FACULDADE DE TECNOLOGIA DE AMERICANA

Curso de Analise e Desenvolvimento de Sistemas

Pedro Mariano Sampaio

Uma Aplicação para o auxílio à detecção de falhas em redes de computadores.

Americana, S. P.

2016

FACULDADE DE TECNOLOGIA DE AMERICANA

Curso de Analise e Desenvolvimento de Sistemas

Pedro Mariano Sampaio

Uma Aplicação para o auxílio à detecção de falhas em redes de computadores.

Trabalho Monográfico, desenvolvido em cumprimento à exigência curricular do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas da Fatec-Americana, sob orientação do Prof. MSc. Rossano Pablo Pinto

Área: Desenvolvimento de Sistemas

Americana, S. P.

2016

FICHA CATALOGRÁFICA – Biblioteca Fatec Americana - CEETEPS
Dados Internacionais de Catalogação-na-fonte

S185u

Sampaio, Pedro Mariano
Uma aplicação para o auxílio à detecção de falhas em redes de computadores. / Pedro Mariano Sampaio. – Americana: 2016.
57f.

Monografia (Graduação em Tecnologia em Análise e Desenvolvimento de Sistemas). - - Faculdade de Tecnologia de Americana – Centro Estadual de Educação Tecnológica Paula Souza.
Orientador: Prof. Me. Rossano Pablo Pinto

1. Redes de computadores I. Pinto, Rossano Pablo II. Centro Estadual de Educação Tecnológica Paula Souza – Faculdade de Tecnologia de Americana.

CDU: 681.519

Pedro Mariano Sampaio

Uma aplicação para o auxílio à detecção de falhas em redes de computadores

Trabalho de graduação apresentado como exigência parcial para obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas, pelo CEETEPS/Faculdade de Tecnologia – Fatec/ Americana.
Área de concentração: Desenvolvimento

Americana, 23 de junho de 2016.

Banca Examinadora:

Rossano Pablo Pinto (Presidente)
Mestre
Fatec Americana

Clerivaldo José Roccia (Membro)
Mestre
Fatec Americana

Wladimir da Costa (Membro)
Mestre
Fatec Americana

AGRADECIMENTOS

Em primeiro lugar agradeço primeiramente à minha família por todo o suporte durante o decorrer do meu curso. Agradeço também aos meus amigos, à Fatec e ao corpo docente.

DEDICATÓRIA

Dedico este trabalho à Deus, aos meus pais, pelo apoio e paciência, aos meus amigos que aqui fiz e que eternamente estarão em minhas lembranças.

RESUMO

Este trabalho apresenta o desenvolvimento de uma aplicação para detecção de falhas em redes de computadores. Seu principal objetivo é expor a importância de um software que assista programadores a determinar problemas em ambientes de redes, os quais podem ser a causa do mal funcionamento de aplicações e/ou sistemas. Considera-se necessário o desenvolvimento deste software com uma interface simples e intuitiva devido ao baixo conhecimento dos programadores sobre a área de redes.

Palavras Chave: SNMP; Sistema de Monitoração, Preparação de dados.

ABSTRACT

This project addresses the development of an application for the detection of problems in computer network environments. It's main purpose is to expose the importance of a software that assist developers to determine problems in network environments, which can be the cause of malfunctions of applications or systems. It's considered necessary the development of this software with a simple and intuitive interface due to the lack of knowledge of software developers in the area of computer network.

Keywords: SNMP, Monitoring System, Data Preparation.

SUMÁRIO

LISTA DE FIGURAS

INTRODUÇÃO

Durante a década de 60 havia uma preocupação sobre a importância do *software* nas atividades da sociedade, assim como sobre a necessidade de técnicas baseadas em fundações teóricas para o desenvolvimento de *software*, da mesma forma que as outras áreas da engenharia. Estas preocupações levaram à criação da primeira Conferência de engenharia de *software* em 1968. Nesta foram determinadas diversas boas práticas para o desenvolvimento de *software*. [Mens, Demeyer, 2008]

A partir deste momento os *software*s começaram a se desenvolver exponencialmente, gerando cada vez mais dados, um maior número de usuários, suportando diversos tipos de dispositivos e, principalmente, tendo conexão à rede. Com esse crescimento, aplicações se tornaram cada vez mais comuns em empresas, corporações, governos e diversos outros lugares, necessitando assim de manutenções ocasionais. [Mens, Demeyer, 2008]

Estas manutenções muitas vezes não são necessárias devido a um problema com o próprio *software*, mas sim devido a um problema no ambiente de redes.

No campo do desenvolvimento de software não há um grande conhecimento da área de redes de computadores, o que torna difícil identificar a fonte do mal funcionamento.

Torna-se clara então a necessidade de uma aplicação que dê assistência a desenvolvedores ou suporte na determinação de problemas no ambiente de redes. Ou seja, uma aplicação que não requeira que o usuário disponha de um conhecimento da área, que informe de forma rápida e clara as principais informações do ambiente, e que determine automaticamente a necessidade ou não de uma manutenção do mesmo.

Consequentemente o **Problema** encontrado aqui é o baixo conhecimento da área de redes por parte dos programadores e a falta de uma aplicação que de assistência no momento de determinação de causa de problemas.

A **Solução** encontrada foi o desenvolvimento de uma aplicação que apresente de forma simplificada informações básicas sobre a rede, como:

- Taxa de erros;
- Latência;
- Perda de Pacotes
- Erros Físicos;

Dessa forma torna mais fácil reconhecer falhas ou gargalos da rede, diminuindo o tempo e esforço necessários para determinar a causa de problemas que afetam as aplicações.

O **Objetivo Geral** então é auxiliar desenvolvedores no momento de detecção de falhas em softwares assim como determinar a necessidade ou não de uma manutenção.

Já o **Objetivo Específico** é o desenvolvimento de uma aplicação que ofereça assistência aos programadores a encontrar falhas no ambiente de redes quando trabalhando em problemas que afetem o *software* e assim compreender a importância de um bom gerenciamento de redes para o software.

Tem-se como **Justificativa** o benefício que o desenvolvimento de um *software* que forneça informações básicas sobre a rede, como taxa de erros e latência, trará a um desenvolvedor.

O conhecimento destas informações não só facilita a encontrar problemas afetando a própria aplicação como também problemas afetando usuários que a utilizam.

Com estes dados simples, pode-se facilmente encurtar o tempo necessário para encontrar falhas e além disso diminuir o esforço e custos.

É importante lembrar que este tipo de melhoria no suporte a um *software* não só provê vantagens a curto prazo (como a maior velocidade), mas também vantagens a longo prazo, como a maior confiança e satisfação da parte dos usuários.

O trabalho foi dividido em 5 capítulos, sendo que o primeiro se conceitua na definição de qualidade de serviços e de gerenciamento de redes além de todas as tecnologias e técnicas relacionadas. O segundo traz a explicação do que é um ambiente de redes e de todos os equipamentos que o compõe. O terceiro descreve o processo de desenvolvimento do *software*, assim como suas funcionalidades e aplicações. No quarto há uma aplicação do *software* em um cenário virtual, assim como testes, resultados e suas respectivas soluções.

Com base nos dados obtidos no quarto capitulo, o quinto se atém às conclusões dos testes e considerações finais.

1. REFERENCIAL TEÓRICO

Antes de iniciar o assunto sobre gerenciamento de redes, deve-se lembrar que o mesmo tem todas as suas funções e utilidades direcionadas para um único fim que é o bom funcionamento do ambiente de redes.

Com o bom funcionamento do ambiente de redes, espera-se que o software, tendo todos seus requisitos atendidos, funcione de forma correta.

O mal funcionamento do software traz consigo a necessidade de uma manutenção.

1.1 MANUTENÇÃO DE SOFTWARE

O IEEE 1219 (padrão para manutenção de software) define manutenção de software como:

> "A modificação de um produto de software depois de sua entrega a fim de corrigir falhas, melhorar a performance ou outros atributos, ou para adaptar o produto a um ambiente modificado. " (IEEE, 1998, p. 3-4) [1]

Além disso o IEEE 1219 divide a manutenção de software em 4 tipos diferentes:

- **Manutenção corretiva**: Modificação reativa de um produto de software feita após a entrega para corrigir falhas
- **Manutenção de emergência**: Manutenção corretiva não agendada feita para manter o sistema operacional.
- **Manutenção adaptativa**: modificação de um produto de *software* feita após a entrega para mantê-lo estável em um ambiente que foi ou que está sendo modificado.
- **Manutenção perfectiva**: Modificação de um produto de *software* feita após a entrega para melhorar a performance ou capacidade de manutenção.

Neste trabalho será abordada principalmente a manutenção adaptativa.

---

[1] "Modification of a software product after delivery to correct faults, to improve performance or other attributes, or to adapt the product to a modified environment" (*Tradução nossa*).

A manutenção da aplicação dá-se necessária quando o ambiente sofre uma mudança ou atualização, o que pode causar uma perda de dados ou mal funcionamento, e isso principalmente no caso do ambiente de redes.

Muitas vezes uma mudança no mesmo, programada ou não, não requer a manutenção do *software*, mas sim do próprio ambiente, pois o mesmo deixou de atender aos requisitos do sistema, os quais são garantidos pelo QoS (Qualidade de Serviços).

1.2 QUALIDADE DE SERVIÇOS

Basicamente qualidade de serviços ou QoS é a habilidade da rede de satisfazer as necessidades de um usuário ou aplicação. É também a capacidade de segmentar ou diferenciar tráfegos e tratá-los de forma diferente. (MARCHESE, 2007)

Do ponto de vista da aplicação, QoS é a segurança de que seus requisitos serão satisfeitos.

De acordo com Marchese (2007, p.24) QoS é reconhecida em 3 categorias diferentes:

- **QoS intrínseca**: é diretamente provida pela rede e é qualificada por parâmetros objetivos como latência, perda de pacotes e estabilidade, abordados no capítulo 2.
- **QoS percebida**: é a qualidade percebida pelos usuários. Depende principalmente da qualidade da rede, mas é medida pelo MOS (*mean opinion score*), onde a qualidade do serviço é medida a partir de votações dos usuários, as quais variam de 1 a 5 (onde 1 é péssimo e 5 é ótimo).
- **QoS avaliada**: se refere a capacidade da rede de manter seu usuário. Isso está diretamente relacionado a questões de marketing, preços e suporte. Como por exemplo, o usuário aceitaria um problema de conexão em um serviço grátis ou de baixo custo, entretanto em um serviço de alto custo

como o de empresas ou organizações a tolerância do usuário já é diferente.

Um dos problemas encontrados quando se trata de QoS é que mesmo que os requisitos das aplicações ou de uma rede sejam atendidos perfeitamente nem sempre os usuários percebem variações no serviço para melhor ou pior. Nem sempre, no caso da QoS percebida, a qualidade da rede é analisada e classificada de forma justa ou correta a partir do MoS, pelo fato de ser um teste com muitas variáveis, tornando os resultados de difícil análise. (MARCHESE, 2007)

Por outro lado, a mudança para pior, passando despercebida pelo usuário, pode ser percebida pela aplicação, causando assim um mal funcionamento, o que então eventualmente será reportado pelo usuário.

Consequentemente a verificação da qualidade de serviços é um ponto extremamente importante no momento de determinar a necessidade ou não de mudanças ou manutenções.

Entretanto, a boa qualidade de serviço é o último passo de um longo processo que se inicia com o bom gerenciamento de redes.

1.3 GERENCIAMENTO DE REDES

Como já é de conhecimento de todos, a internet revolucionou os meios de comunicação da sociedade moderna. Desde sua criação na década de 60, sua usabilidade e impacto cresceram vertiginosamente tendo um aumento gigantesco nas últimas décadas.

Com esse aumento, a rede tornou-se uma necessidade básica em empresas, governos e até mesmo no dia a dia sendo utilizada por centenas de tipos diferentes de dispositivos. Consequentemente, tendo centenas de milhares de equipamentos conectados de uma organização a outra ou de uma casa a outra, era esperado que

eventualmente algum componente desta conexão imensa viesse a apresentar um mal funcionamento. (KUROSE, ROSS, 2013)

O gerente de redes, cujo papel é manter a rede funcionando de forma estável, e com possivelmente centenas de equipamentos de rede espalhados por uma área, claramente necessita de uma ferramenta de monitoração e administração das conexões e componentes. (KUROSE, ROSS, 2013)

1.3.1 Modelo FCAPS

Com o crescimento da importância das redes de computadores, viu-se como necessária uma forma mais sistemática para a resolução de problemas.

Pensando nisso *a International Organizations for Standardization* (ISO) ou Organização internacional para Padronização criou um modelo de gerenciamento de redes. Este modelo, conhecido como FCAPS (*Fail*, *Configuration*, *Accounting*, *Performance* and *Security*) foca em resolver questões como configuração, falhas, desempenho, segurança e contabilizações relacionadas a rede as quais foram divididas em áreas funcionais. (MOQADI, 2011)

As cinco áreas de gestão descritas neste modelo são:

- **Gestão de Performance**: Quantifica, mede, reporta, analisa e controla a performance dos componentes de rede (não somente um equipamento em si, mas também a rota que percorre).
- **Gestão de Falhas**: Tem como principal objetivo a detecção, determinação, o diagnóstico e resolução de falhas ou problemas, que vem a ser compreendido como qualquer incidente que cause o mal funcionamento do ambiente.

- **Gestão de Configuração**: Permite ao gerente de redes determinar os hardwares presentes na rede, seus respectivos *software*s e suas configurações. Permite também a instalação de novos dispositivos, alteração e atualização dos *software*s correntes.
- **Gerência de Contabilização**: Compreende a contabilização dos recursos e custos da rede referentes à utilização de dispositivos e provedoras, assim como o planejamento de crescimento e a detecção de utilizações não esperadas.
- **Gestão de Segurança**: Tem a função de garantir que as políticas de segurança do ambiente estão sendo utilizadas, monitorando o uso de recursos, acessos, históricos e o fluxo de dados e informações.

Tuncay Saydam e Thomas Magendanz, baseados nestes modelos e em estudos de caso em diferentes ambientes, concluíram que:

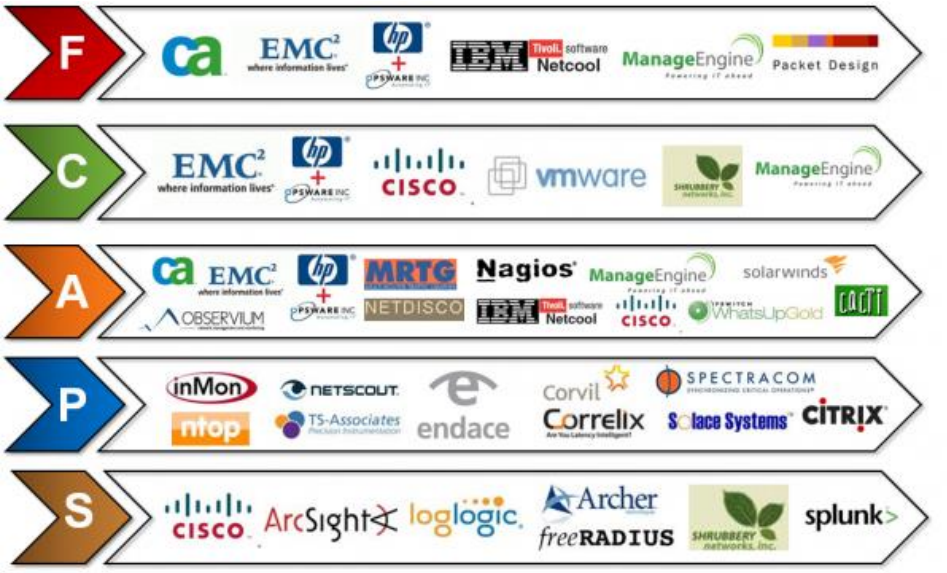
> “Gerenciamento de rede inclui o desenvolvimento, integração e a coordenação de elementos de hardware, *software* e humanos, para monitorar, testar, apurar, configurar, analisar, avaliar e controlar os recursos da rede, e de elementos, para satisfazer aos requisitos de performance operacional e qualidade de serviços em tempo real e a um preço justo. ” (SAYDAM, MAGENDANZ, 1996) [1]

Atualmente, com a popularização deste campo da gestão, diversos softwares foram desenvolvidos a fim de prestar assistência e tornar o monitoramento algo mais simples e rápido, como por exemplo o ArcSight (software de gerenciamento de segurança empresarial) e outros diversos programas, representados pela figura 1.

---

[1]"Network management includes the deployment, integration and coordination of the hardware, software and human elements to monitor, test, poll, configure, analyze, evaluate and control the network and element resources to meet the real-time, operational performance, and Quality of Service requirements at a reasonable cost." *(Tradução nossa)*

Figura 1 – Softwares de gerenciamento por campos de gestão:



Fonte: SNMP Center[1]

A fim de possibilitar o monitoramento e gerenciamento do ambiente de redes, é necessário que os dados contidos nos equipamentos pertencentes ao mesmo sejam adquiridos e salvos para que possam ser lidos e interpretados.

Necessita-se então de aplicar ao ambiente um protocolo capaz de adquirir tais informações.

No caso deste trabalho foi utilizado o protocolo SNMP.

1.4 SNMP

SNMP (*Simple network Management Protocol*) é um protocolo utilizado para o monitoramento de redes de computadores.

Diferente de seu antecessor, SGMP (*Simple Gateway Management Protocol*), o qual era utilizado apenas para a monitoração de roteadores de internet (*Gateways*),

[1]. *Disponível em http://snmpcenter.com/tag/fcaps/*

o SNMP pode monitorar não somente roteadores, mas também switches, impressoras, servidores, etc. (MAURO, SCHMIDT, 2005)

De acordo com Mauro e Scmidt:

> "As formas que você pode utilizar o SNMP variam do mundano para o exótico: É bem simples utilizar o SNMP para monitorar a saúde dos seus roteadores, servidores e outras partes do seu hardware de redes, mas você também pode utilizá-lo para controlar seus dispositivos de rede, mandar mensagens, ou tomar ações automáticas caso problemas apareçam" (MAURO, SCHMIDT, 2005, p. 13). [1]

Basicamente os equipamentos que utilizam este protocolo são divididos entre gerenciadores e agentes. Os gerenciadores, também conhecidos com NMS (*Network Management Stations*) são equipamentos que rodam alguma espécie de software que e capaz de interagir com o agente. (MAURO, SCHMIDT, 2005)

O gerenciador e responsável por requisitar informações do agente, receber informações geradas por *traps* e além disso gerar respostar automáticas para cada tipo de informação.

Os agentes são todos e qualquer *software* instalado a um equipamento conectado à rede e que tenha suporte ao protocolo SNMP. Estes são responsáveis por monitorar o hardware e enviar estas informações ao gerenciador. Além disso, como já mencionado, o agente pode setar *traps*, as quais enviam uma informação ao gerenciador caso uma mudança predeterminada pelo mesmo ocorra.

Entretanto nem todas as informações são pertinentes ao gerenciador. Para dividi-las, então, foi criado o SMI.

---

[1] The ways you can use SNMP range from the mundane to the exotic: it`s fairly simple to use SNMP to monitor the health of your routers, servers, and other pieces of network hardwre, but you can also use it to contol your network devices, page someone, or take other automatic actions if problems arise. *(Tradução nossa)*

1.5 SMI

A *Structure of Management Information*, ou em português, Estrutura de gerenciamento de informação provê uma forma de definir objetos gerenciados e seus comportamentos. (MAURO, SCHMIDT, 2005)

De acordo com James Kurose e Keith Ross SMI é:

> "[...] a linguagem usada para definir a informação de gerenciamento que reside em um ambiente de redes gerenciado. Tal linguagem de definição é necessária para garantir que a sintaxe e a semântica dos dados da monitoração da rede sejam bem definidas e compreensíveis. " (KUROSE, ROSS, 2013, p.766). [1]

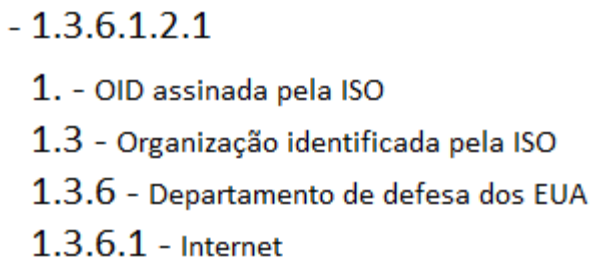
De forma geral este determina os dados que o agente pode e devera monitorar.

Estes dados, ou objetos, geram uma lista de informações as quais podem ser utilizadas pelo agente ou NMS. Cada um deles é diferenciado e classificado a partir de um OID.

1.6 OID

Os *Object Identifiers* ou identificadores de objeto são uma sequência numérica que tem como função identificar e diferenciar cada objeto determinado pelo SMI, como por representado pela figura 2.

Figura 2 – OID:



Própria fonte

[1][…]the language used to define the management information residing in a managed-network entity.Such definition language is needed to ensure that the syntax and semantics of the network management data are well defined and unambiguous.*(Tradução nossa)*

Os objetos são organizados em uma estrutura de árvore, onde há um número inicial denominado raiz. Todos os outros objetos que também tiverem ramificações serão denominados subárvores e os que não tiverem, folha. (MAURO, SCHMIDT, 2005)

A partir desta sequência numérica as MIBs, tratadas na seção 1.7, são capazes de reconhecer o objeto e a informação que este está disponibilizando e apresentando-a de forma entendível ao usuário.

1.7 MIB

Andando juntamente com o SMI estão as MIBs (*Management Information Base* ou Base de Gerenciamento de Informações) que contém todos os objetos ou informações que os agentes monitoram.

Enquanto o SMI define quais são os objetos monitorados, as MIBs são as definições dos objetos em si, podendo não só reconhecer seus respectivos OIDs, mas também o tipo de dado que contém. Além disso podem ser destinadas para a monitoração dos dados de apenas um objeto ou de vários.

> "[...] MIB pode ser imaginada como uma loja virtual de informações, mantendo objetos gerenciados, os quais coletivamente refletem o estado atual da rede. " (KUROSE, 2013, p.770)

Um agente pode conter diversas MIBs, dependo da necessidade de quem gerencia a rede, entretanto todos devem ter uma em comum conhecida como MIB-II, a qual provê informações básicas para o monitoramento de redes baseadas no protocolo TCP/IP.

Existem três tipos básicos de MIBs:

- **MIB-II**, evolução da não mais utilizada MIB-I, fornecendo diversas novas informações, dentre elas: número de pacotes, estado da interface, etc.

---

[1][...] MIB, can bethought of as a virtual information store, holding managed objects whose values colectively reflect the current "state" of the network. *(Tradução nossa)*

- **MIB Experimental**, as quais se encontram em fase de testes e geralmente são focadas em prover informações mais específicas, que se adequam apenas a certo tipo de cliente.
- **MIB Privada**, são aquelas que provêm informações específicas de cada equipamento, como por exemplo configuração, sendo capaz também de executar pequenas ações como abrir ou fechar portas. (MAURO, SCHMIDT, 2005)

Entretanto mesmo com a utilização das MIBS, o dado recebido nem sempre está "limpo", geralmente contém diversas informações não necessárias ou não desejadas. Para isso, foi utilizada a técnica de preparação de dados, na qual todos os dados não desejados são removidos, de forma que se tornam adequados para serem utilizados pelo sistema desenvolvido.

1.8 PREPARAÇÃO DE DADOS

Com a evolução da tecnologia a sociedade tonou-se capaz de gerar e capturar uma quantidade imensa de dados e informações. Sua importância em diversas áreas, como a da saúde ou governamental, foi aumentando gradualmente com o passar dos anos, e hoje tem um papel fundamental.

Esse aumento gigantesco da quantidade de dados gerados, criou uma necessidade de técnicas para sua leitura e tradução pois, na realidade, apenas uma pequena fração dos dados será realmente utilizada devido a sua complexidade para leitura ou devido a presença de informações desnecessárias. (KANTARDZIC, 2011)

Essa necessidade criou a área hoje conhecida como *Data Mining* (Mineração de dados) ou também conhecida como *Knowledge Discovery from Data* (KDD). Esta técnica baseia-se na captura de informações relevantes que constituem algum tipo de

conhecimento. Estas informações estão presentes em grandes bancos de dados, armazéns de dados, na internet, entre outros. (HAN, PEI, KAMBER, 2012)

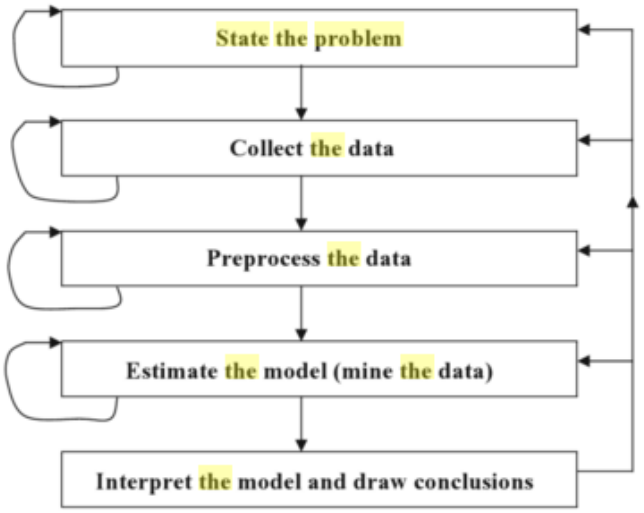
Kantardzic classifica a mineração como:

> "Mineração de dados é a procura por informações novas, valiosas e não triviais em grandes volumes de dados. É um esforço cooperativo de humanos e computadores. " (KANTARDZIC, 2011, p. 2) [1]

Devido às diversas descrições do que é realmente a mineração de dados, e a definição de suas fases e processos, há uma pequena divergência entre diferentes autores.

De acordo com Mehmed Kantardzic (2011, p.7), o processo de mineração de dados é divido em 5 fases: A determinação do problema, a coleta de dados, o pré-processamento dos dados, a mineração, e as conclusões, as quais são representadas pela figura 3.

Figura 3 – *Fases do processo de mineração segundo KANTARDZIC:*



Fonte: KANTARDZIC (2011, p.9)

[1] Data Mining is the search for new, valuable, and nontrivial information in large volumes of data. It is a cooperative effort of humans and computers. *(Tradução nossa)*

1.8.1 Determinação do problema e formulação da Hipótese:

Nesta fase são determinados os problemas com os dados e o que deve ser retirado deste. Deve-se haver uma intensa interação entre o especialista de mineração de dados e o gerente da aplicação ou sistema. Desta forma todas as necessidades serão determinadas e caso esta interação seja bem-sucedida, esta fase perdurará durante todo o processo de preparação e mineração dos dados. (KANTARDZIC, 2011).

1.8.2 Coleta dos dados:

Esta fase preocupa-se com a geração e coleta de dados. Há basicamente duas possibilidades:

- ***Designed Experiment* (Experimento Planejado)**: Ocorre quando a geração de dados está sob o controle do especialista ou modelador.
- ***Observation Approach* (Abordagem de Observação)**: Ocorre quando o especialista não pode influenciar no processo de geração de dados.

O modelo observacional, também conhecido como geração de dados randômica, está presente na maioria das aplicações de mineração de dados. (KANTARDZIC, 2011).

1.8.3 Pré-processamento ou Preparação de dados:

No modelo observacional os dados são provenientes em geral de armazéns de dados (*Data Warehouses*), bacos de dados, etc. Assim o pré-processamento de dados é constituído de duas fases:

A primeira fase é a detecção e remoção de discrepâncias. Discrepâncias são dados não esperados ou de nenhuma utilidade que estão presentes no banco. Estes são gerados normalmente a partir de erros no código, na gravação de dados ou até mesmo naturalmente sem causa definida. Estas informações podem afetar seriamente a

absorção de dados posteriormente. Consequentemente é necessário lidar com estes dados anteriormente à leitura ou utilização dos mesmos. (KANTARDZIC, 2011).

Para isso há duas formas básicas:

- Detecção e remoção de discrepâncias como uma parte da fase de preparação dos dados.
- Desenvolvimento de um código ou métodos de absorção de dados que ignorem o que não é esperado.

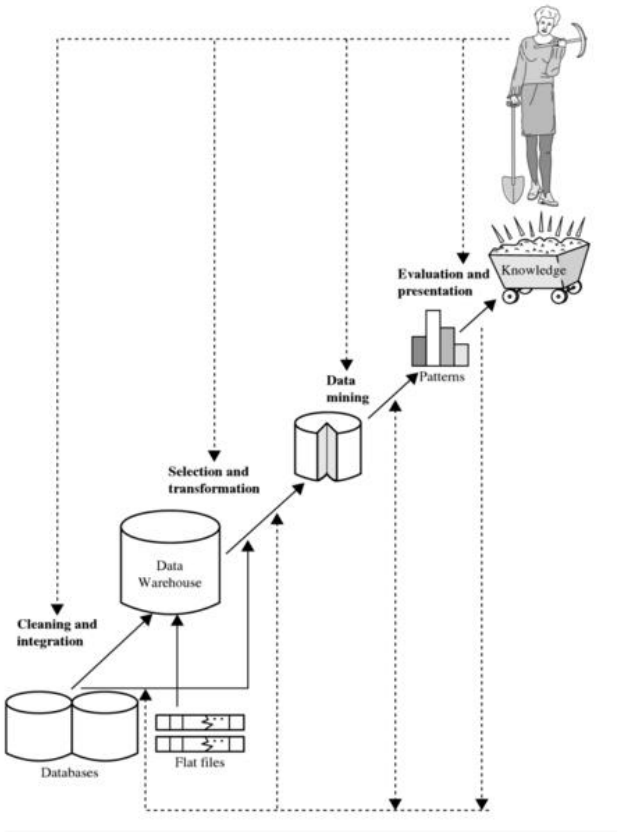
A segunda fase do pré-processamento de dados é constituída pela redução, codificação, e seleção de recursos.

Um exemplo de redução ou codificação de informações seria no caso de duas entradas, uma pertencente ao grupo dos números naturais (inteiros positivos) e outra pertencente ao grupo dos inteiros (positivos e negativos). Estas duas entradas não teriam o mesmo peso posteriormente e então seriam tratadas de modo diferente, o que poderia causar erros ou mal funcionamento. Consequentemente é necessária a redução de um deles a fim de igualar os campos. (KANTARDZIC, 2011).

Já de acordo com Jiawei Han, Jian Pei e Micheline Kamber (2012, p.6) o processo de mineração é dividido em 7 partes: A limpeza de dados, a integração, a seleção, a transformação, a mineração, a avaliação de padrões e a apresentação de conhecimento conforme apresentado na figura 4.

A redução de dados, neste modelo, não e considerada como uma fase do pré-processamento de dados apesar de muitas vezes ser um recurso.

Figura 4 – *Fases do processo de mineração segundo HAN, PEI e KAMBER*:



Fonte: HAN, PEI, KAMBER (2012, p.7)

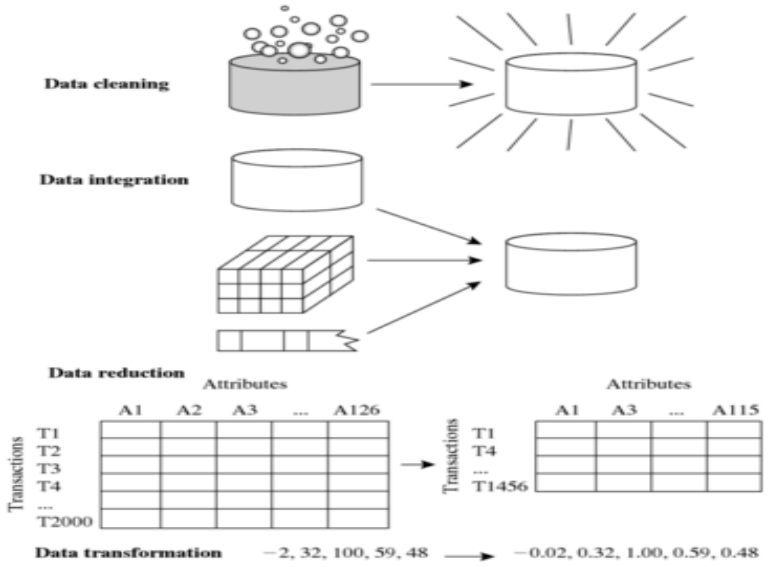
Neste modelo o pré-processamento de dados seria constituído pelas quatro primeiras fases do processo, as quais preparam as informações para a mineração, como representado na figura 5:

- **Limpeza de dados**: Nesta fase todos os dados que diferem do que é esperado são removidos.
- **Integração de dados**: Onde diversas fontes de dados são combinadas para então serem salvas em uma única base.
- **Seleção de dados**: Onde os dados relevantes são resgatados do banco.
- **Transformação de dados**: Onde os dados são transformados em formas mais simples para o processo de mineração.

A redução de dados, neste modelo, não é considerada como uma fase do pré-processamento de dados apesar de muitas vezes ser um recurso utilizado neste ponto do processo.

Basicamente a redução de dados gera uma representação reduzida dos mesmos, mantendo sua integridade o máximo possível. (HAN, PEI, KAMBER, 2012)

Figura 5 – ***Forma de pré-processamento de dados:***



Fonte: HAN, PEI, KAMBER (2012, p.87)

No próximo capitulo será explicado o conceito de rede de computadores e suas diferenciações em relação aos meios de transmissão e seus tamanhos.

2. AMBIENTE DE REDES

Como já mencionado anteriormente a conexão de equipamentos e/ou sistemas em rede, a fim de trocar informações de forma simples e rápida, tornou-se uma necessidade.

Uma rede de computadores é formada a partir do momento em que dois equipamentos autônomos são conectados entre si e começam a trocar informações. Estes são normalmente interligados por cabos de cobre, cabos de fibra ótica, micro-ondas, infravermelho ou até mesmo por satélite, e a partir destas conexões geram redes de diferentes tipos e tamanhos. (Tanenbaum, 2011)

2.1 TIPOS DE TRANSMISSÃO

Existem duas classificações de meios de transmissão de dados entre equipamentos, os meios guiados e os meios não guiados.

Os meios de transmissão guiados são aqueles que utilizam equipamentos físicos para o transporte de informações. De forma geral todos têm diferentes velocidades e níveis de confiabilidade.

Segundo Tanenbaum e Wetherall (2011, p.95) este grupo é composto por:

- Mídia Magnética.
- Par trançado.
- Cabo coaxial.
- Rede Elétrica.
- Fibras Óticas.

Já os meios de transmissão não guiados são os que não utilizam nenhum tipo de cabeamento ou conexão física para o transporte de informações. Este tipo de transmissão é basicamente composto por um emissor de ondas eletromagnéticas e uma

antena receptora e seu uso é determinado pelo intervalo de frequência do espectro de luz que utilizam.

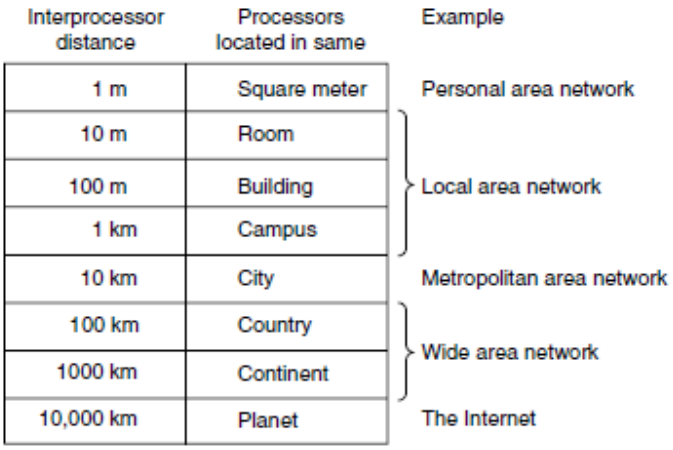
Os principais tipos de ondas, segundo Tanenbaum (2011, p.105), são:

- Ondas de rádio.
- Micro-ondas.
- Infravermelho.
- Transmissão por luz.

2.2 ESCALA DE REDES

Além da diferenciação por tipo de transmissão há também a classificação por escala. As redes podem ser divididas entre redes pessoais, locais, metropolitanas, redes amplas e a própria internet, como apresentado na figura 6.

Figura 6 – *Classificações de rede por escala*



Fonte: TANENBAUM (2011, p.18)

- ***Personal Area Network (PAN):*** Permite que equipamentos se comuniquem no alcance de uma pessoa.

- **Local Area Network (LAN):** Rede privada que opera em um escritório ou em uma casa.
- **Metropolitan Area Network (MAN):** Rede que compreende a area referente a uma cidade.
- **Wide Area Networks (WAN)*:*** Compreende uma grande área como um país inteiro ou até mesmo um continente.

Independente dos diferentes tipos de transmissões e tamanhos de rede, todas as redes necessitam de um bom gerenciamento a fim de prevenir ou corrigir os erros e falhas mais comuns dos ambientes, que são:

- **Perda de Pacotes**: Todas as informações transmitidas em uma rede são fragmentadas e transformadas em o que chamamos de pacotes. Toda vez que um destes pacotes não chega ao seu destinatário temos uma perda de pacote, o que interfere na formação final da informação.
- **Erros**: Os pacotes recebidos são incompletos ou malformados.
- **Alta Utilização**: toda rede tem seu limite de transmissão e de recepção, quando este limite é atingido e informações continuam sendo transmitidas, pode-se ter erros ou até mesmo perda de pacotes.

No próximo capítulo serão descritas todas as características da aplicação desenvolvida assim como diagramas explicando seu funcionamento.

3. DESENVOLVIMENTO

O sistema desenvolvido tem a função de buscar informações dos equipamentos de rede a partir do protocolo SNMP, ler estes dados e então apresentar ao usuário os dados obtidos juntamente com a análise dos mesmos e a possível resolução dos problemas encontrados.

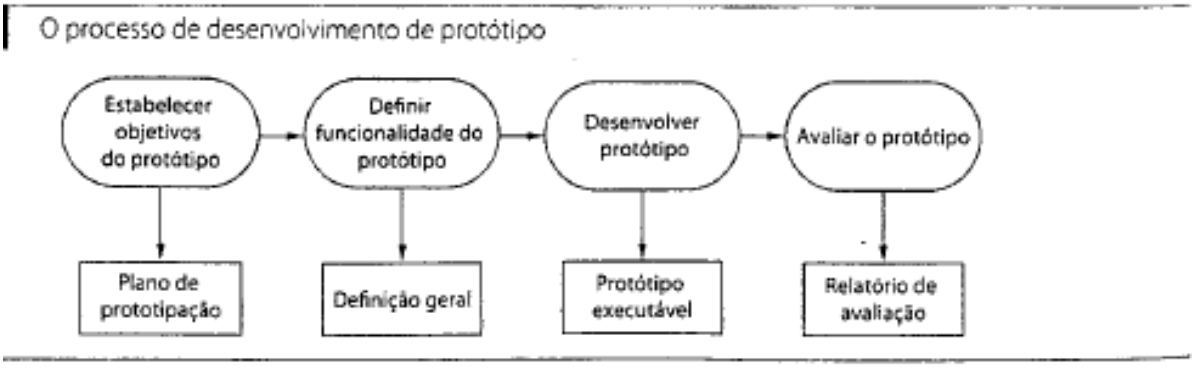
3.1 TÉCNICA DE DESENVOLVIMENTO

Para o desenvolvimento deste sistema foi utilizado o modelo de prototipação evolucionária devido a facilidade na determinação dos requisitos do projeto. Além disso não haverá mudança nos requisitos, o que cancelará a necessidade de refazer as fases iniciais do processo.

Segundo Sommerville (2011, p.30), o protótipo é a versão inicial do programa usada para determinar possíveis problemas e suas soluções, além de poder ser utilizado para antecipar eventuais mudanças.

O processo de prototipação representado pela figura 7, inicia-se com a determinação dos objetivos do protótipo como por exemplo desenvolver um sistema para prototipar a interface do usuário, sendo que um mesmo protótipo não pode cumprir todos os objetivos determinados pelo cliente. (SOMMERVILLE, 2011)

Figura 7 – Processo de Prototipação segundo Sommerville:



Fonte: Sommerville (2011, p.20)

A próxima fase do processo é a definição das funcionalidades do protótipo. Nesta serão determinadas as funções que serão adicionadas ou removidas do protótipo, assim como deixar de lado requisitos como velocidade ou gerenciamento de erros. (SOMMERVILLE, 2011)

Esta fase é seguida do desenvolvimento do protótipo, que após ser concluído, traz consigo a necessidade de uma avaliação, a qual é a última fase do processo de prototipação.

A avaliação é baseada em cada objetivo dos protótipos. Após um tempo de utilização os usuários serão capazes de determinar falhas ou divergências de requisitos.

Tendo conhecimento do processo de desenvolvimento do protótipo, dá-se início ao modelo de prototipação evolucionária, no qual um sistema simples é ampliado a medida que novos requisitos são encontrados.

Este modelo, representado pela figura 8, tem como principais vantagens a rapidez e facilidade de comunicação no atendimento às necessidades do ciente. Entretanto tem como desvantagem alguns problemas de gerenciamento devido a constante aumento de requisitos assim como problemas com manutenção.

Figura 8 – Processo de Prototipação Evolucionária:

Fonte: Sommerville (2011, p.21)

3.2 PROCESSO DE DESENVOLVIMENTO

3.2.1 Definição de requisitos

Os requisitos foram determinados baseando-se em outros sistemas de gerenciamento de ambientes de redes como por exemplo o IBM NetCool[1] e CACTI[2].

O IBM NetCool permite a configuração, monitoração, descoberta de novos equipamentos, gerenciamento de falhas, criação de relatórios, entre outras diversas funções as quais serviram de base para a determinação de requisitos do sistema a ser desenvolvido. (IBM, 2016)

Além disso, foi levado em conta o baixo conhecimento da área por parte do público do sistema, de forma que um dos principais requisitos se tornou a necessidade de uma interface simples e autoexplicativa.

3.2.2 Projeto de sistema

Então, todos os requisitos foram divididos entre requisitos funcionais e não funcionais:

- **Requisitos funcionais*:*** Como funções do sistema temos: análise de erros, utilização, perda de pacotes, versão do sistema do equipamento, estado das *fans*, status das interfaces do equipamento, logs do sistema, tempo de funcionamento, e último boot.

[1]. Disponível em: http://www.cacti.net/

[2]. Disponível em: http://www-03.ibm.com/software/products/pt/netcool-network-management

É necessária também a configuração dos equipamentos de redes para que forneçam ao sistema as informações requisitadas, assim como a tradução destas informações para que se tornem entendíveis ao sistema e ao usuário.

- **Requisitos não funcionais:** Há a necessidade de uma interface intuitiva e de fácil entendimento de forma que qualquer usuário tenha a capacidade de entender as informações e resultados apresentados. Além disso o sistema não pode apresentar erros no momento de obtenção de dados, o que pode comprometer o resultado.

3.2.3 Implementação:

Utilizando a linguagem Java foram desenvolvidas inicialmente as telas de seleção de dados e do relatório. A tela de relatório analisa o estado das opções presentes na tela anterior a fim de determinar exatamente os dados a serem buscados.

Para a obtenção das informações de um equipamento de rede utilizando o protocolo SNMP, necessita-se das MIBs, as quais farão a leitura e tradução destes dados de forma que o usuário já possa entender. Entretanto a forma em que os dados são entregues pela MIB não são adequados para que sejam utilizados pelo sistema, a informação contém muitos dados desnecessários, representados pela figura 8, que acabam por quebrar a lógica do sistema.

Para resolver este problema utilizou-se a técnica, já mencionada neste trabalho, de preparação de dados.

O comando a seguir, no qual é determinada a versão do SNMP, o usuário e o IP do equipamento desejado, salva todos os dados obtidos no arquivo teste.txt:

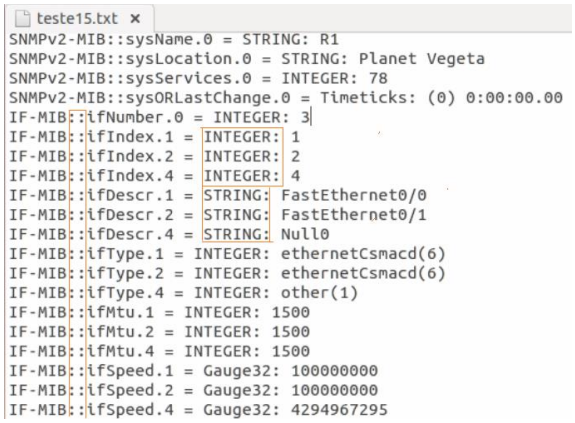
```
snmpwalk -c -v2 VegetaRocks 10.0.0.2 > teste1.txt
```

Posteriormente, todos os dados não desejados são removidos utilizando-se comandos de edição de texto:

```
tr -d ":" <teste.txt> teste1.txt
```

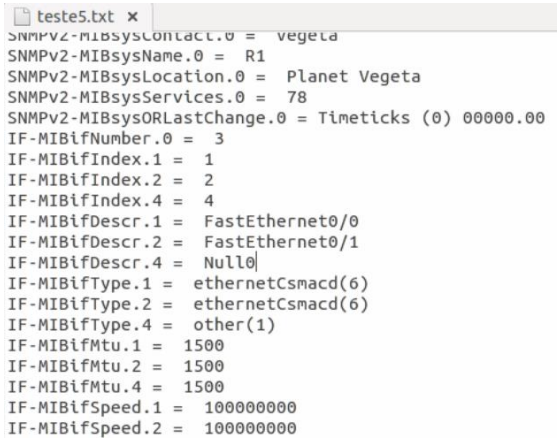
E então o arquivo resultante é salvo, tendo todos os dados indesejados removidos, como representado pela figura 9.

Figura 9 –Dados antes do processo de preparação



Própria fonte

Figura 10 –Dados após o processo de preparação



Própria fonte

Seguindo o modelo de prototipação, cada pesquisa por dado selecionado foi desenvolvida separadamente.

Para a busca e obtenção dos dados referentes a cada MIB foi utilizada a função properties:

```
       if(porta){

                    textArea.append("\n\n#################### Status de Interface
######################");

                    textArea.append("\n\nStatus da porta 1 por configuracao---> "
+ props1.getProperty("IF-MIBifAdminStatus.1"));

                    stat1 = (props1.getProperty("IF-MIBifAdminStatus.1"));

                    textArea.append("\nStatus da porta 2 por configuracao---> "
+props1.getProperty("IF-MIBifAdminStatus.2"));

                    stat2 = (props1.getProperty("IF-MIBifAdminStatus.2"));

                    textArea.append("\nStatus     atual     da     porta     1--->
"+props1.getProperty("IF-MIBifOperStatus.1"));

                    stat3 = (props1.getProperty("IF-MIBifOperStatus.1"));

                    textArea.append("\nStatus     atual     da     porta     2--->
"+props1.getProperty("IF-MIBifOperStatus.2"));

                    stat4 = (props1.getProperty("IF-MIBifOperStatus.2\n"));



                    if (stat1.equals(stat3)){



                            textArea.append("\n\nSuas interfaces estao OK");

                        }

                    else{textArea.append("\n\nAlgumas de suas interfaces entao em
estado DOWN.\nE indicada a checagem da configuracao das interfaces e
cabeamentos.\nCaso o problema persista, chame um tecnico");

                        }        }
```

Como pode ser observado, inicialmente a função busca pela MIB especificada e então a exibe na caixa de relatório.

Assim que os dados são selecionados, um cálculo é feito a fim de determinar se a rede apresenta um problema ou não:

- Inicialmente os dados são salvos em uma variável:

```
Int inoc1 = Integer.parseInt(props1.getProperty("IF-MIBifInOctets.1"));
```

- E então o cálculo é feito para localizar as falhas (caso existam):

```
if ((inerr1) > (inoc1/100) || (inerr2) > (inoc2/100) || (outerr1) > (outoc1/100) || (outerr2) > (outoc2/100))
```

Caso falhas sejam encontradas o programa gerará um alerta informando o erro, caso contrário será informado que a rede está “OK”.

Finalmente após todas as funções estarem corretas, foram agrupadas, dando origem a primeira versão do projeto final.

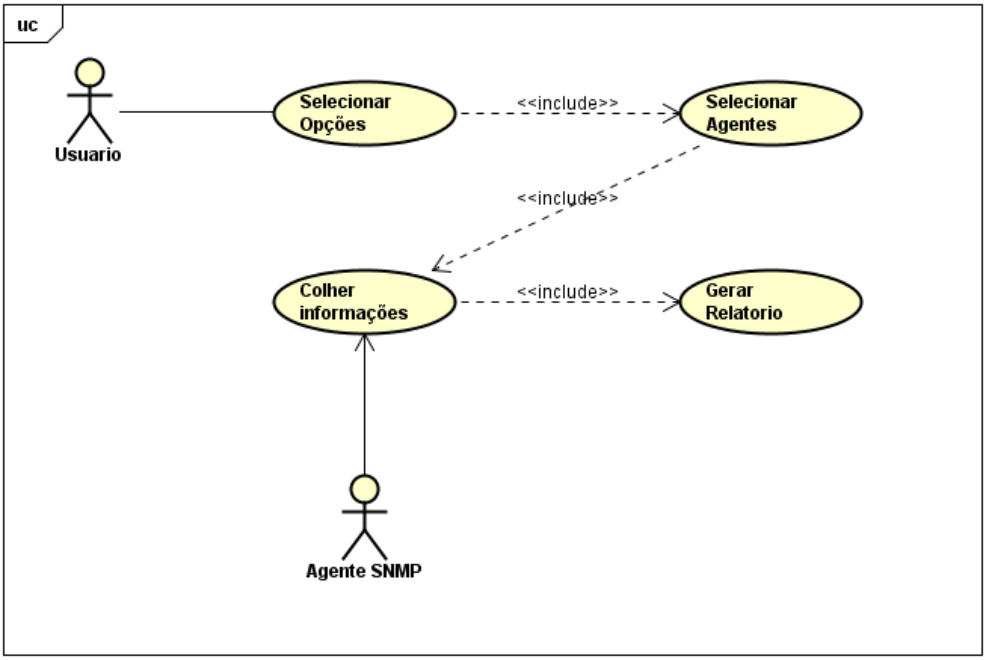
3.3 DIAGRAMA DE CASO DE USO

O diagrama de caso de uso tem a finalidade de apresentar ao usuário o funcionamento do sistema de forma simplificada. O diagrama mostra a interação dos equipamentos de rede com o sistema e as ações tomadas por este a fim de gerar e apresentar o resultado final representado pela figura 10.

O usuário tem inicialmente seleciona as opções de dados desejados. Assim que terminar a seleção deve fornecer, na tela de geração de relatório, as informações referentes ao agente SNMP e então requisitar o relatório.

Antes do relatório ser gerado, as informações são requisitadas ao agente informado pelo usuário. As informações fornecidas são lidas e preparadas para a leitura do sistema. Então, finalmente o sistema lê os dados gerados e fornece o relatório.

Figura 11 - Diagrama de Caso de Uso



Própria fonte

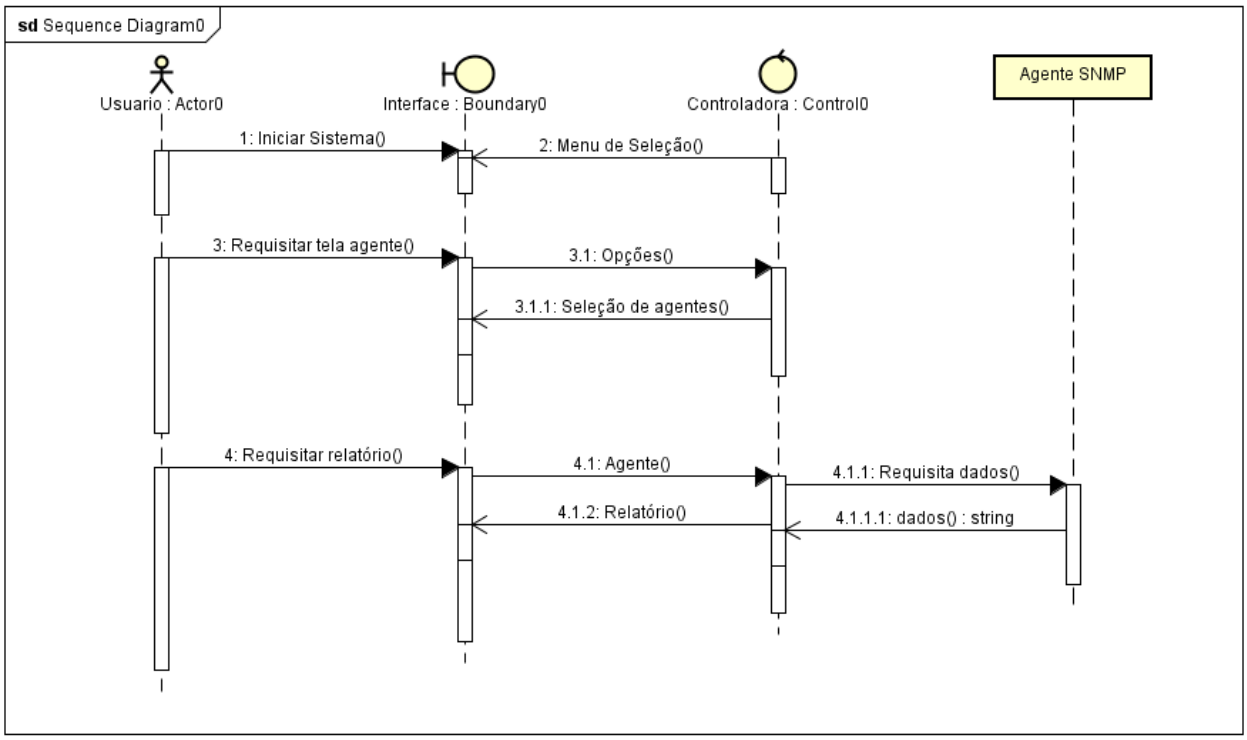
3.4 DIAGRAMA DE SEQUÊNCIA

Os diagramas de sequência ilustram a graficamente a sequência de acontecimentos demonstrados pela figura 12.

Neste diagrama, representado pela figura 12, são ilustradas as operações necessárias para que o relatório final seja gerado. Após a inicialização do sistema, e a seleção dos dados desejados, há a ocorrência do método de abertura da tela de seleção de agente, e de armazenamento das opções selecionadas.

Após a determinação do agente, com a requisição de um relatório, um novo método é utilizado para a inicialização do script, o qual fará a requisição dos dados ao agente e a leitura e preparação dos mesmos. O sistema então fará a leitura dos dados resultantes e gerará o relatório ao usuário.

Figura 12– Diagrama de Sequência



Própria fonte

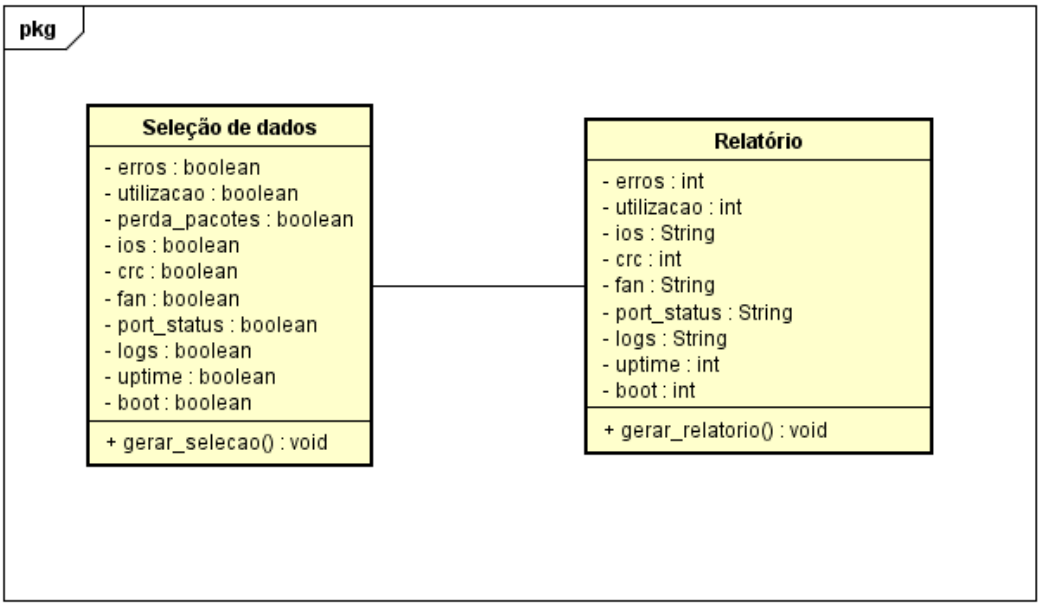
3.5 DIAGRAMA DE CLASSE E ATIVIDADE

O diagrama de classe apresenta as principais classes do sistema, juntamente com seus atributos, métodos e seus respectivos relacionamentos.

Pode-se observar na figura 13 todas as variáveis referentes a tela de seleção de dados assim como o método responsável pela obtenção do estado das opções (true ou false) e apresentação da tela de seleção de agente. Além disso, há a tela de relatório, na qual pode-se observar todas as variáveis referentes às opções da tela anterior e o método responsável pela inicialização do script, responsável pela obtenção e preparação dos dados do agente, e geração do relatório.

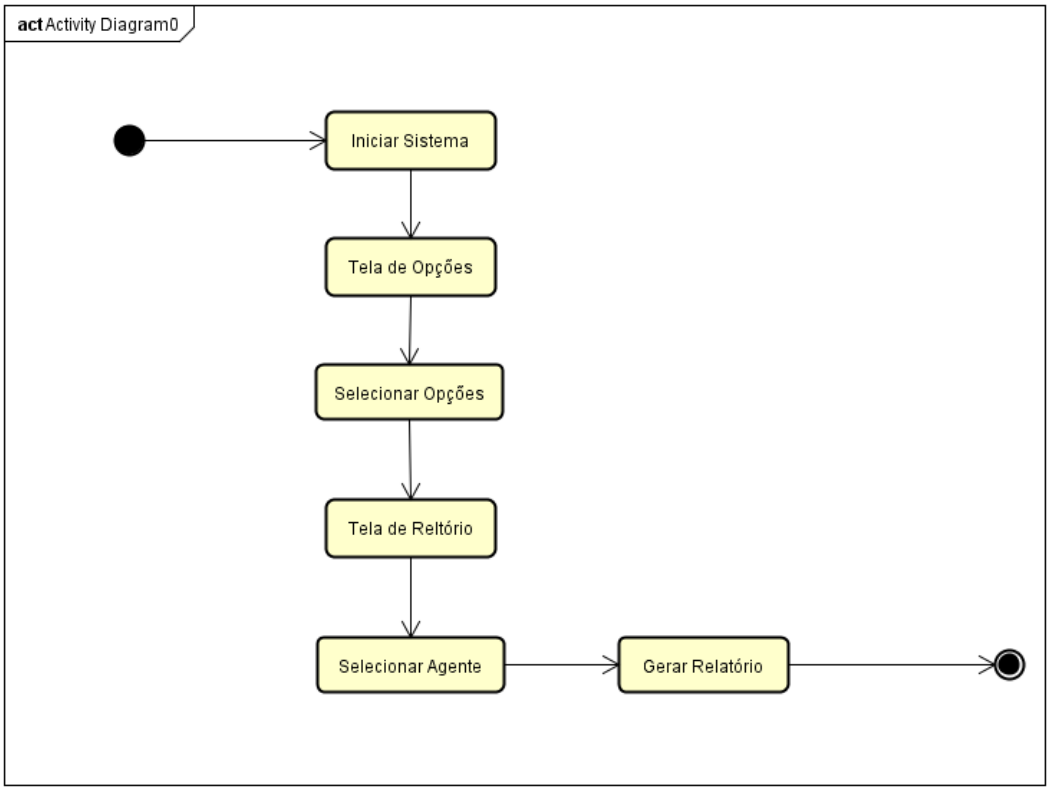
Já o diagrama de atividades, representado pela figura 14, apresenta de forma simplificada as principais funções e telas do sistema, facilitando a compreensão de seu funcionamento.

Figura 13 – Diagrama de Classes



Própria fonte

Figura 14 – Diagrama de Atividades



Própria fonte

3.6 SISTEMA FINAL

As próximas figuras retratam a primeira versão do projeto.

Figura 15 – Pagina Inicial

Própria fonte

As únicas telas existentes no momento são a tela inicial, representada pela figura 15, seleção dos dados do relatório, figura 16, a determinação do equipamento alvo e a geração do relatório, representada pela figura 17.

Futuramente pretende-se adicionar diversas outras funções ao programa como monitoração ativa, armazenamento de resultados em banco de dados, análise de histórico, etc...

A figura 15 refere-se a entrada do sistema onde há simplesmente as opções de iniciar o programa ou fecha-lo.

Figura 16 – Pagina de Seleção

Própria fonte

A figura 16 apresenta as opções disponíveis para o usuário. O relatório final irá variar de acordo com as opções selecionadas, onde cada uma trará um dado diferente.

As opções disponíveis são:

- **Erros / CRC:** Tanto a opção de erros quanto a opção de CRC apresentam a quantidade de erros lógicos presentes em cada interface do equipamento alvo. Então, será feito um cálculo baseado na quantidade total de pacotes transferidos pela interface e na quantidade de erros. Caso os erros passem de 1% o programa indica como necessária a checagem dos cabeamentos ou da configuração das interfaces relacionadas a fim de corrigir os erros. Estas informações são adquiridas pelo sistema a partir do código:

```
int inerr1 = Integer.parseInt(props1.getProperty("IF-MIBifInErrors.1"));
```

- **Utilização:** O sistema checa a utilização das interfaces e caso esta ultrapasse 75 % da banda total, isto poderá causar uma perda de pacotes ou erros. Será então indicado ao usuário o aumento da banda ou o melhor controle da utilização.
  Estas informações são adquiridas pelo sistema a partir do código:

```
int inerr2 = Integer.parseInt(props1.getProperty("IF-MIBifSpeed.2"));
```

- **Perda de pacotes:** Esta opção checa a perda de pacotes das interfaces do equipamento monitorado e caso o número de perda de pacotes seja maior que 5% do total de pacotes, indica ao usuário a checagem dos cabeamentos.
  Estas informações são adquiridas pelo sistema a partir do código:

```
int outerr1 = Integer.parseInt(props1.getProperty("IF-MIBifInDiscards.1"));
```

- **IOS:** Esta opção mostra a versão do *software* do roteador.
  Estas informações são adquiridas pelo sistema a partir do código:

```
textArea.append("\n\nDescricao do Sistema---> +props1.getProperty ("SNMPv2- MIBsysDescr.0"));
```

- **FAN:** Mostra o estado das *fans* presentes no roteadores. Caso alguma delas esteja danificada, a troca será recomendada.
  Estas informações são adquiridas pelo sistema a partir do código:

```
textArea.append("\n\Status da Fan---> +props1.getProperty ("CISCO-ENVMON-MIBciscoEnvMonFanState.1"));
```

- **Status das Portas:** Nesta opção são apresentados os atuais estados de todas as interfaces do equipamento assim como o estado que deveriam estar de acordo com a configuração. Desta forma pode-se detectar interfaces danificadas e apontar a necessidade de uma checagem de configuração, do equipamento ou do cabeamento.
  Estas informações são adquiridas pelo sistema a partir do código:

```
textArea.append("\nStatus da porta 2 por configuracao---> "
+props1.getProperty("IF-MIBifAdminStatus.2"));
```

- **Logs:** São apresentadas as logs do equipamento.
  Estas informações são adquiridas pelo sistema a partir do código:

```
textArea.append("\n Logs Do Sistema---> " props1. getProperty ("jnx
SyslogEntry.1"));
```

- **Uptime:** Tempo que o equipamento está conectado à internet (Neste caso somente quando o equipamento que se conecta a WAN).
  Estas informações são adquiridas pelo sistema a partir do código:

```
int inoc1 = Integer.parseInt(props1.getProperty("DISMAN-EVENT-
MIBsysUpTimeInstance"));
```

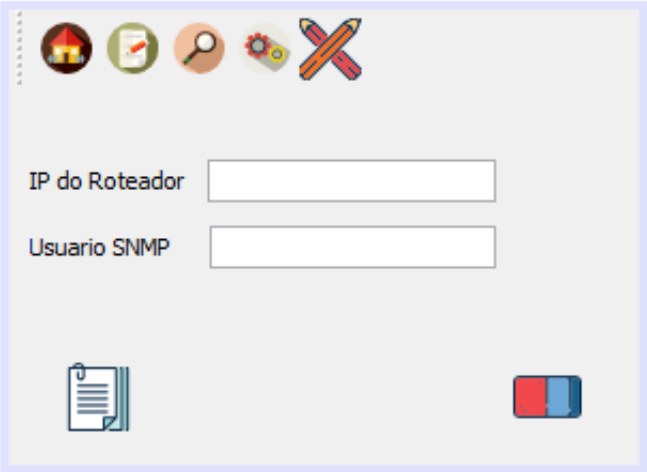
***Last boot*:** O tempo que o equipamento está ligado e a forma que foi ligado (*power-on* significara por falha de energia, e *reload*, por um *reboot* normal).

Estas informações são adquiridas pelo sistema a partir do código:

```
textArea.append("\nUltimo boot do sistema---> " +props1. getProperty
("ARROWPOINT-BOOTEXT-MIB"));
```

Após serem selecionadas as opções, o usuário deverá clicar na opção de relatório representado pela imagem do “papel e lápis” e então será levado a próxima página.

Figura 17 – Pagina de introdução de dados do equipamento



Própria fonte

A figura 17 representa a tela de determinação do equipamento a ser monitorado. Nesta tela o usuário deverá fornecer o IP do roteador e o nome do usuário SNMP (necessário para a requisição SNMP).

Finalmente, após a introdução dos dados, o usuário deverá clicar no ícone do relatório e uma página será gerada, apresentando o resultado dos dados requisitados, os problemas encontrados e suas possíveis soluções.

Uma próxima versão do projeto pretende identificar automaticamente os equipamentos de rede e apresentá-los para serem selecionados.

O próximo capitulo apresentará os ambientes utilizados e os testes realizados no mesmo.

4. CENÁRIOS

Como já mencionado anteriormente, foram desenvolvidos dois ambientes de redes utilizando o programa GNS3.

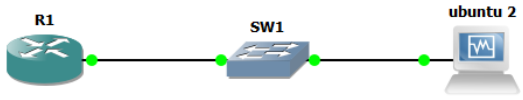
O GNS3 ou *Graphical Network Simulator-3* é um software de emulação de redes que permite criação de equipamentos e máquinas virtuais e a conexão destes com equipamentos reais. Desta forma é possível criar facilmente diversos tipos de ambientes de redes.

O primeiro ambiente teve o objetivo de simplesmente permitir o desenvolvimento das funções do sistema, assim como seus respectivos testes.

A figura 18 representa o primeiro ambiente com apenas um roteador, um switch e a máquina virtual com sistema Ubuntu.

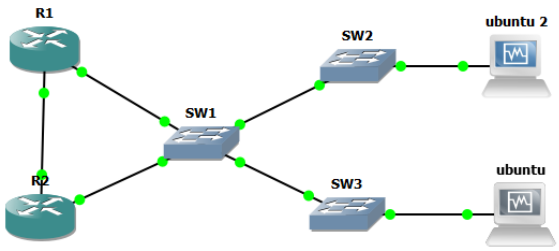
Já figura 19 representa o segundo ambiente de redes desenvolvido. Este possuindo dois roteadores, três switches e duas máquinas virtuais. Este ambiente representa o ambiente de um futuro cliente, e, além de ser maior que o primeiro, apresenta algumas falhas, as quais o sistema desenvolvido identificara e apresentara suas respectivas soluções.

Figura 18 – Ambiente 1



Própria fonte

Figura 19 - Ambiente 2



Própria fonte

4.1 TESTES REALIZADOS E RESULTADOS OBTIDOS

Com o objetivo de testar o sistema, ele foi utilizado para verificar o ambiente 2, que como já citado, apresenta problemas de rede.

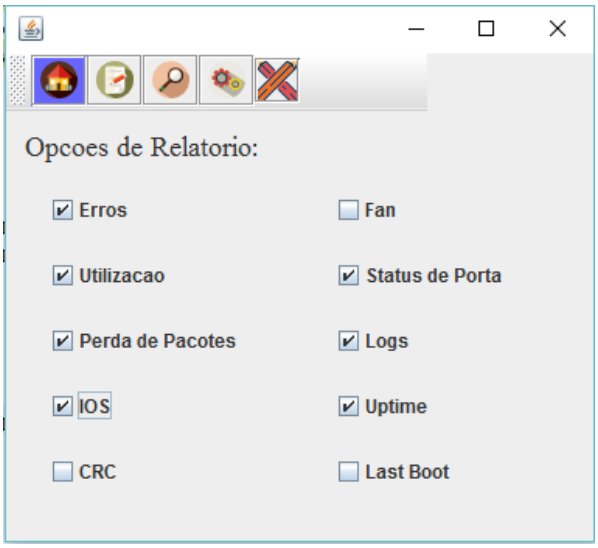
O Sistema foi instalado na máquina virtual número 2, que se encontra conectada diretamente no switch 2.

O equipamento focado pelo sistema foi o primeiro roteador, o qual apresenta duas conexões, uma para o switch 1 e outra para o roteador 2.

Para este teste foram selecionadas apenas as opções de erros, perda de pacotes, status de porta, logs, uptime, IOS e utilização. A opção de CRC assim como a de fan não foi selecionada pois pelo equipamento ser virtual não há probabilidade de problemas com fan ou erros físicos. A opção de last boot também não foi selecionada por não haver formas diferentes de boot em um ambiente virtual.

Então, as opções selecionadas foram:

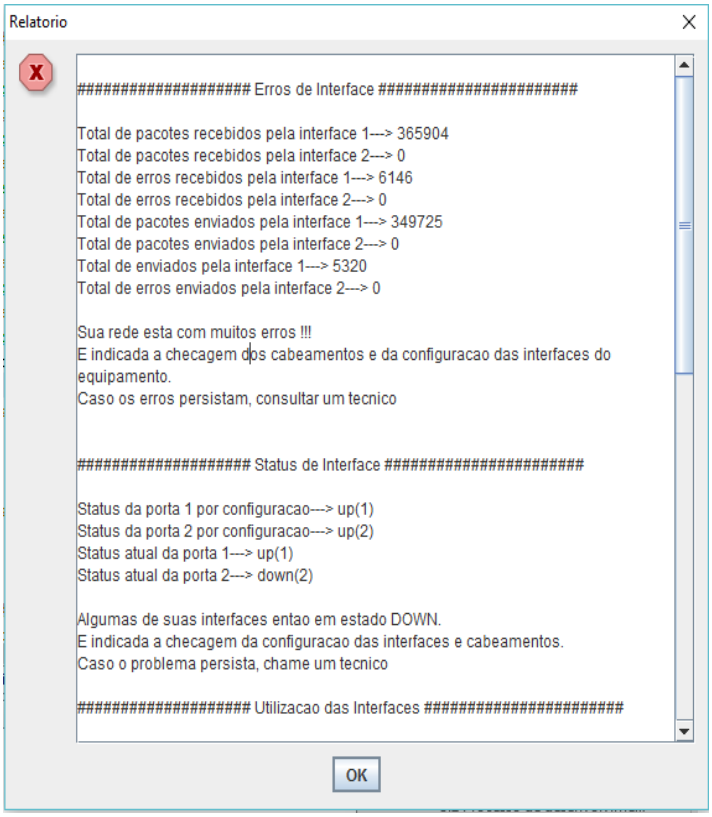
Figura 20 – Seleção de opções



Própria Fonte

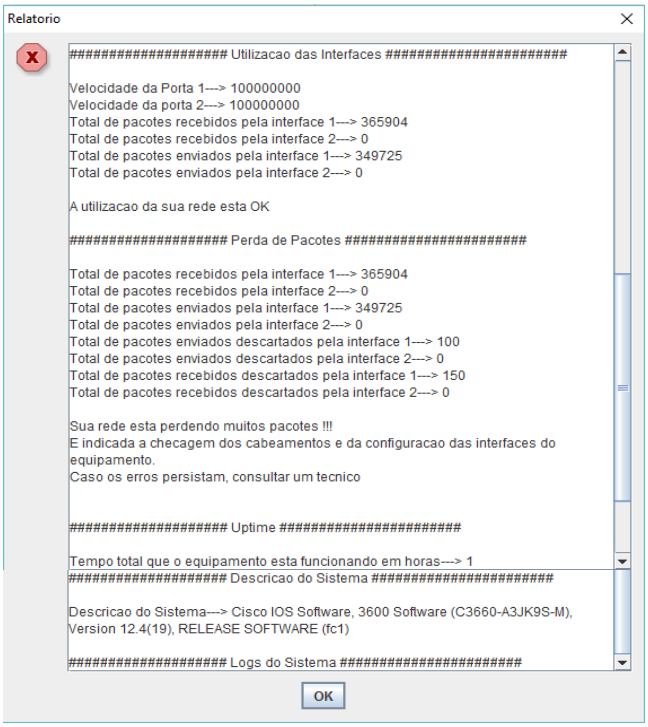
O resultado obtido pode ser observado nas figuras 21 e 22:

Figura 21 – Resultados



Própria Fonte

Figura 22 – Resultados



Própria fonte

Pode-se observar que há falhas na rede relacionadas a número de erros, status de interface e perda de pacotes.

4.2 RESOLUÇÃO DOS PROBLEMAS

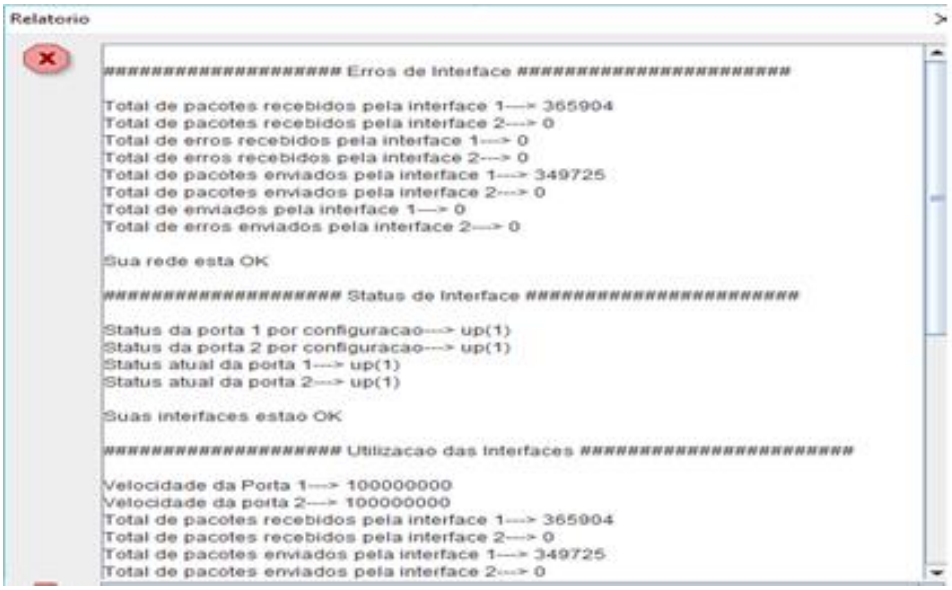
De acordo com os resultados apresentados pelo programa, o equipamento em questão apresenta problemas de erros e perda de pacotes em sua primeira interface e a sua segunda interface não está funcionando corretamente.

A partir destes dados, a rede passou por uma checagem e descobriu-se problemas de configuração no equipamento que se conectava ao roteador na interface 1.

Além disso o roteador, que se encontrava conectado na interface 2 do roteador apresentava suas interfaces fechadas, consequentemente a interface do roteador primário não apresentava conectividade.

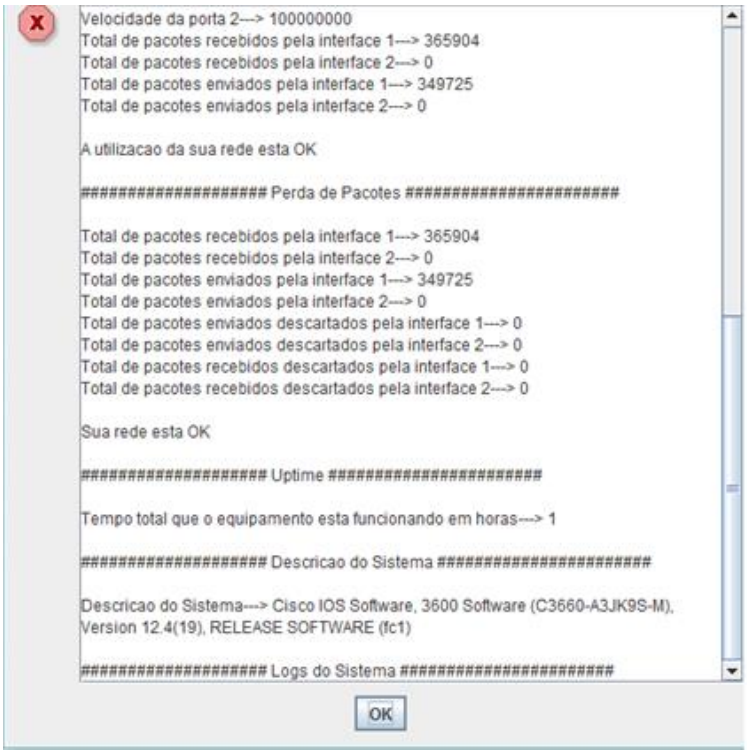
Após estes erros serem corrigidos, um novo teste for feito no roteador, e os resultados obtidos são representados pela figura 23 e 24.

Figura 23 – Resultado após problemas resolvidos



Própria fonte

Figura 24 – Resultado após problemas resolvidos



Própria fonte

Pode-se observar que o sistema funcionou como esperado já que este apresentou todos as falhas presentes no ambiente, e assim que as mesmas foram corrigidas, a informação de que não haviam mais falhas.

5. CONCLUSÃO

Observando o sistema em funcionamento e os resultados obtidos, pode-se perceber que a interface simples e intuitiva permite o fácil entendimento dos problemas, permitindo a rápida resolução dos mesmos.

A utilização deste programa em um ambiente empresarial ou até mesmo residencial, facilitaria a determinação dos problemas, e como já citado, da necessidade ou não de manutenção de outros *softwares*. Apesar de não corrigir os problemas, apresentar a exata causa de um mal funcionamento diminui consideravelmente o tempo gasto para sua resolução, o que também resulta na redução do capital gasto.

Além disso, com as próximas funções planejadas para o *software*, será possível monitorar continuamente a rede, permitindo um estudo do ambiente em conjunto com o gerenciador de sistemas, o que facilitara a determinação de mal funcionamentos ou gargalos de comunicação.

Como outro trabalho futuro têm-se a associação de um baco de dados com o sistema. Com isso, todas as informações obtidas serão armazenas de forma que será possível gerar gráficos e análises do ambiente de redes a longo prazo, indicando épocas de maior criticidade ou até mesmo padrões de falhas.

5.1 REFERÊNCIAS

ADAMI, David; MARCHESE, Mario; RONGA, Luca Simone. T**CP/IP based Multimedia Applications and Services over Satellite Links:** Experience of an ASI/CNIT Project**.** CNIT - National Inter-University Association for Telecommunications. Parma, Itália, vol. 8, n. 3, Junho de 2001, p. 20–27

CLARO, Daniela Barreiro, SOBRAL, João Bosco Mangueira. **Programação em Java**. Florianópolis/SC: Pearson Education, 2008. 89p.

DIAS, Beethovem Zanella; ALVES JÚNIOR, Nilton. Protocolo de Gerenciamento SNMP. Notas Técnicas do Trabalho de Pesquisa do Curso de Computação n. º 006/01. In.: **Centro Brasileiro de Pesquisas Físicas.** Rio de Janeiro, 2001, 16p. Disponível em: <http://www.rederio.br/downloads/pdf/nt00601.pdf>. Acesso em: 13 maio 2016, às 16h37min.

GUEDES, Gilleanes T. A. **UML 2: uma abordagem prática**. São Paulo: Novatec Editora, 2009.

HAN, Jiawei, KAMBER, Micheline, PEI, Jian. **Data Mining Concepts and Techniques.** Estados Unidos: Elsevier, 2012. 707p.

HELMKE, Matthew, GRANER, Amber. **The Official Ubuntu Book.** 7ª ed. Crawfordsville: Pearson, 2012. 336p.

IBM. **Netcool Network Management**. Disponível em: <http://www-03.ibm.com/software/products/pt/netcool-network-management >. Acesso em 6 de junho de 2016

KANTARDZIC, Mehmed**. DATA MINING Concepts, Models, Methods, and Algorithms**. Estados Unidos: John Wiley& Sons, 2011. 534p.

KUROSE, James, ROSS, Keith. **Computer Networking: A Top-Down Approach**, 6a edição. Estados Unidos: Pearson Education, 2013.

MARCHESE, Mario. **QoS over heterogeneous networks**. Inglaterra: John Wiley & Sons, 2007. 307p.

MAURO, Douglas R, SCHMIDT, Kevin J. **Essential SNMP.** 2ª ed. Estados Unidos: O`Reilly, 2005. 458p.

MENS, Tom, DEMEYER, Serge. **Software Evolution**. Estados Unidos: Springer Berlin Heidlberg, 2008. 341p.

MOQADI, Kanan Ali Abdulla. **Uso de ferramentas de gerência de rede para análise de desempenho de uma rede local**. 2011. 19 f. TG (Trabalho de graduação do Curso Superior em Tecnologia de Redes), Universidade Luterana do Brasil, Canoas, 2011. Disponível em: <http://www.ulbra.inf.br/joomla/images/documentos/TCCs/2011_02/PROJETO_RC_KANAN_ALI_ABDULLA_MOQADI.pdf>. Acesso em: 13 maio 2016, às 16h37min.

SAYDAM Tuncay, MAGENDANZ Thomas. **"From Networks and Network Management into Service and Service Management," Journal of Networks and System Management**, Vol. 4, No. 4, 1996. p. 345-348.

SOMMERVILLE, Ian. **Engenharia de Software**. 9ª edição. Brasil: Pearson Education, 2011. 529 p.

TANENBAUM, Andrew S; WETHERALL, Daviv J. **Computer Networks.** 5th. Edition. Boston, Massachussetts: Pearson Education, 2011. 905p.

THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS. **IEEE Standard for Software Maintenance**. United States of America: Institute of Electrical and Electronics Engineers, 1998. 47p. Disponível em: <http://www.cs.uah.edu/~rcoleman/CS499/CourseTopics/IEEE_Std_1219-1998.pdf>. Acesso em: 13 maio 2016, às 16h37min.

5.2 APÊNDICES

5.2.1 Gerando ambiente GNS3

1. Inicialmente é necessária a obtenção de um sistema operacional de roteador cisco (IOS) a fim de que o programa possa utilizá-lo virtualmente.
2. Configuração de máquina virtual com o sistema operacional Ubuntu e conectá-la a partir de uma conexão NAT com o roteador.
3. O roteador deve ser configurado seguindo as linhas de comando:
   - `snmp-server community (nome da comunidade) ro`
   - `snmp-server contact (nome do contato)`
   - `snmp-server location (nome da localidade)`
4. Posteriormente instale o pacote SNMP e as MIBs no Ubuntu a partir dos comandos:
   - `apt-get install snmp`
   - `apt-get install snmp-mibs-downloader`
   - `download-mibs`
5. Finalmente, todas as informações do roteador (agente) devem ser requisitadas pelo computador (gerenciador) a partir do comando:
   - `snmpwalk -v2 -c (nome da comunidade) (endereço IP do roteador)`
6. Desta forma todas as informações do roteador serão requisitadas pela máquina virtual. O resultado será este:

```
SNMPv2-MIB::sysDescr.0 = STRING: Cisco IOS Software, 3600
Software (C3660-A3JK9S-M), Version 12.4(19), RELEASE SOFTWARE
(fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2008 by Cisco Systems, Inc.
Compiled Fri 29-Feb-08 23:47 by prod_rel_team
SNMPv2-MIB::sysObjectID.0 = OID: SNMPv2-
SMI::enterprises.9.1.341
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (607655)
1:41:16.55
```

```
SNMPv2-MIB::sysContact.0 = STRING: Vegeta
SNMPv2-MIB::sysName.0 = STRING: R1
SNMPv2-MIB::sysLocation.0 = STRING: Planet Vegeta
SNMPv2-MIB::sysServices.0 = INTEGER: 78
SNMPv2-MIB::sysORLastChange.0 = Timeticks: (0) 0:00:00.00
IF-MIB::ifNumber.0 = INTEGER: 3
IF-MIB::ifIndex.1 = INTEGER: 1
IF-MIB::ifIndex.2 = INTEGER: 2
IF-MIB::ifIndex.4 = INTEGER: 4
IF-MIB::ifDescr.1 = STRING: FastEthernet0/0
IF-MIB::ifDescr.2 = STRING: FastEthernet0/1
IF-MIB::ifDescr.4 = STRING: Null0
IF-MIB::ifType.1 = INTEGER: ethernetCsmacd(6)
IF-MIB::ifType.2 = INTEGER: ethernetCsmacd(6)
IF-MIB::ifType.4 = INTEGER: other(1)
IF-MIB::ifMtu.1 = INTEGER: 1500
IF-MIB::ifMtu.2 = INTEGER: 1500
IF-MIB::ifMtu.4 = INTEGER: 1500
IF-MIB::ifSpeed.1 = Gauge32: 100000000
IF-MIB::ifSpeed.2 = Gauge32: 100000000
IF-MIB::ifSpeed.4 = Gauge32: 4294967295
IF-MIB::ifPhysAddress.1 = STRING: cc:1:2:8c:0:0
IF-MIB::ifPhysAddress.2 = STRING: cc:1:2:8c:0:1
IF-MIB::ifPhysAddress.4 = STRING:
IF-MIB::ifAdminStatus.1 = INTEGER: up(1)
IF-MIB::ifAdminStatus.2 = INTEGER: down(2)
IF-MIB::ifAdminStatus.4 = INTEGER: up(1)
IF-MIB::ifOperStatus.1 = INTEGER: up(1)
IF-MIB::ifOperStatus.2 = INTEGER: down(2)
IF-MIB::ifOperStatus.4 = INTEGER: up(1)
IF-MIB::ifLastChange.1 = Timeticks: (447) 0:00:04.47
IF-MIB::ifLastChange.2 = Timeticks: (447) 0:00:04.47
IF-MIB::ifLastChange.4 = Timeticks: (0) 0:00:00.00
IF-MIB::ifInOctets.1 = Counter32: 365904
IF-MIB::ifInOctets.2 = Counter32: 0
IF-MIB::ifInOctets.4 = Counter32: 0
IF-MIB::ifInUcastPkts.1 = Counter32: 2820
IF-MIB::ifInUcastPkts.2 = Counter32: 0
IF-MIB::ifInUcastPkts.4 = Counter32: 0
IF-MIB::ifInNUcastPkts.1 = Counter32: 565
IF-MIB::ifInNUcastPkts.2 = Counter32: 0
IF-MIB::ifInNUcastPkts.4 = Counter32: 0
IF-MIB::ifInDiscards.1 = Counter32: 0
IF-MIB::ifInDiscards.2 = Counter32: 0
IF-MIB::ifInDiscards.4 = Counter32: 0
IF-MIB::ifInErrors.1 = Counter32: 0
IF-MIB::ifInErrors.2 = Counter32: 0
IF-MIB::ifInErrors.4 = Counter32: 0
IF-MIB::ifInUnknownProtos.1 = Counter32: 0
IF-MIB::ifInUnknownProtos.2 = Counter32: 0
IF-MIB::ifInUnknownProtos.4 = Counter32: 0
IF-MIB::ifOutOctets.1 = Counter32: 349725
```

```
IF-MIB::ifOutOctets.2 = Counter32: 0
IF-MIB::ifOutOctets.4 = Counter32: 0
IF-MIB::ifOutUcastPkts.1 = Counter32: 3428
IF-MIB::ifOutUcastPkts.2 = Counter32: 0
IF-MIB::ifOutUcastPkts.4 = Counter32: 0
IF-MIB::ifOutNUcastPkts.1 = Counter32: 118
```