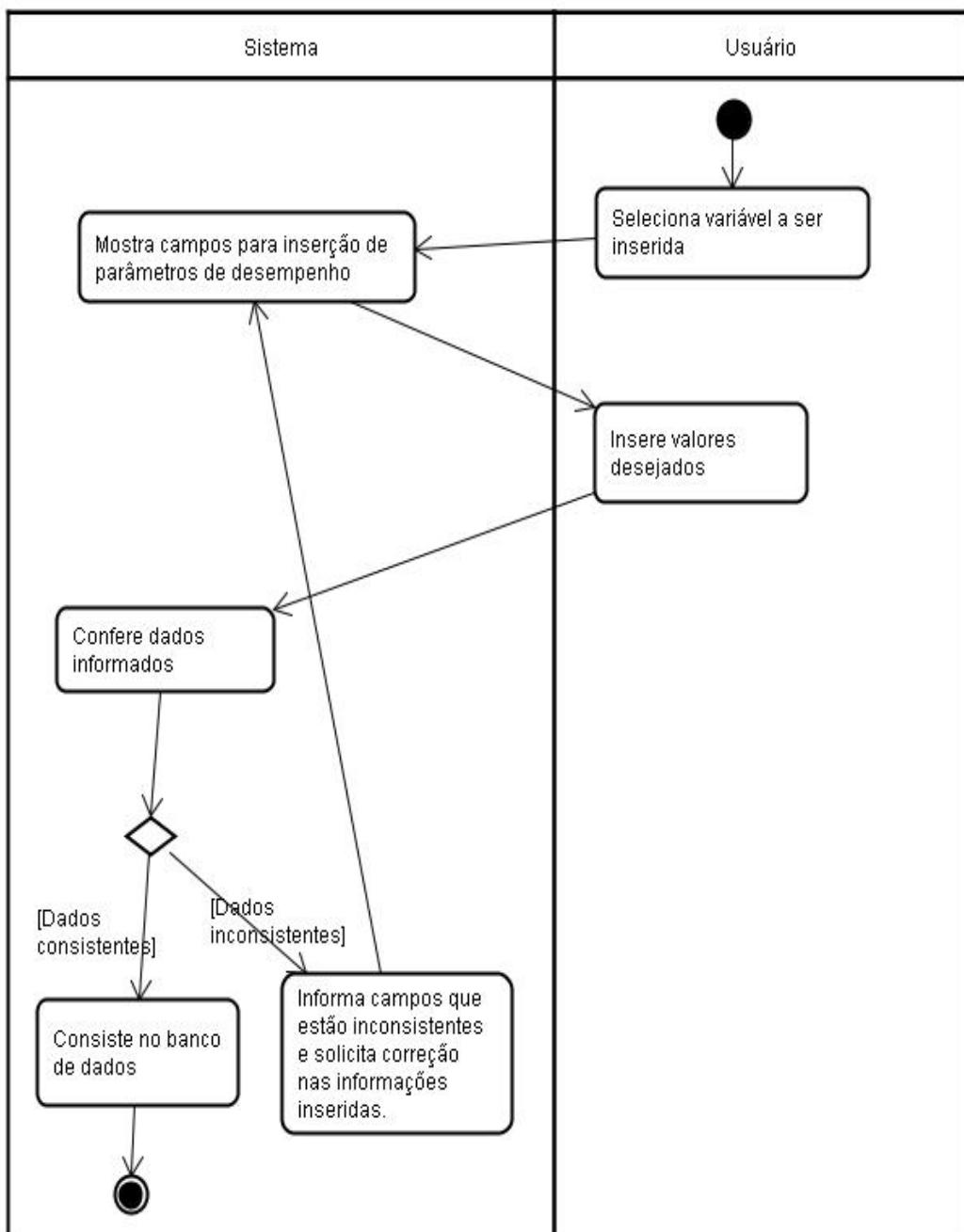
actDiagrama de atividade - Atualizar tarefas



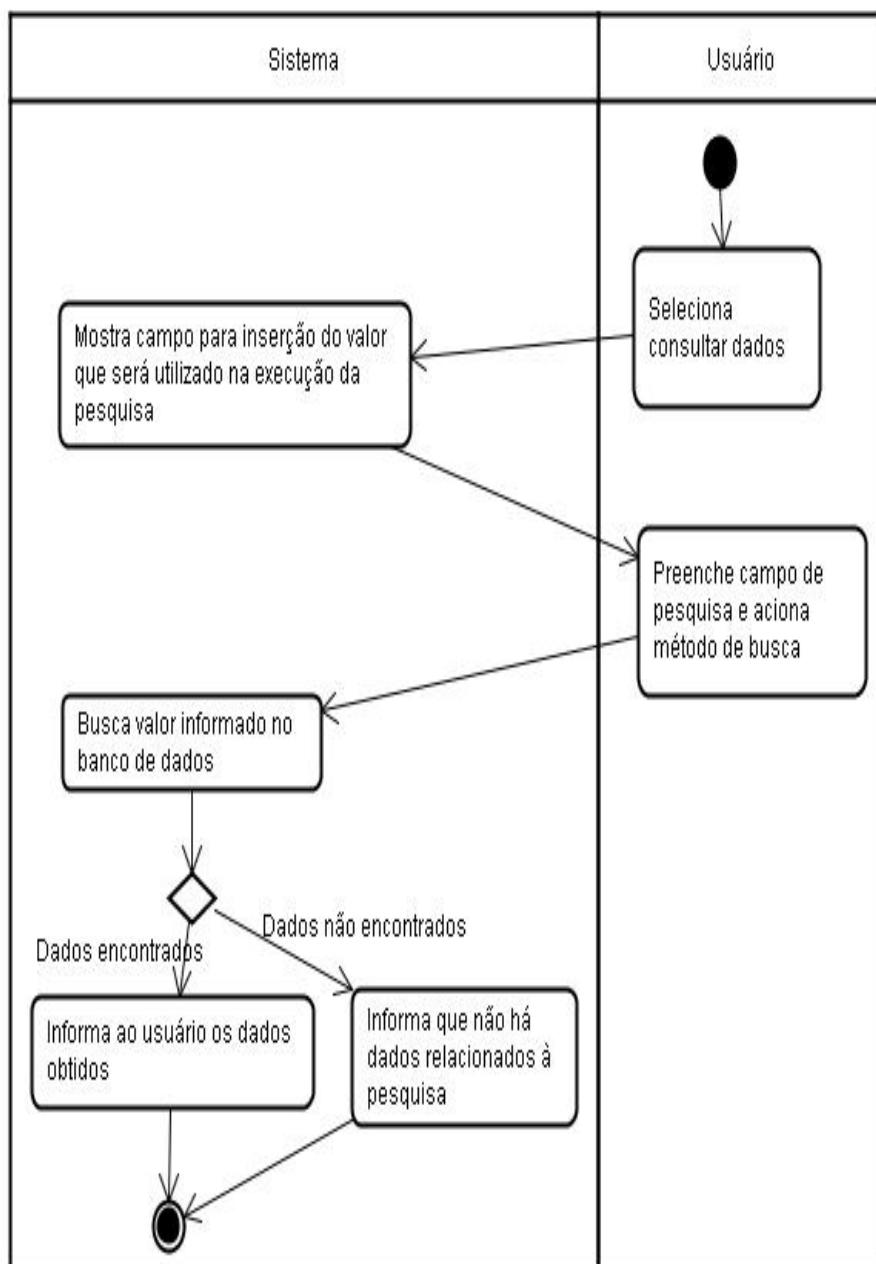
actDiagrama de atividade - Atualizar variáveis monitoradas



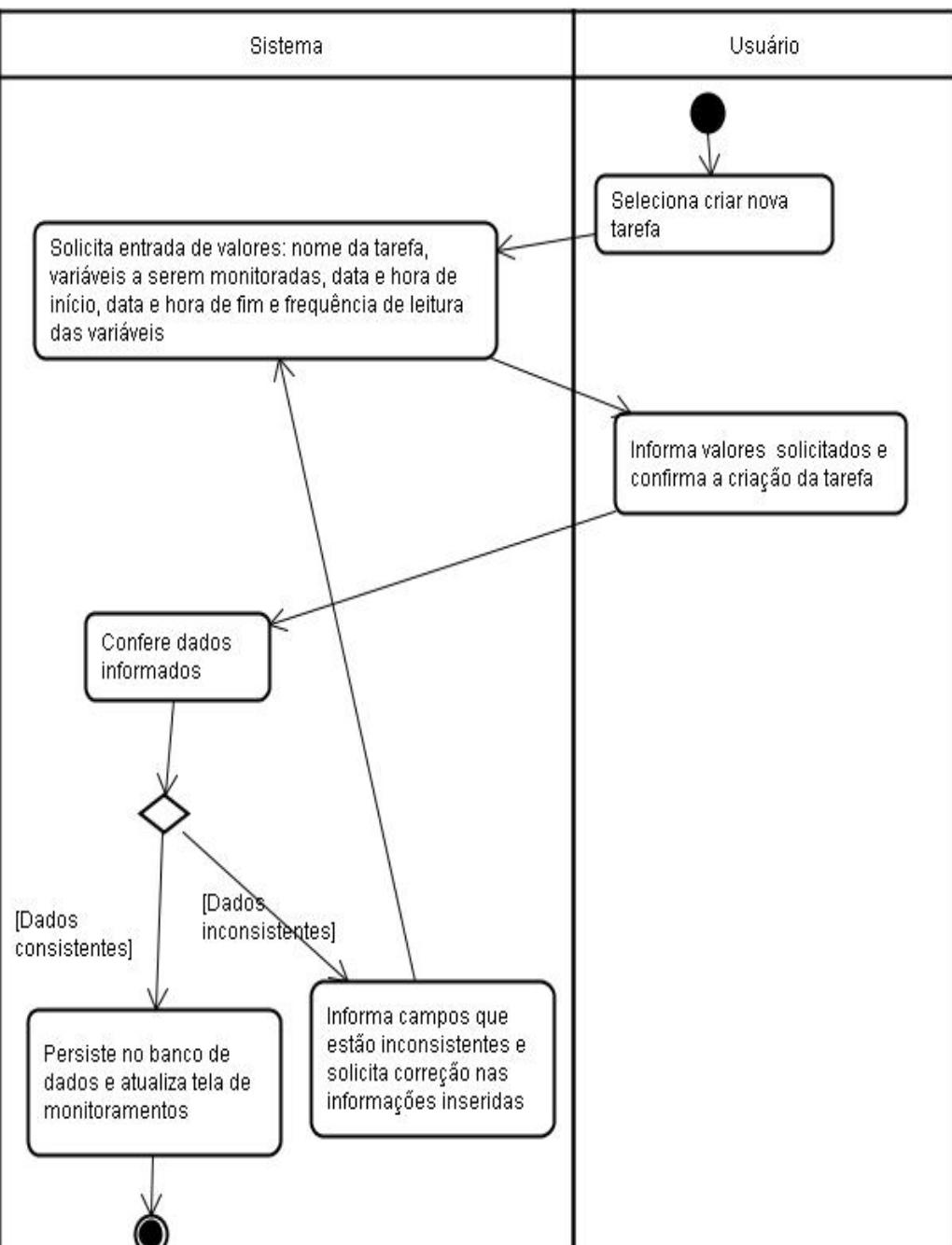
actDiagrama de atividade - Cadastrar variáveis monitoradas



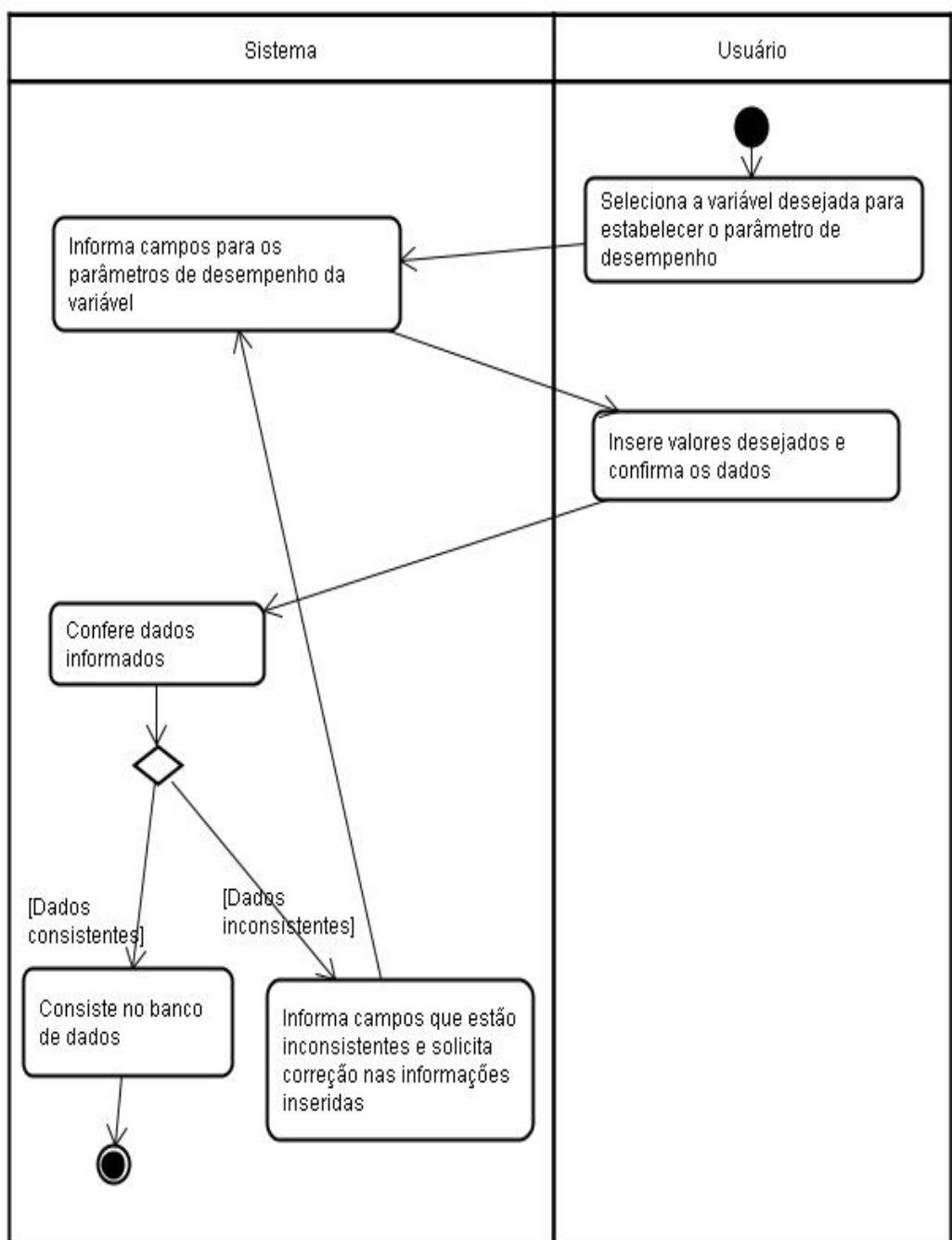
actDiagrama de atividade - Consultar dados



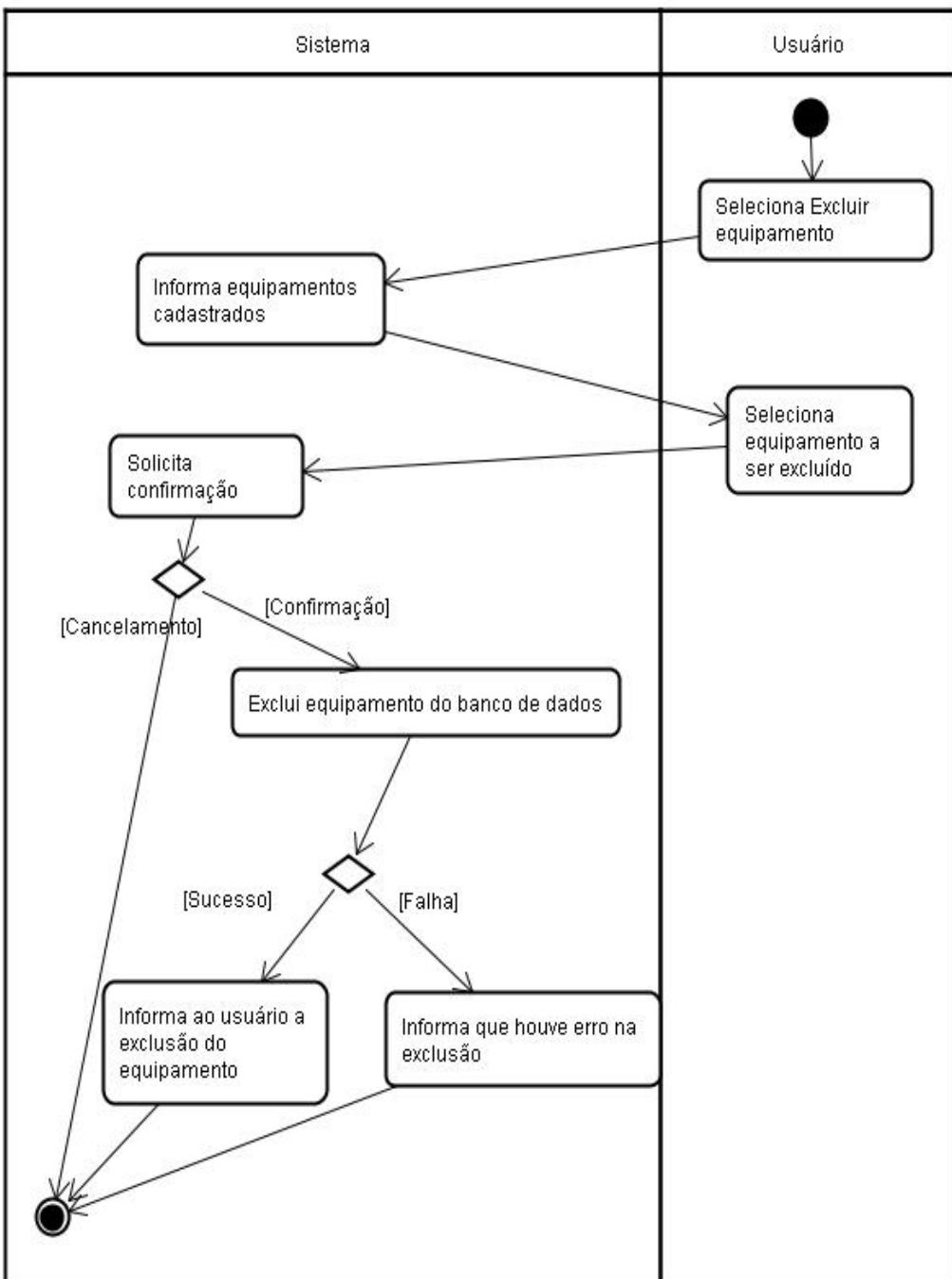
actDiagrama de atividade - Criar tarefas



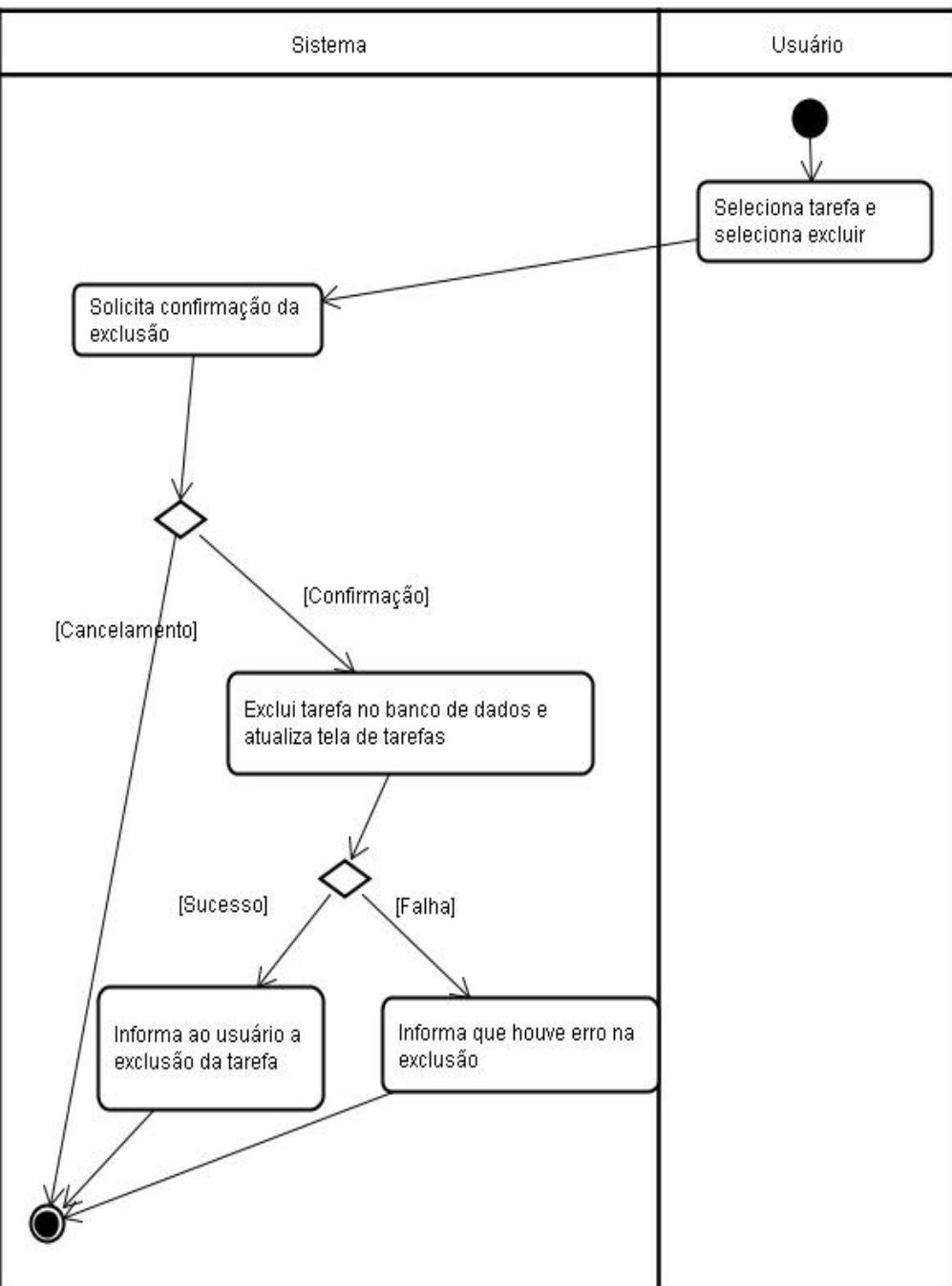
**act Diagrama de atividade - Estabelecer parâmetros de desempenho para variáveis monitoradas**



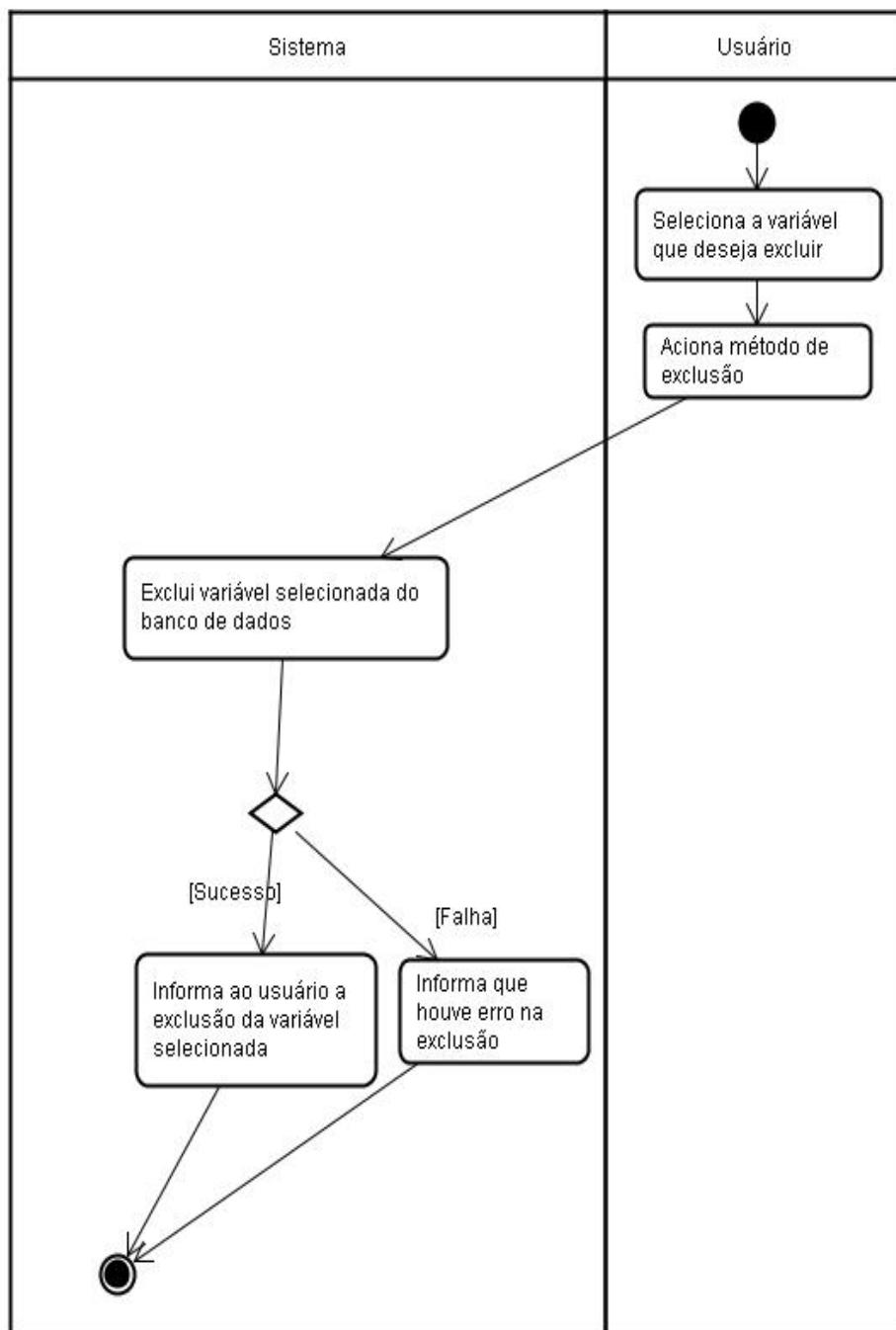
actDiagrama de atividade - Excluir Equipamentos



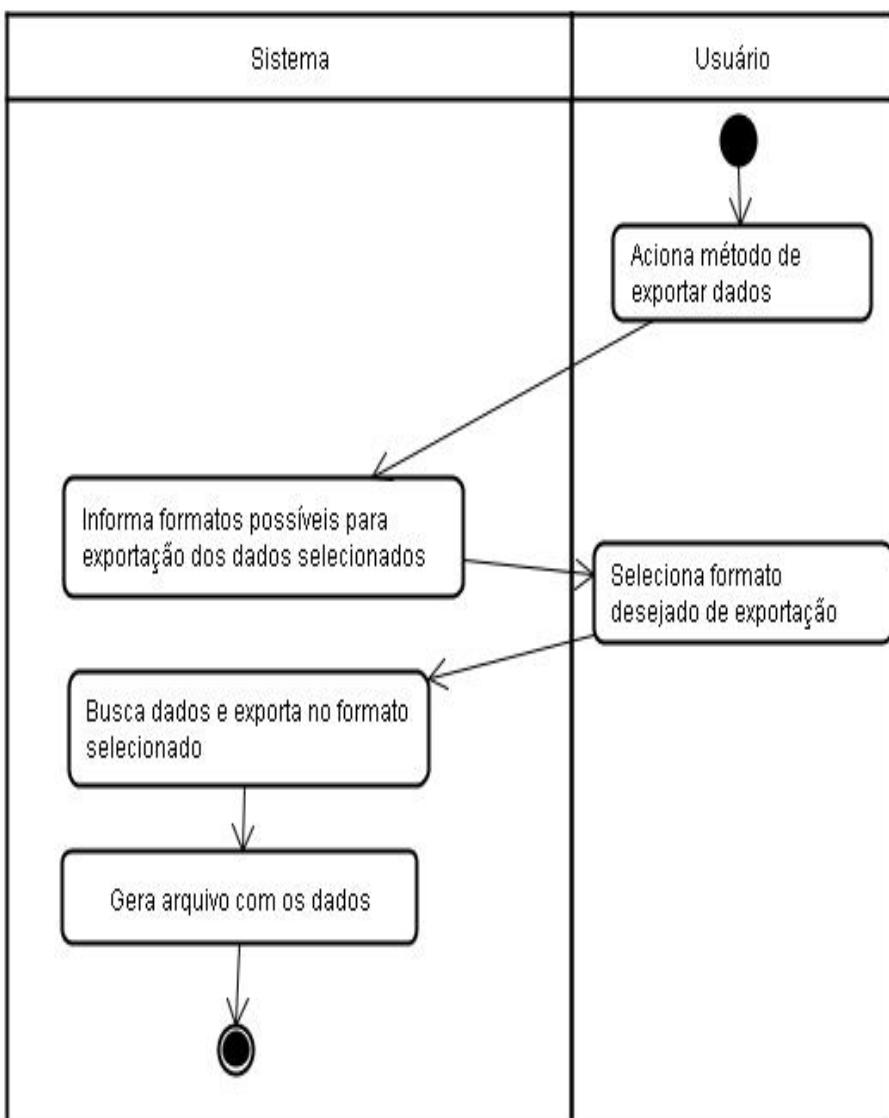
actDiagrama de atividade - Excluir tarefas



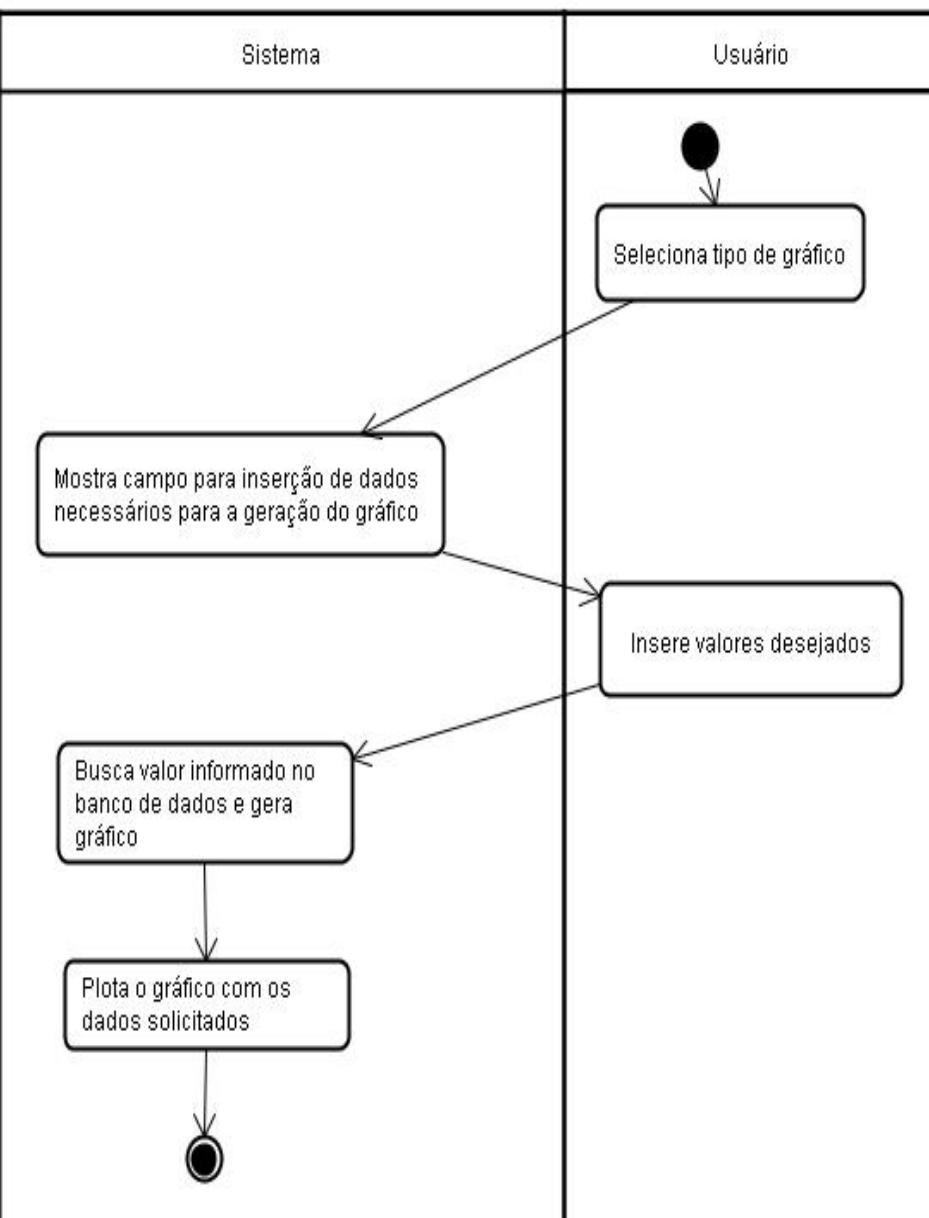
actDiagrama de atividade - Excluir variáveis monitoradas



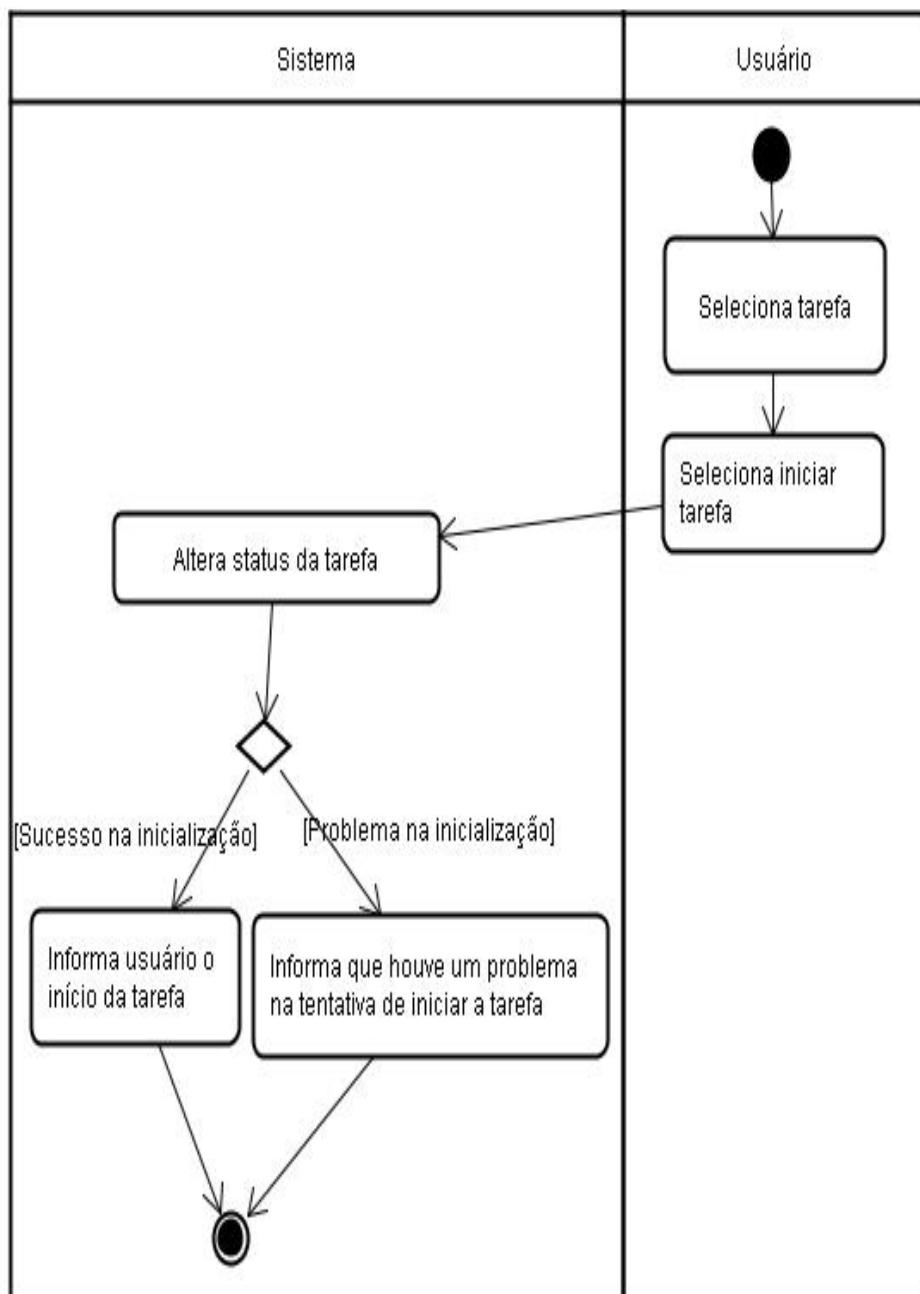
actDiagrama de atividade - Exportar dados estatísticos



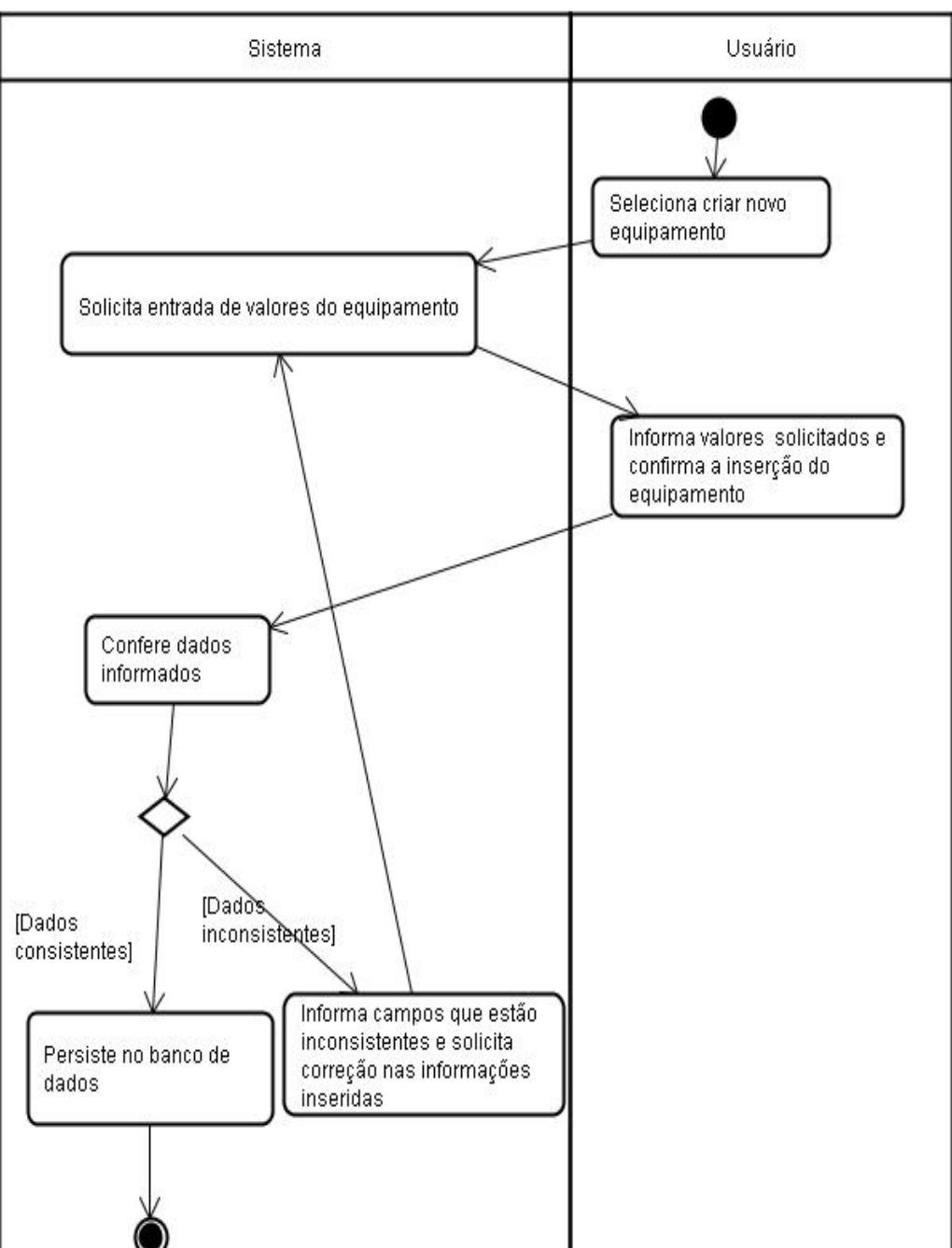
actDiagrama de atividade - Gerar gráficos e visualizações



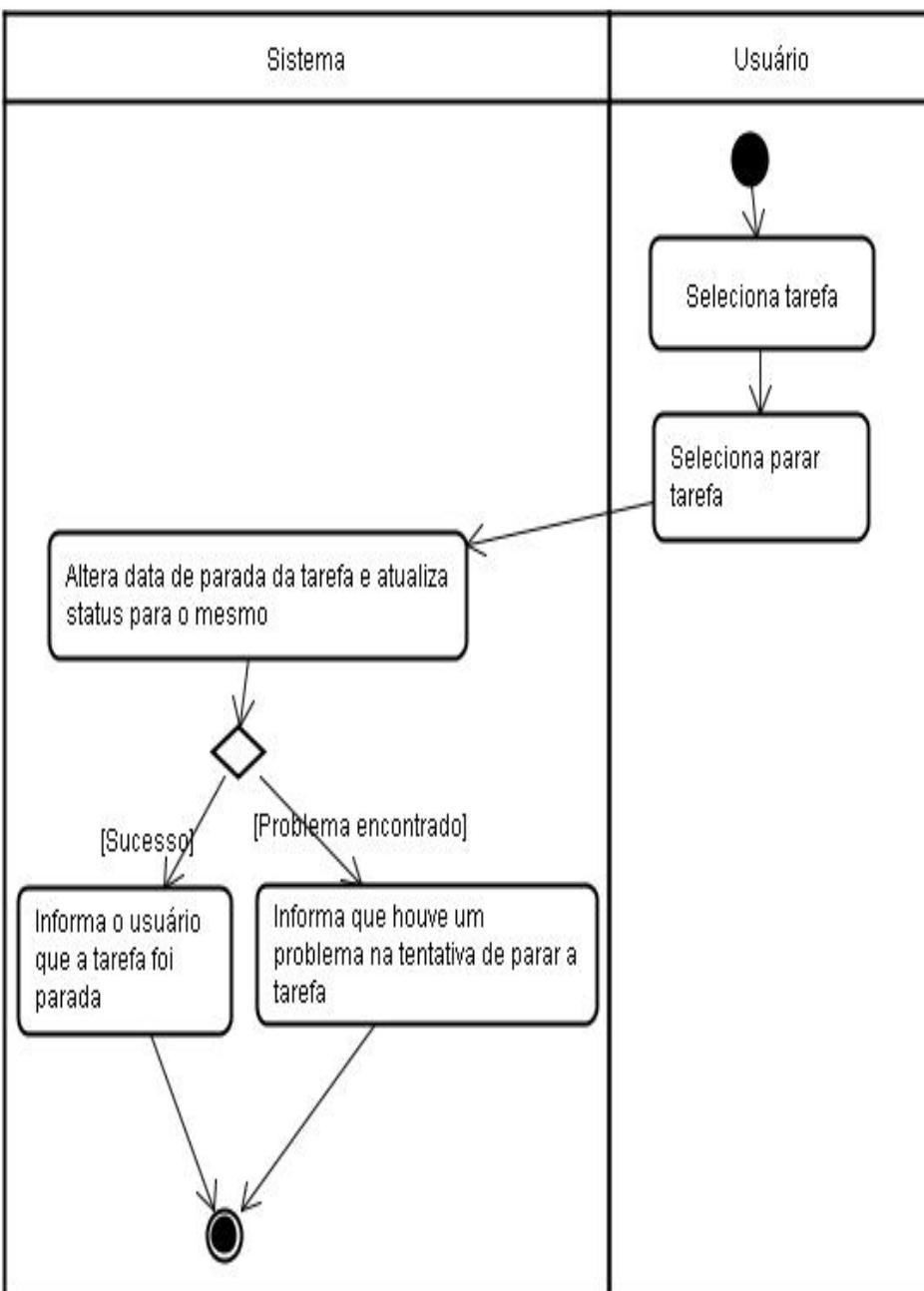
actDiagrama de atividade - Iniciar tarefas



actDiagrama de atividade - Inserir equipamentos



actDiagrama de atividade - Parar tarefas



# Anexo B — Diagramas de Classes

Neste anexo são apresentados os diagramas de classes UML modelados para a implementação do sistema. São apresentados os diagrama de classes, inicialmente uma figura contendo todos os pacotes e classes (Figura 5.1). A seguir cada um dos pacotes é melhor detalhado nas demais figuras.

O pacote *model* contém as classes que são as entidades do programa, os DAOs de acesso às entidades e enumerações utilizadas nas classes entidades (Figura 5.2).

O pacote *controller* contém as classes de controle de acesso às classes DAOs e classes de acesso ao protocolo SNMP (Figura 5.3).

O pacote *util* contém as classes de utilidades do sistema, como configuração de acesso ao banco de dados, exportação de dados, mensagens de erros do sistema, escalonador de leituras e classe inicial do sistema (Figura 5.4).

Os pacotes *graphics* e *charts* contém as classes que geram as visualizações do sistema (Figura 5.5 e Figura 5.6).

E, o pacote *view* contém as classes que representam as janelas e interfaces gráficas do sistema (Figura 5.7).

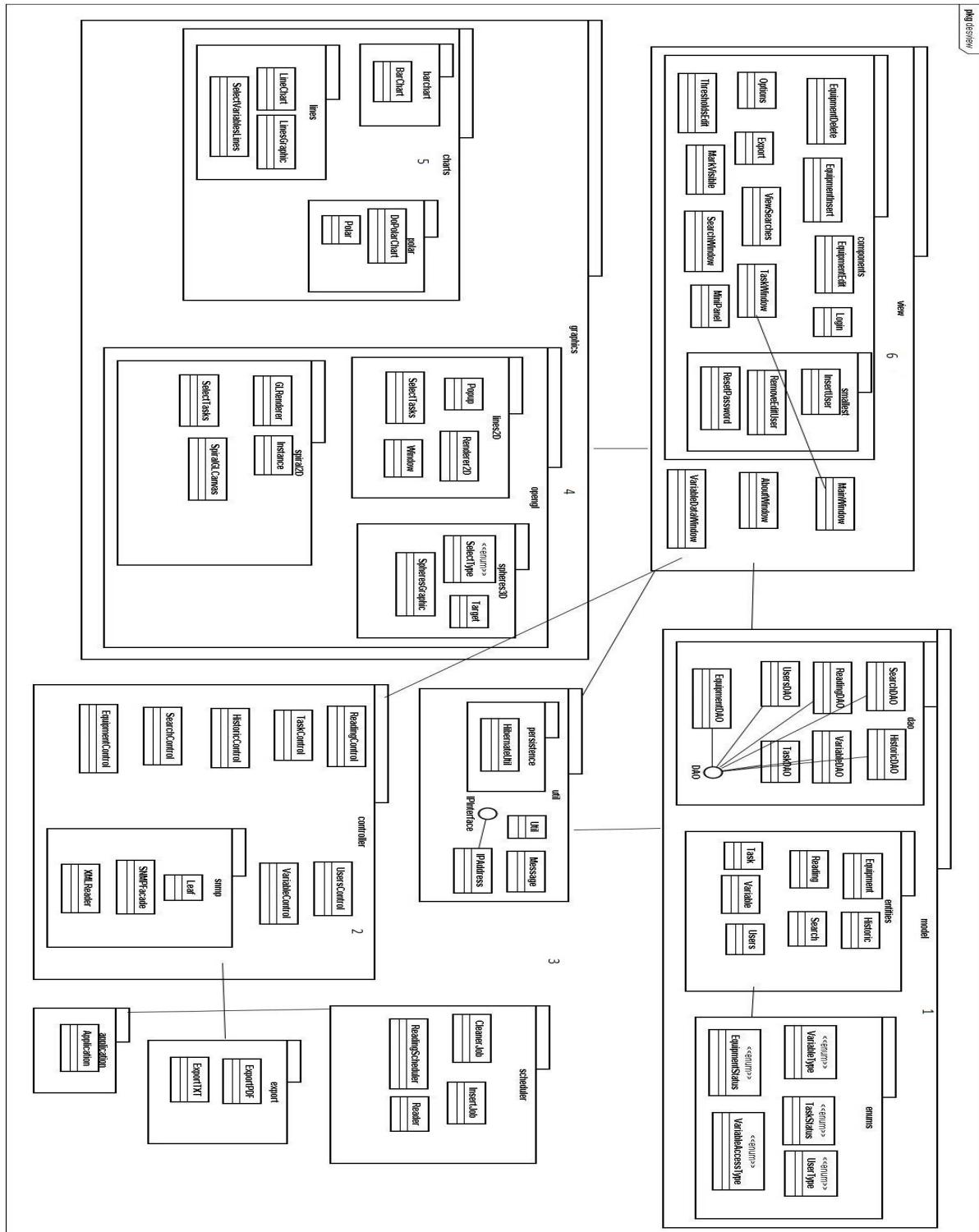


Figura 5.1: Modelo conceitual de classes do sistema

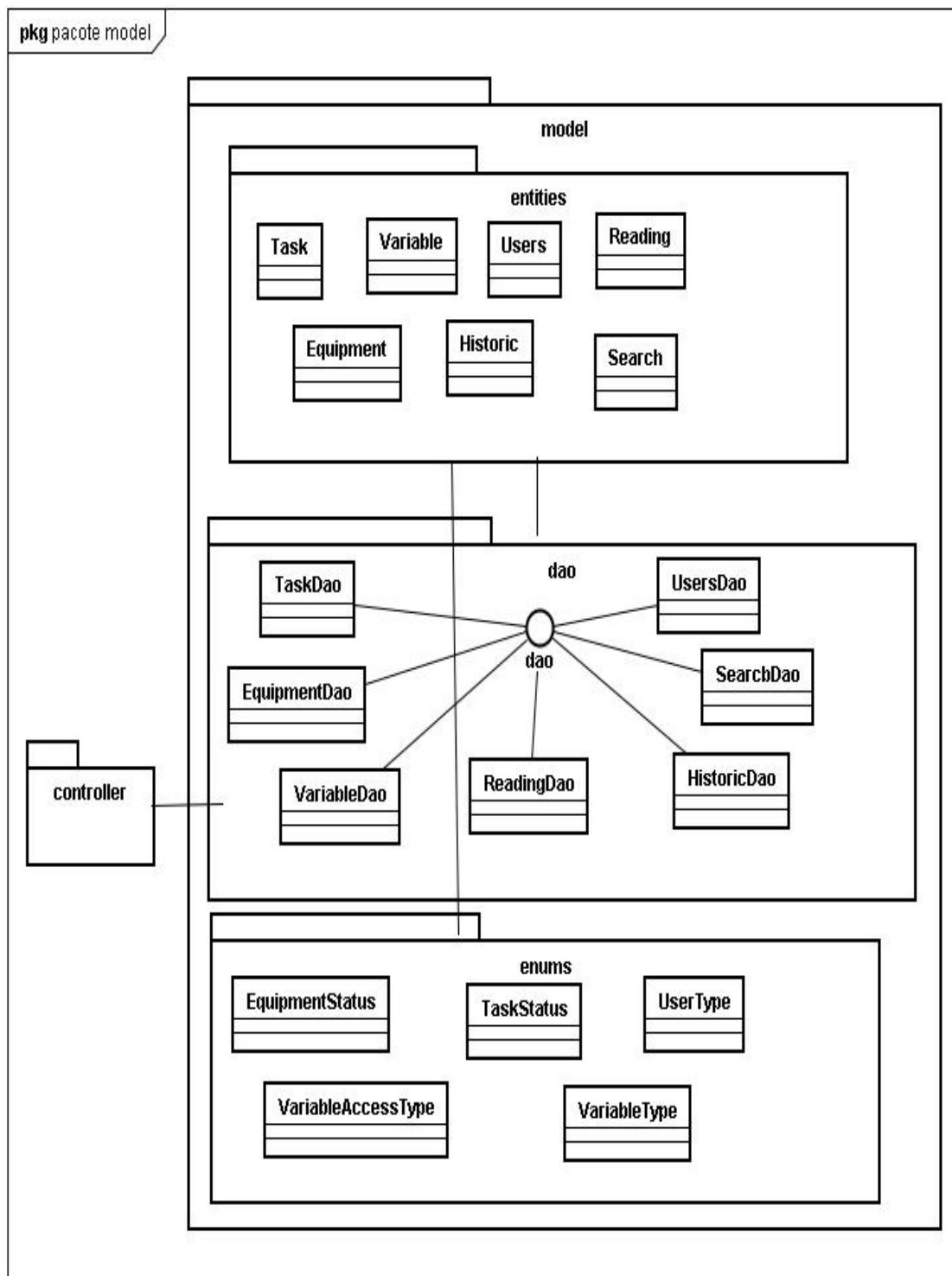


Figura 5.2: Modelo conceitual de classes: parte 1 - pacote *model*.

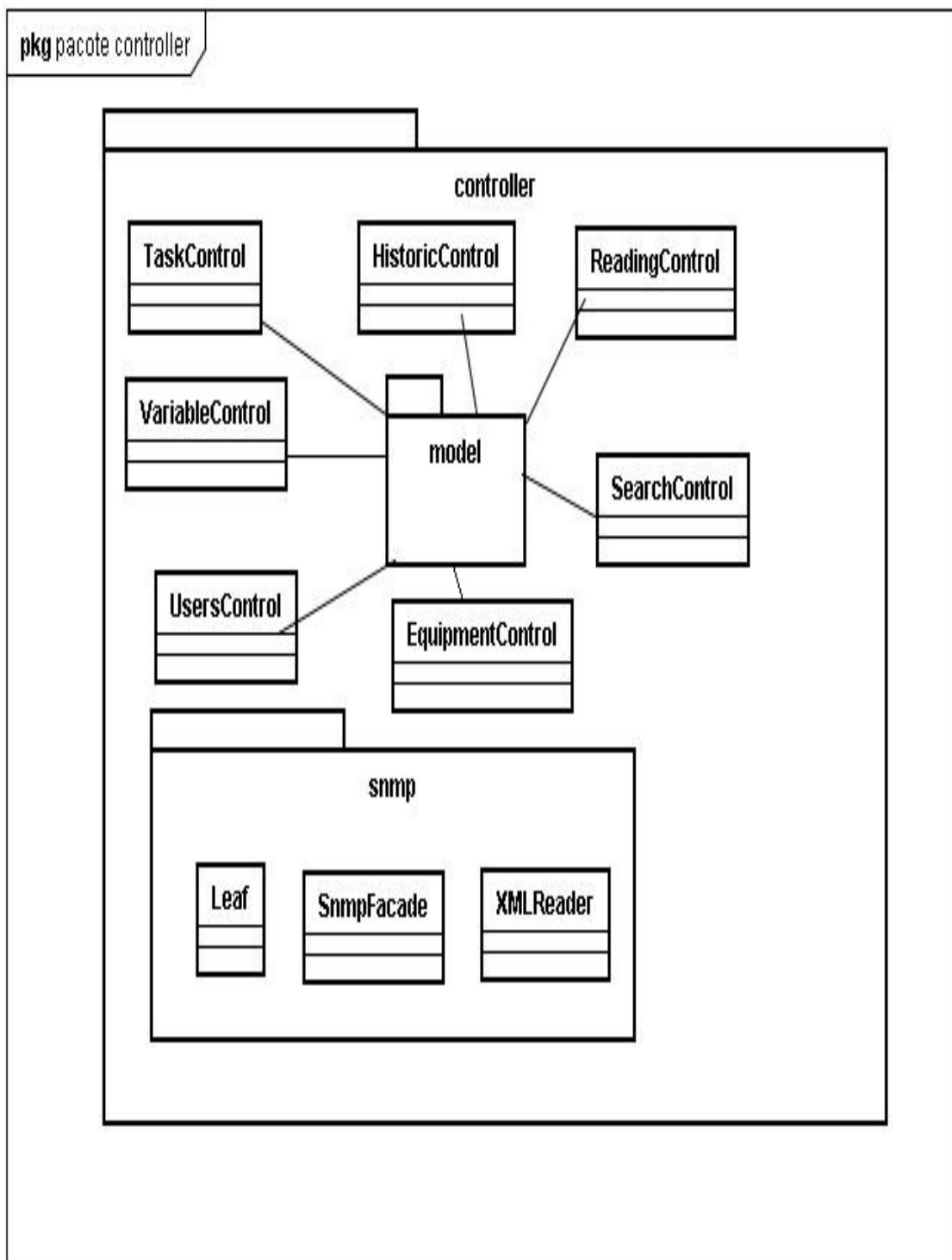


Figura 5.3: Modelo conceitual de classes: parte 2 - pacote *controller*.

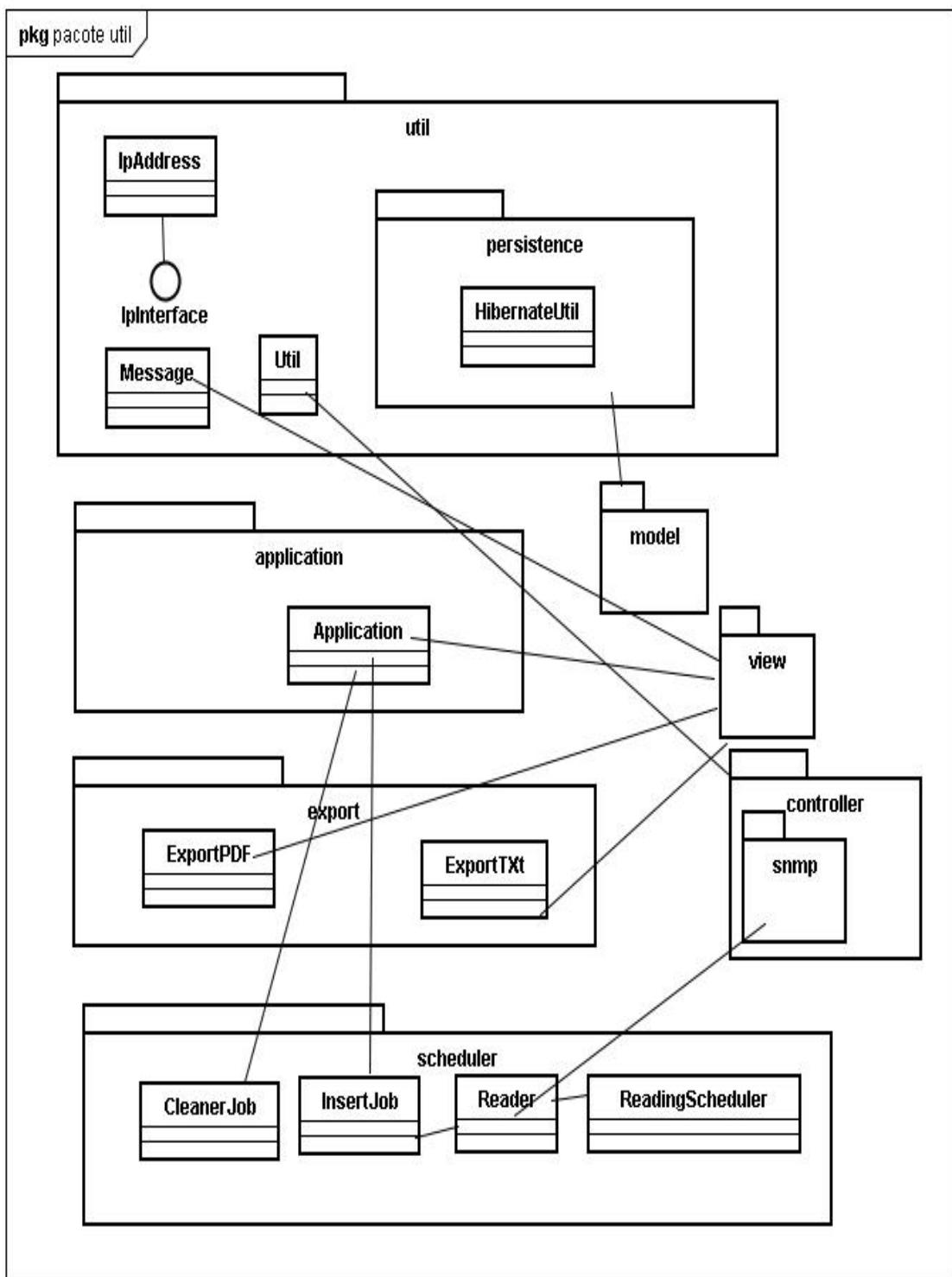


Figura 5.4: Modelo conceitual de classes: parte 3 - pacote *util*.

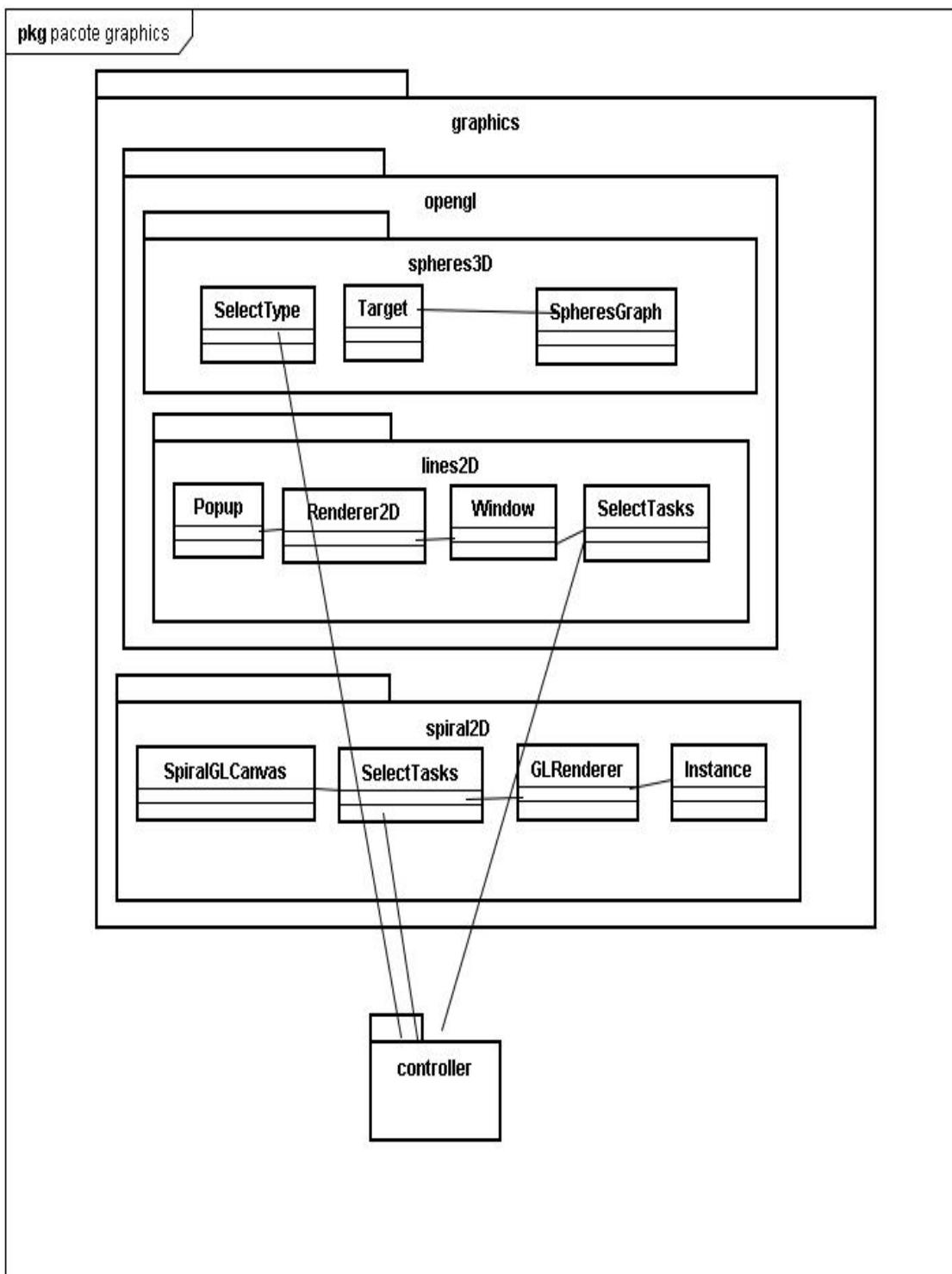


Figura 5.5: Modelo conceitual de classes: parte 4 - pacote *graphics*.

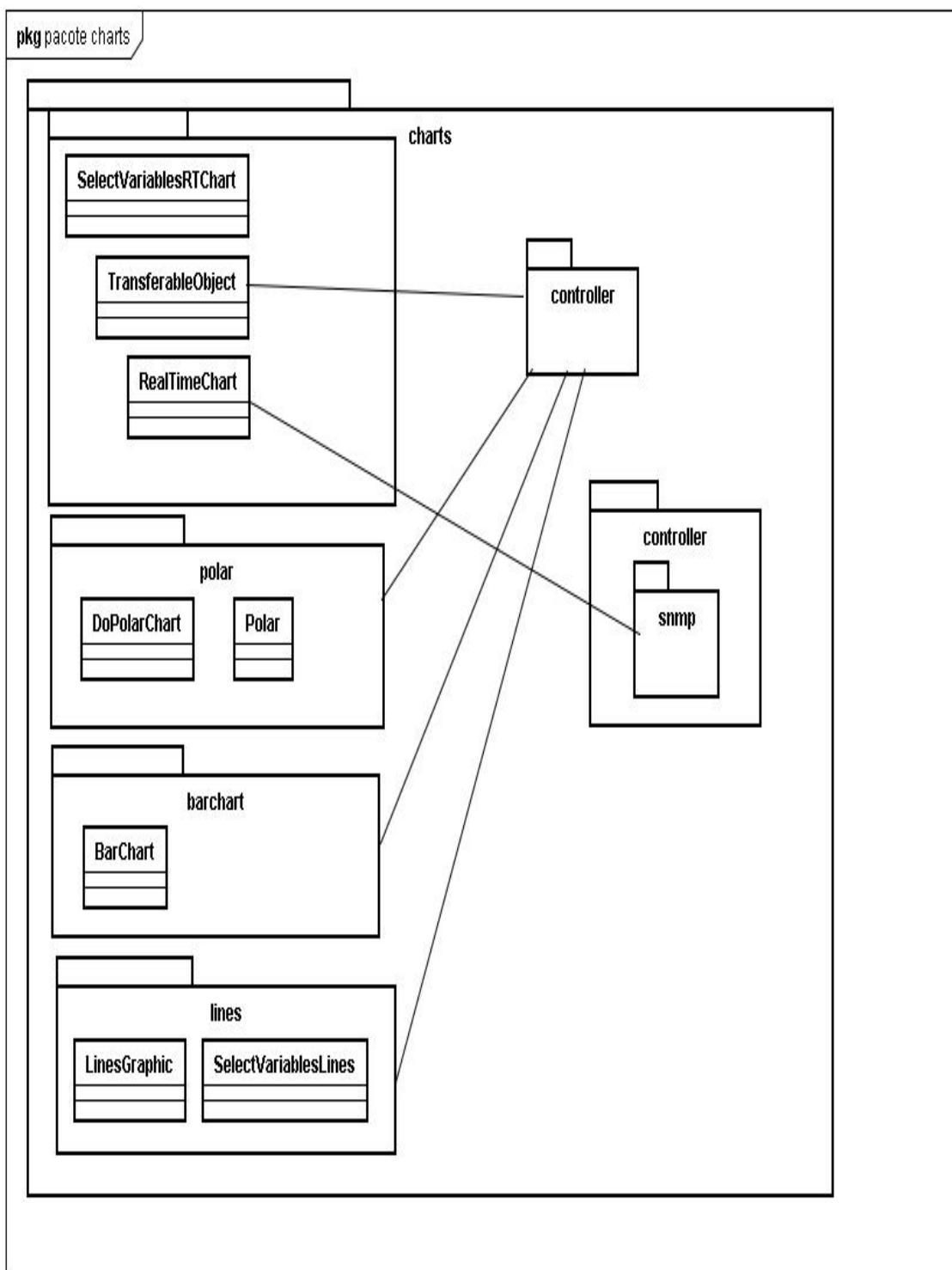


Figura 5.6: Modelo conceitual de classes: parte 5 - pacote *charts*.

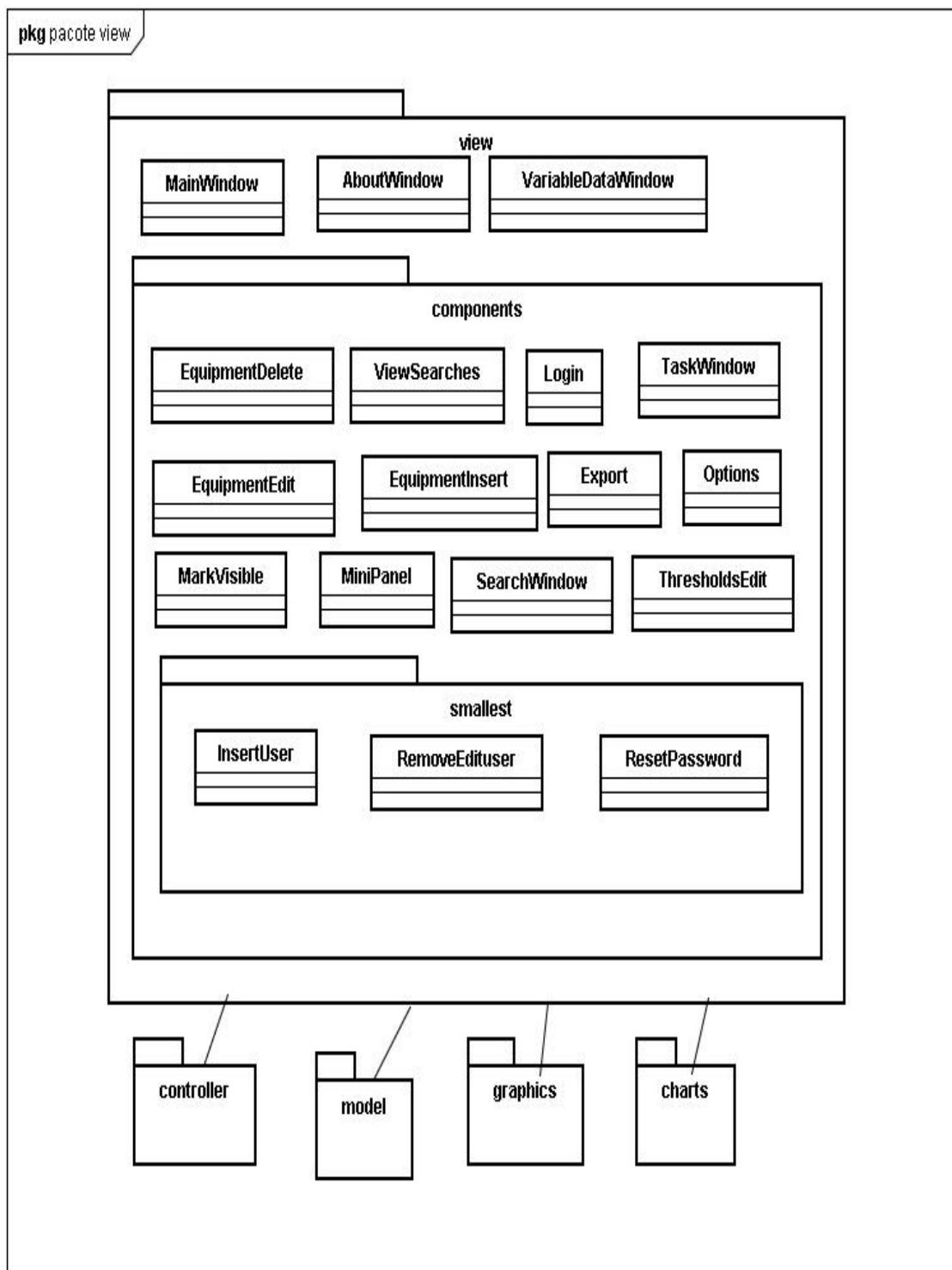


Figura 5.7: Modelo conceitual de classes: parte 6 - pacote *view*.

# Anexo C — Manual de utilização

Neste anexo é apresentado o manual de utilização do sistema Desview. É mostrado como inserir equipamentos, tarefas e variáveis no protótipo e como utilizar as visualizações implementadas.

## *Login*

Para utilizar o sistema é necessário possuir uma conta com usuário e senha. Assim, a tela de *login* é mostrada no início da aplicação, como ilustra a Figura 5.8. Caso não possuir um usuário, crie um, e se esquecer da senha utilize o *link* de recuperação de senha.



Figura 5.8: Tela de *login* do sistema.

## Inserção de equipamentos, de tarefas e de variáveis

Após entrar no sistema é aberta a tela inicial do sistema, conforme ilustra a Figura 5.9. Para iniciar as monitorações é necessário inserir equipamentos e a seguir inserir tarefas, com variáveis de MIB a serem monitoradas.

Para inserir equipamentos e tarefas, utilize o menu *Tools*, a seguir selecione *Insert* e *Equipment* ou *Task* (Figura 5.10). Também é possível inserir tarefas através do botão *New* da janela principal.

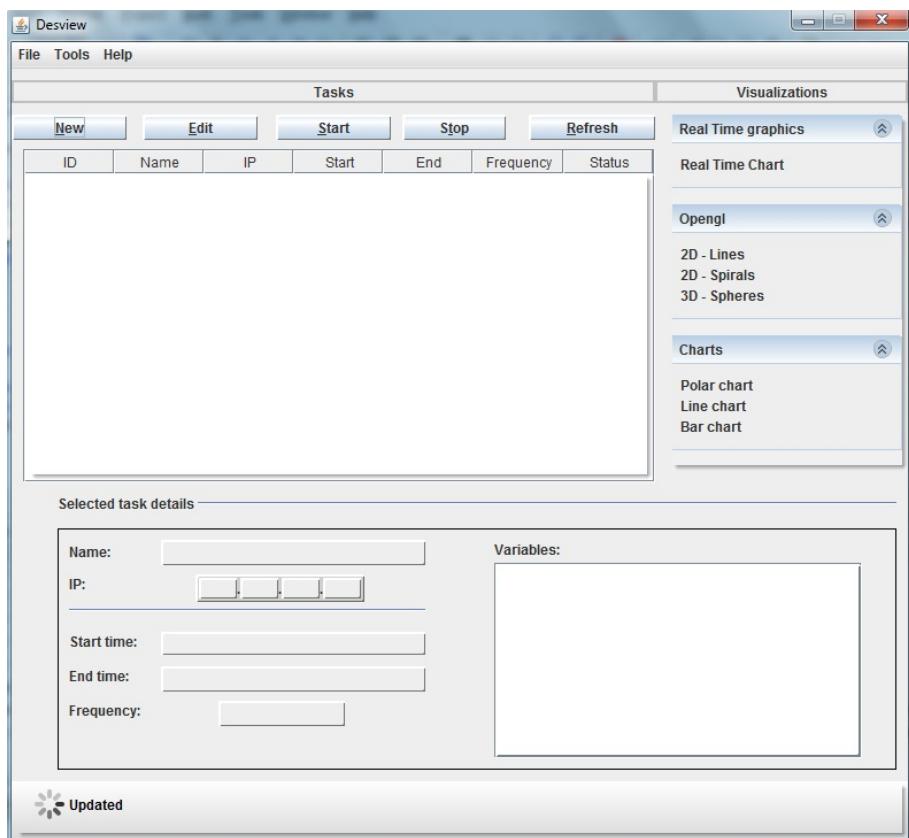


Figura 5.9: Tela inicial do sistema com tarefas e visualizações.

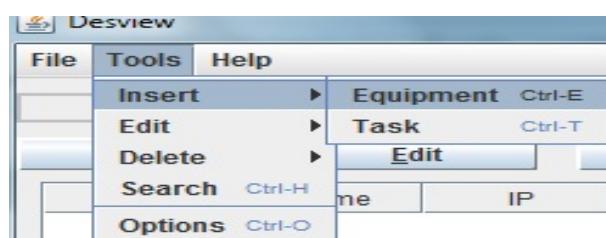


Figura 5.10: Inserindo equipamentos e tarefas.



Figura 5.11: Inserção de equipamentos.

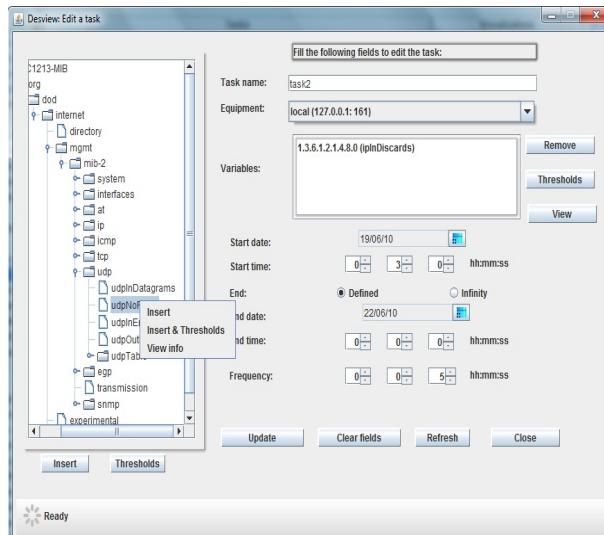


Figura 5.12: Inserção de tarefas e variáveis.

Ao inserir equipamentos, é necessário informar o IP do equipamento, o nome do equipamento, comunidade de leitura e de escrita, *timeout*, número de *retries* e porta de acesso SNMP, como apresentado na Figura 5.11.

Para inserir tarefas, os dados necessários são os seguintes: nome da tarefa, equipamento em que será efetuada a leitura das variáveis, variáveis de MIB a serem monitoradas, data de início e fim e intervalo entre as leituras, como apresentado na Figura 5.12. Cada variável a ser inserida pode possuir *thresholds* superior e inferior ao serem monitoradas. Para inserir os *thresholds*, selecione um elemento da árvore apresentada na Figura 5.12, à esquerda e selecione *Insert & Thresholds*. Será apresentada a janela de edição dos valores ilustrada na Figura 5.13, na qual deve ser inseridos e salvos os valores desejados. A variável é então adicionada à tarefa e passa a ser monitorada juntamente com as demais variáveis que venham a ser adicionadas à tarefa.

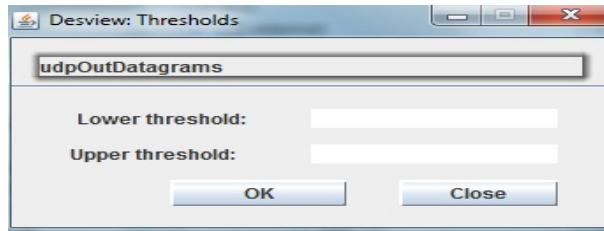


Figura 5.13: Definição de *thresholds* de variáveis.

## Edição de equipamentos, de tarefas e de variáveis

Para editar um equipamento, uma tarefa ou dados de uma variável associada a uma tarefa deve-se utilizar o menu *Tools*, *Edit* e a seguir *Equipment* ou *Task*, como representado na Figura 5.14. Nas janelas de edição dos dados já cadastrados que serão apresentadas, deve-se inserir e salvar as alterações desejadas.



Figura 5.14: Edição de equipamentos e tarefas.

## Remoção de equipamentos, tarefas e variáveis

Para remover uma tarefa selecione a tarefa desejada na tabela de tarefas em monitoração da janela principal (Figura 5.9), utilize o menu *Tools*, *Delete* e a seguir *Task*, como representado na Figura 5.15. Para remover um equipamento inicialmente selecione *Equipment*, após será apresentada uma janela como a mostrada na Figura 5.16, escolha o equipamento desejado e remova-o. Também é possível tornar uma tarefa invisível, ou seja, apenas mantendo-a no banco de dados, porém sem aparecer na janela de tarefas e sem efetuar leituras. Assim, os dados estatísticos da mesma não são removidos do banco de dados. Para remover variáveis abra a tarefa desejada e remova utilizando o botão *Remove*, mostrado na Figura 5.12.

## Parar e iniciar tarefas

As tarefas cadastradas que estejam executando podem ser paradas caso desejado. Também é possível iniciar uma tarefa que esteja parada. Essas ações podem ser executadas

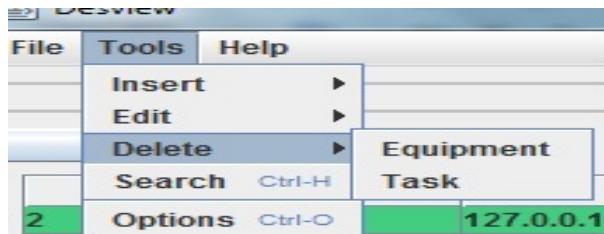


Figura 5.15: Remover de equipamentos e tarefas.

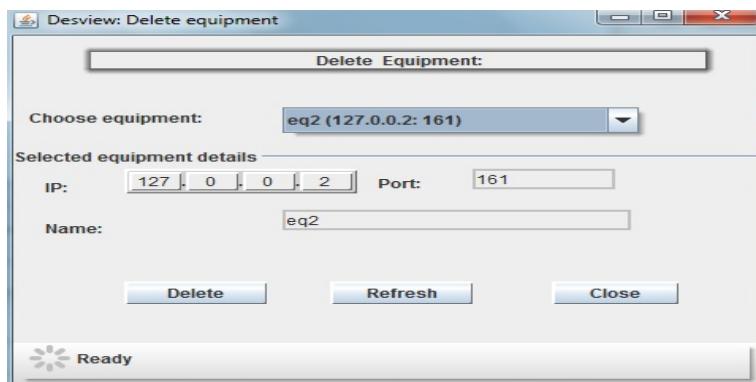


Figura 5.16: Janela de escolha de equipamento a ser deletado.

através dos botões *Start* e *Stop*, mostrados na Figura 5.17. Tarefas que possuem o estado *Stopped* não são lidas pelo escalonador, ou seja, não realizam consultas SNMP nem inserem o resultado das consultas no banco de dados. Também é possível atualizar a tela com as tarefas e verificar o estado de cada uma das tarefas, criar uma nova tarefa e editar uma tarefa já cadastrada.



Figura 5.17: Botões de ações para as tarefas.

## Exportar dados

O protótipo permite exportar dados nos formatos texto (TXT) e PDF. Para exportar os dados lidos de leituras, deve-se utilizar o menu *File*, a seguir *Export*. Escolha o formato de exportação como mostrado na Figura 5.18. A seguir, informe o diretório a ser salvo o arquivo.

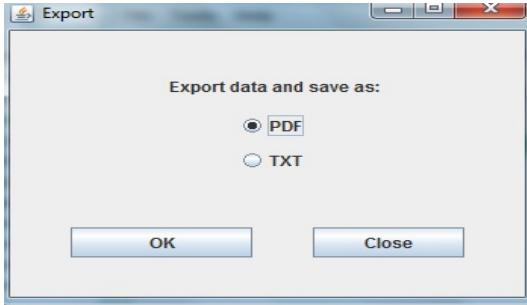


Figura 5.18: Tela de escolha de formatos de exportação.

## Utilizando as visualizações

São oferecidas as seguintes visualizações no sistema (Figura 5.19):

- Visualização em linhas;
- Visualização em espirais;
- Visualização em esferas;
- *Chart* de monitoração em tempo real de variáveis;
- *Chart* polar, de barras e de linhas.

### Visualizações em *charts*

O *chart* de tempo real tem por objetivo a monitoração de até 5 variáveis de diferentes tarefas e assim, verificar o valor atual de cada uma variáveis e monitorá-las a partir de um *threshold*, informando se algum dos valores lidos das variáveis passou do *threshold* informado.

Os demais *charts* são utilizados para verificar estatísticas das leituras efetuadas. O protótipo possui *charts* em formato de barras, de linhas e gráfico polar. Para utilizar informe os dados solicitados para construção do *chart*, como: ano, mês e variável para que o gráfico seja plotado.

## Visualizações em OpenGL

### Gráfico 2D de linhas e de espirais

Para gerar os gráficos 2D de linhas ou de espirais, inicialmente é necessário escolher qual é a tarefa que se deseja visualizar, como mostrado na Figura 5.22, após são gerados os gráficos mostrando se os valores lidos estão dentro, acima ou abaixo do intervalo determinado como normal para a variável, como mostrado na Figura 5.23, que apresenta os dois gráficos monitorando diferentes variáveis.

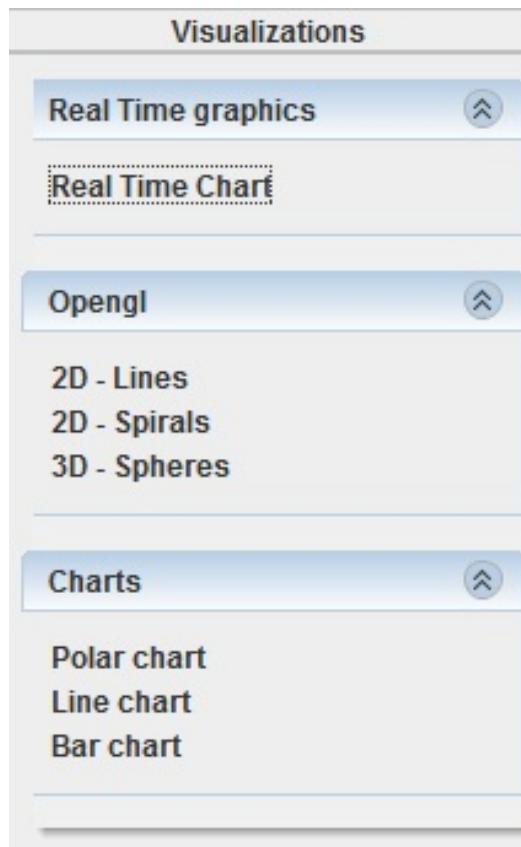


Figura 5.19: Tela de escolha de visualizações a serem plotadas.

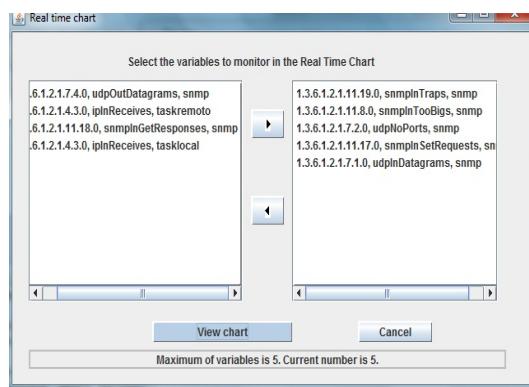


Figura 5.20: Janela de escolha de variáveis.

## Gráfico 3D

Para gerar os gráficos 3D de esferas, inicialmente é necessário escolher qual é a tarefa que se deseja visualizar, como mostrado na Figura 5.22. Após são plotadas sequências de gráficos 3D informando as médias das variáveis nos meses do ano e o estado da média em

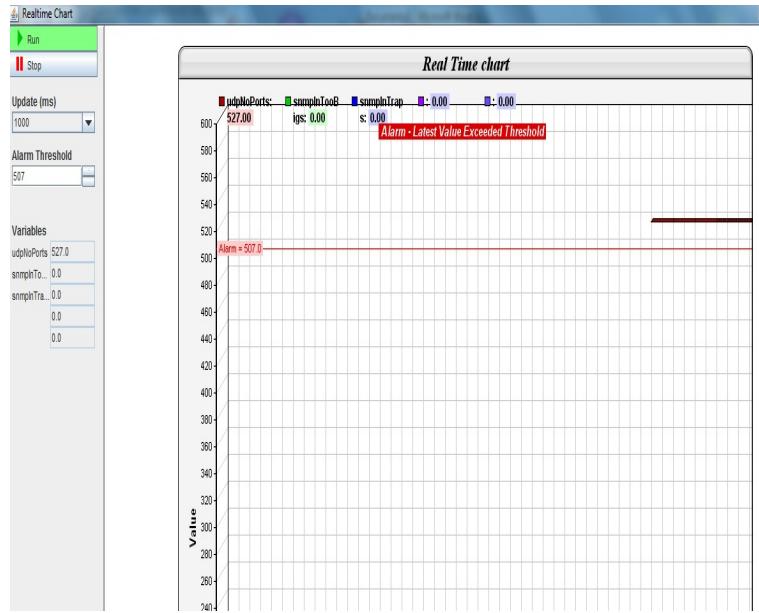


Figura 5.21: Exemplo de gráfico de tempo real gerado.

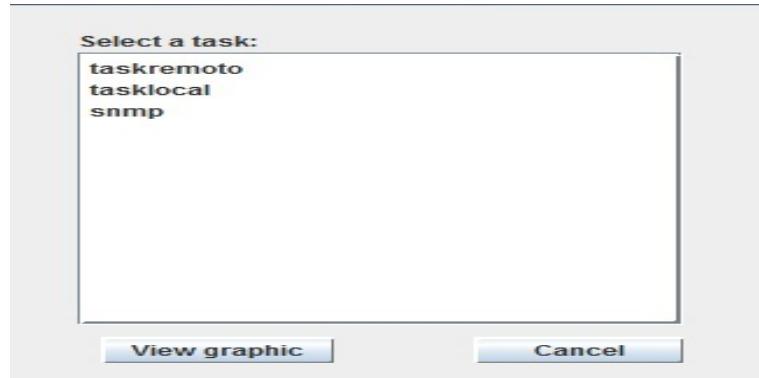


Figura 5.22: Janela de seleção de tarefas para gráficos OpenGL.

relação aos *thresholds*, ao clicar em uma das variáveis é trazido a média dos dias do ano Figura 5.26, novamente informando a média em relação aos *thresholds*. Dois exemplos de imagens geradas pela visualização em esferas são mostrado na Figura 5.24 e Figura 5.25.

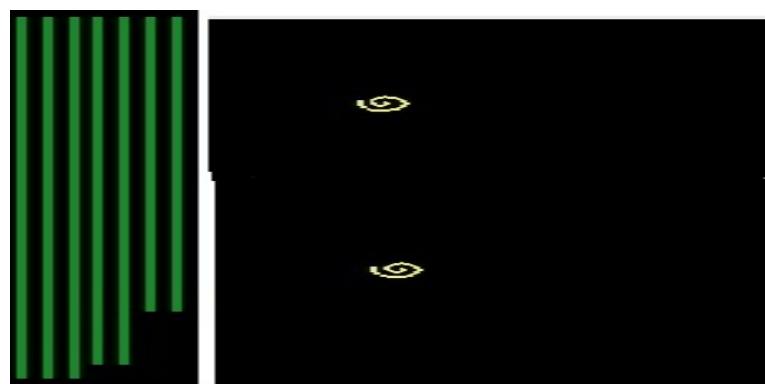


Figura 5.23: Exemplo de gráfico de linhas e de espirais gerados.

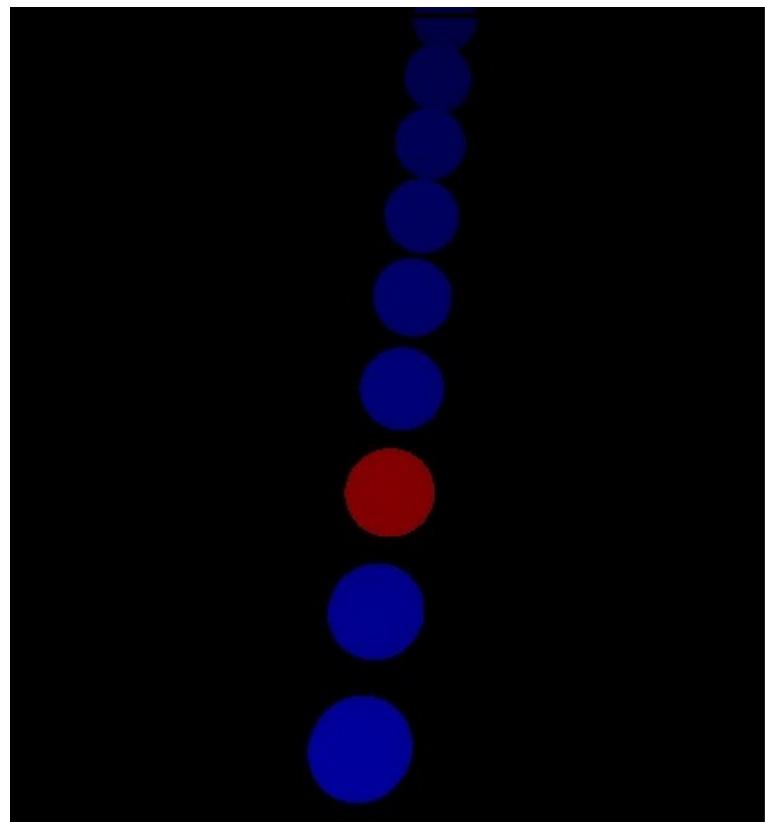


Figura 5.24: Exemplo 1 de gráfico em esferas.

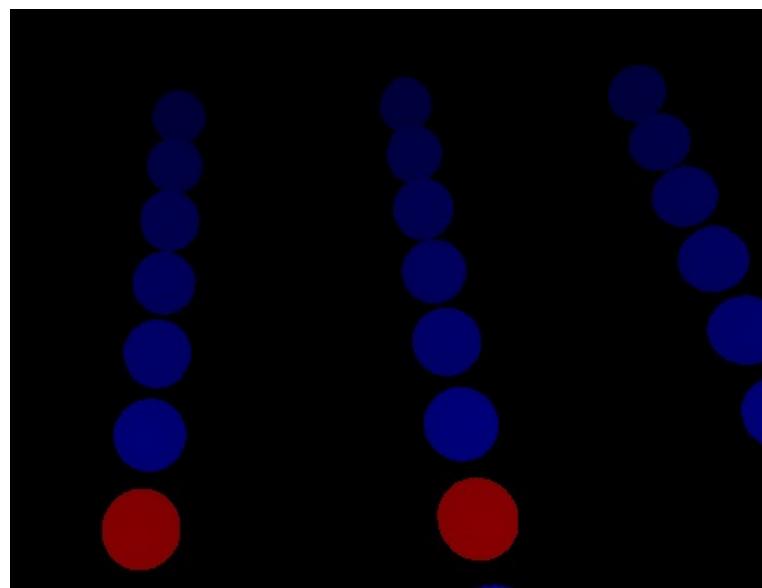


Figura 5.25: Exemplo 2 de gráfico em esferas.

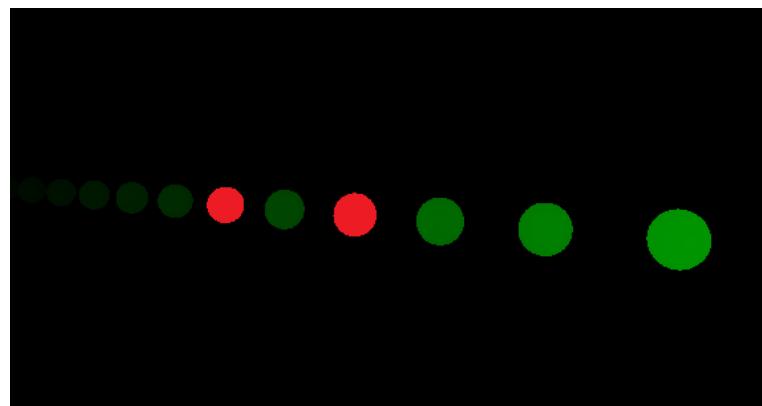


Figura 5.26: Nível de dias de uma tarefa selecionada.