FACULDADES INTEGRADAS DE TAQUARA

CURSO DE SISTEMAS DE INFORMAÇÃO

SISTEMA PARA AVALIAR SITES WEB, NA VISÃO DO USUÁRIO UTILIZANDO QOE

TIAGO SOUZA DE SOUZA

Taquara

2010

TIAGO SOUZA DE SOUZA

SISTEMA PARA AVALIAR SITES WEB, NA VISÃO DO USUÁRIO UTILIZANDO QOE

Trabalho de conclusão de curso apresentado ao Curso de Sistemas de Informação das Faculdades Integradas de Taquara, sob orientação do Professor M. Giovani Facchini.

Taquara

2010

AGRADECIMENTOS

Em primeiro lugar agradeço a Deus, por me guiar e estar comigo sempre, me dando saúde e tudo mais que um ser humano precisa para sobreviver. Aos meus pais, José Antonio e Ana Souza por me darem incentivo para continuar nesta caminhada. A minha namorada Greice pela compreensão de inúmeras vezes que estive ausente realizando este trabalho.

Agradeço aos colegas e professores pela troca de informações e transmissão de conhecimento, por causa disso faz do curso de Sistemas de Informação da FACCAT uma das melhores do Brasil. E por fim gostaria de agradecer ao professor Giovani Facchini pelo esforço realizado na orientação deste trabalho de conclusão.

RESUMO

Hoje em dia com o avanço da *internet* e a ploriferação de sistemas on-line, tanto para compra e venda como para gerenciamento de empresas, surge a necessidade de que seus sistemas estejam no seu pleno funcionamento para que não haja frustrações dos usuários, acarretando em uma possível má imagem para a empresa, desenvolveu-se a idéia de monitoração remota de software.

O objetivo deste trabalho é criar uma ferramenta para identificar e monitorar os casos em que os sistemas falham em suas funcionalidades ou não atingem o SLA acordado.

Palavras-chave: SLA. QOE. *Web Service*.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE SIGRAS

TI – Tecnologia da informação

API – Application Programming Interface

PC – Personal Computer

SMS – Short Message Service

HTTP – Hypertext Transfer Protocol

SUMÁRIO

1 INTRODUÇÃO

No Brasil o número de pessoas que utilizam a *internet* vem crescendo consideravelmente a cada ano, de 2006 até 2009 o número de internautas subiu de 32,5 milhões para 66,3. Nom período entre 2000 e 2007 a quantidade de *hosts* subiu de 877 para 17787. [TELECO, 2010]

Hoje um dos serviços que mais se plorifera é o comercio eletrônico. Em 2009 o Brasil movimentou certa de R$ 10.060.000,00 bilhões como consta na figura 1 (eCommerceOrg, 2008)

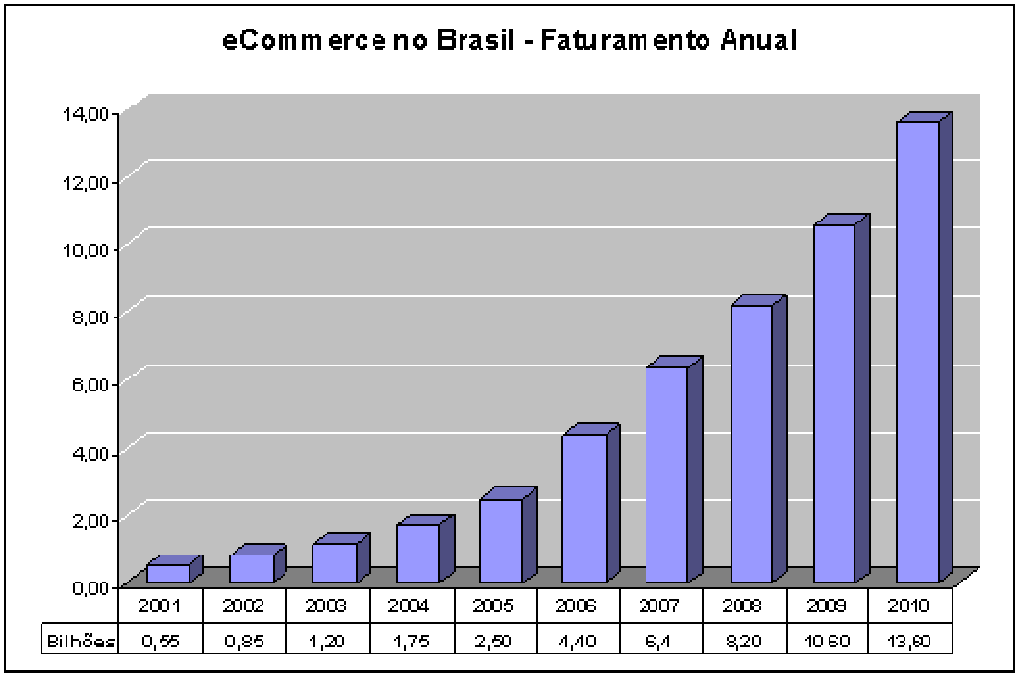


Figura 1 – Evolução do faturamento brasileiro em e-commerce

Deste faturamento se da grande parte a produtos como: livros, revistas, informática, eletrônicos, eletrodomésticos, saúde e beleza. [EBIT 2010]

Segundo Zuffo (2003) A Sociedade e a Economia no novo Milênio (2003, p. 1)

> A infra-estrutura de servidores deve atender uma série de serviços fundamentais, alem do próprio comercio eletrônico e da permuta eletrônica de dados (*Eletronic Data Interchanges, EDI*), destacando se Teia, Correio, Diretório, Apresentação, Barreira de Proteção, Gerenciamento, Relatórios e Cobrança, Comunidade e Conversas, Acesso Remoto, Noticiário e Informações, Aplicações, Arquivos e Base de Dados, Autenticação, Desenvolvimentos e Estágios Probatórios e, evidentemente, Duplicatas e Documentos (back-up) e Sistema de Recuperação de Eventos Desastrosos.

Ainda Zuffo (2003) diz mais, fatores básicos como Robustez, escalabilidade e

a facilidade de acesso são primordiais para criar condições de aceitação em sistemas de negócios eletrônicos, por consequência garantindo a preferência e a fidelidade do cliente. Mas para isso se concretizar a empresa deve dispor de uma infra-estrutura eficiente de distribuição de produtos, atendimento ao público externo e de suporte financeiro às operações de vendas.

No que diz respeito a prestação de serviços, existes normas definidas antes mesmo da aquisição destes. Como por exemplo os SLAs.

Conforme Albertin e Sanches, Outsourcing de TI (2008, p. 120):

> Um *Service Level Agreement* (SLA) é um contrato que define parâmetros de negócios e/ou de suporte técnico que um provedor de serviço fornecerá a seu cliente, especificando medidas de performance e conseqüências por não atingimento ou falhas.

O SLA é um acordo entre a contratante e a contratada. Estes acordos estabelecem limites de suas responsabilidades que são organizados em níveis.

1.1 Justificativa

Hoje na *internet* se encontram inúmeros sites que realizam vendas. Esta prática já vem sendo realizada faz algum tempo, e se denomina *e-commerse*. O objetivo do comercio eletrônico é facilitar a venda de produtos e serviços tendo um custo reduzido, pois não possui estrutura física. Uma das vantagens desse modelo de lojas eletrônicas em relação a loja comum é a capacidade de poder ser acessada a qualquer momento. (E-COMMERCE, 2008)

Mas com todos estes benefícios, alguns sites podem estar inoperantes ou algum de seus módulos apresentarem problemas. Já houve casos de pessoas que tentaram comprar produtos via *internet* e quando estavam perto de concluir a compra, o site acusou erro e não concluiu a venda, mas o valor do produto veio descontado no cartão de credito. Casos como este são comuns, mas para empresas que dependem exclusivamente de vendas *online* isto acaba se tornando um ponto negativo, pois o transtorno gerado pode fazer com que esse cliente em potencial não volte mais para a loja.

Para evitar casos como o citado acima, uma alternativa para esse problema seria ter uma ferramenta que percorresse constantemente alguns passos que o

cliente realiza no ato de compra, verificando se as funcionalidade estão desempenhando o seu papel corretamente.

1.2 Objetivos

1.2.1 Objetivos gerais

Criar um software capaz de monitorar aplicações *web* e realizar testes em um site, para verificar seu funcionamento, como por exemplo: se *links* estão disponíveis (retornando), se os formulários estão funcionando e retornando o resultado esperado. O sistema a ser desenvolvido será instalado fora do servidor onde a aplicação a ser testada estará e seus resultados como por exemplo, verificação de resultados de log, poderão ser vistos por uma página *web* própria.

1.2.2 Objetivos específicos

a) produzir e gerenciar logs de sistema para analise futura;
b) simular um acesso a um determinado link ou preenchimento de formulário de algum site;
c) identificar o tempo de resposta do site;
d) visualizar resultados de algum período em forma de gráfico;
e) disponibilizar interface *web* para análise remota;
f) visualizar os comparativos na forma de gráficos;

1.3 Organização do trabalho

O presente trabalho está organizado em seis seções, sendo elas:

a) a seção 1 apresentou uma introdução, justificativa e objetivo trabalho;
b) na seção 2 é realizada a fundamentação teórica dos principais conceitos envolvidos;
c) a seção 3 contempla a metodologia utilizada no desenvolvimento;
d) na seção 4 são explicadas as tecnologias empregadas;
e) a seção 5 apresenta os resultados obtidos com o sistema desenvolvido;
f) a seção 6 expõe as conclusões do trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

O objetivo deste software é atuar como um monitor de algum sistema *web*, todo este monitoramento será feito de forma remota. Sendo assim serão utilizadas algumas tecnologias, ferramentas e conceitos que serão descritas. São elas:

a) Qualidade da informação;
b) QoE – *Quality of Experience*;
c) SLA – *Service Level Agreement*;
d) PHP – Hypertext Preprocessor
e) Crontab
f) Protocolos *web*

2.1 Qualidade da Informação

Quando um usuário depara-se pela primeira vez com um site, ele tende a observar se o site é fácil de usar, se a informação tem qualidade e também se a autoria da informação é de confiança. Fazendo esta análise o usuário será capaz de ter uma primeira impressão da página *web*. (Ana Amélia, 2003)

Na literatura acadêmica há um consenso geral de que a qualidade da informação é composta por uma gama de atributos e dimensões, mas ao mesmo tempo não há uma concordância de quais as dimensões que definem a qualidade da informação (TAYI e BALLOU, 1998)

Tais dimensões são encontradas nas várias definições, são elas: acurácia, relevância e facilidade de acesso. Dimensões menos freqüentes são mencionadas por outros autores, entre as quais são: custo beneficio, e admissibilidade, que tem por objetivo avaliar se o acesso, armazenamento e o uso da informação são admitidos por fatores culturais, éticos, morais ou legais. (CANHETTE, 2004)

2.2 Qualidade de Experiência do Usuário (QOE)

A Qualidade de Experiência do Usuário é uma medida, um parâmetro que representa o desempenho do produto, através da percepção dos usuários sobre tal serviço prestado. (REVISTA ENGENHARIA, 2009).

A Revista Engenharia ainda segue falando que QoE pode ser confundido com outra medida de parâmetro chamada de QoS (Quality of Service) que neste caso designa a capacidade de fornecer um serviço, conforme as exigências de matéria e tempo de resposta e de banda.

2.3 SLA (Service Level Agreement)

O Service Level Agreement (SLA) é um documento que será negociado entre as partes para uma eventual contratação de serviços para TI. Ele é colocado geralmente como anexo do contrato e tem por objetivo especificar os requisitos mínimos aceitáveis para o serviço proposto. O não cumprimento do SLA implica em penalidades, estipuladas no contrato para o provedor de serviço (TELECO, 2003)

Um SLA abrange itens como qualidade do serviço, critérios de cobrança, realizar previsões, processo de atendimento e relatórios fornecidos ao cliente.

O acordo de nível de serviços também é um instrumento para a gestão das expectativas do cliente. Possui o objetivo de definir uma estrutura para a gestão da qualidade e quantidade dos serviços entregues e após atender a demanda dos clientes a partir de um entendimento claro do conjunto de compromissos. Esse instrumento serve como uma ferramenta de comunicação e prevenção de conflitos, tornando-se um documento vivo, sendo base para garantir que ambas as partes usarão os mesmos critérios para avaliar a qualidade do serviço. (GOVERDO DO ESTADO DO CEARARÁ, 2010)

Segundo (STURM, 2000) uma gestão de serviços devem obedecer a seis itens:

- Satisfação do cliente: Prioridade na disponibilidade de uma aplicação em relação a outra, são serviços que o cliente percebe como um bom serviço;
- Gestão das expectativas: Em algum momento os serviços estão sendo prestados de maneira eficiente, e os clientes sentem atendidos, mas nada impede trazer novidades. Por isto mais aplicações devem ser gerenciadas e novas tecnologias devem ser incorporadas;
- Marketing interno dos serviços de TI: Muitas vezes os clientes de TI não vêem esta área muito necessária. Mas quando a TI mostra que possui bons serviços e faz com que os negócios prosperem, os clientes constatam que ela faz bem para a empresa;
- Controle de custos: Um SLA em primeiro momento deve esclarecer quais as áreas os níveis de serviços estão satisfatórios, para que os recursos possam ser direcionados de forma eficiente;
- Estabelecimento de uma estratégia defensiva: Com SLAs precisos entre TI e os clientes é possível estabelecer um conjunto de medições (monitoramento) que é capaz de mostrar ao cliente a qualidade do serviço que é prestado.

Segundo (GOMES 2000) os SLAs estão deixando de ser uma ferramenta financeira para se tornar um instrumento para a gestão das expectativas do cliente, uma vez que criam um entendimento sobre os serviços oferecidos, prioridades e responsabilidades, e especificam os parâmetros de TI requeridos para atender os objetivos do negócio.

(GOMES 2000) Ainda relata que nos acordos de serviços ainda há muitas omissões, dificuldades e falhas, como:

- Especificação de resultados ao invés de especificação do esforço: Um Ato falho é dizer que “o serviço de TI X irá prover sua companhia com o resultado Y”, um SLA deve especificar o esforço a ser feito para solucionar problemas, tal como “nós estaremos no local L dentro de um período de tempo X, no caso de seus sistemas caírem”. Com isto, a responsabilidade continua sendo do cliente ao invés de passar a ser do fornecedor de serviços de TI;
- Especificação não clara do serviço: Em muitos casos não ficam claros qual aspecto do serviço será mensurado.

- Especificação incompleta do serviço: Na maioria dos casos a descrição completa e detalhada dos requisitos a respeito dos serviços só são possíveis depois que o sistema foi desenvolvido.
- Gestão insuficiente de custos: Para um conjunto de serviços em TI por exemplo o preço desses serviços são fixos para um conjunto de serviços. Por causa disto determinar uma otimização do preço aliado com o desempenho de cada cliente se torna difícil mensurar por causa do pouco discernimento nos custos dos serviços de TI individuais.
- Documentos do SLA como um "peso-morto": Um SLA afeta todos os usuários dos seus serviços, mas é um documento que pode ser compreendido por uma pequena parcela das pessoas voltadas para TI;

Os relatórios de níveis de serviço são um veículo de comunicação importante entre a área de TI, a comunidade usuária e as linhas de negócio. O público alvo e os detalhes das informações contidas nos relatórios a serem providos pela área de TI devem determinar a sua freqüência de geração: diariamente, semanalmente, mensalmente e/ou trimestralmente. Relatórios efetivos permitem endereçar pró-ativamente dificuldades do serviço e reduzir o efeito negativo da reputação do departamento de TI como resultado de serviços indisponíveis ou da degradação de um serviço. Também podem reduzir a quantidade de chamadas ao *help desk*[1], quando usuários buscam conhecimento acerca de serviços de TI comprometidos. (Sturm 2000)

2.4 Hypertext Preprocessor - PHP

O PHP é a sucessão de um produto mais antigo, chamado PHP/FI. Criado por Rasmus Lerdorf em 1995, com intenção de ser apenas um *script* em Perl, que foi nomeada como *Personal Home Page Tools*. Com o passar do tempo mais funcionalidades foram adicionadas. Mais tarde Rasmus disponibilizou o código fonte do PHP/FI para que outras pessoas pudessem ver e melhorar o código. Em 1997 foi lançada a versão 2.0 do PHP/FI, que obteve milhares de usuários ao redor do

[1] Help Desk = Designa o serviço de apoio a usuários para suporte e resolução de problemas técnicos.

mundo. A segunda versão do software não durou muito, pois no mesmo ano Andai Gutmans e Zeev Suraski desenvolveram o PHP na versão 3.0 (lançada oficialmente em 1998), uma grande vantagem desta versão era a capacidade de extensabilidade, além de oferecer estrutura para diversos bancos de dados, protocolos e APIs[2], o PHP 3.0 atraiu dezenas de desenvolvedores para submeter novos módulos.

Por volta de 2000 é lançada a versão do 4.0 do PHP, que incluía características como suportes a muitos servidores *web*[3] , sessões HTTP, buffer de saída, maneiras mais seguras de manipular input de usuários e muitas construções novas na linguagem.

Após um longo desenvolvimento, em julho de 2004, foi lançado o PHP 5, incluindo novos módulos como por exemplo o core e a Zend Engine 2.0 com um novo modelo de orientação a objeto. Hoje o PHP esta instalado em milhões de sites no mundo inteiro, e esta com 20% dos domínios da *internet*. (PHP 2009)

O PHP e o HTML possuem muita interação pois o PHP pode gerar HTML, e o HTML pode passar informação para o PHP. A imagem abaixo demonstra como a linguagem PHP é incluída dentro do código HTML. A figura 2 ilustra um código escrito em PHP irá imprimir na tela a seguinte frase: Olá, Eu sou um script PHP! As tags “<? php” e “?>” indicam o começo e o termino do código PHP.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
    <head>
        <title>Exemplo</title>
    </head>
    <body>

        <?php
        echo "Olá, Eu sou um script PHP!";
        ?>

    </body>
</html>
```

Figura 2 – Exemplo de código php

[2] API = É um conjunto de rotinas e padrões estabelecidos por um software
[3] Servidor web = Um programa de computador responsável por aceitar pedidos HTTP de clientes, geralmente os navegadores, e servi-los com respostas HTTP

2.4.1 cURL

Criada para mover arquivos entre terminais utilizando diferentes protocolos como por exemplo FTP, HTTP e o SPC, o cURL no inicio era pra ser um utilitário de linha de comando, mas sua aceitação foi tão grande que foi criada uma biblioteca para integrar o comportamento dos aplicativos.

O trecho de código abaixo nos mostra como o cURL realiza estas transações de arquivos de um web site.

```
$ curl -T test.html ftp://user:password@ftp.exampledomain.com/ftpdir/
  % Total    % Received % Xferd  Average Speed    Time    Time     Time    Current
                                 Dload  Upload    Total   Spent    Left    Speed
100 43320    0     0  100 43320      0  38946   0:00:01 0:00:01  --:--:--    124k
$
```

Figura 3 – Exemplo cURL realizando um upload de arquivo em site FTP

Como podemos observar é passado o nome do arquivo a ser baixado e em seguida a url do mesmo, logo após a execução da linha, é colocado alguns detalhes do download como por exemplo: tempo total do download e taxa de transferência.

A partir daí é possível obter detalhes de uma transferência *web* através de uma linha de comando. (IBM, 2010)

2.4.1.1 cURL no PHP

Esta seção descreve mais especificadamente, algumas funções do cURL com o PHP que foi desenvolvido no sistema proposto.

A primeira coisa a fazer é chamar uma função curl_unit(), segue exemplo.

```
1  <?php
2  $ch = curl_init();
3  ?>
```

Figura 4 – Exemplo iniciando função cURL no PHP

Segundo IMASTERS 2006, exemplo a seguir demonstra um dos códigos de maior utilização no sistema proposta, com ele é possível ao executar um código em PHP, saber se uma página *web* existe ou não, retirando a necessidade do acesso a ela via forma normal (entrar com endereço da página no navegador).

```
1  <?php
2  $curl = curl_init();
3
4  curl_setopt($curl, CURLOPT_URL, "http://www.paginainexistente.com");
5  curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
6
7
8  $saida = curl_exec($curl);
9
10 $resposta = curl_getinfo($curl, CURLINFO_HTTP_CODE);
11
12 if ($resposta == '404') {
13         echo 'Esta página não existe';
14 } else {
15         echo $saida;
16 }
17 ?>
```

Figura 5 – Exemplo código em PHP acessando um endereço *web*

Na linha 4 da figura 5, é informado a URL e outras funções ao cURL, a função CURLOPT_RETURNTRANSFER está passando o conteúdo do manipulador em forma de string. Na linha 8 a URL será acessada e na linha 10 a variável $resposta recebe o retorno da página acessada.

Com a resposta do retorno, a variável $resposta é comparada com código http 404 na linha 12, onde significa página inexistente, que neste caso será verdadeira, e então seria impresso na tela “Esta página não existe”.

A cURL juntamente com o PHP abre inúmeras possibilidades para trabalhar com obtenção de conteúdos de outros sites, este exemplo citado acima, é apenas uma forma e explicar as inúmeras maneiras de como usar esta ferramenta. (IMASTERS, 2006)

2.4.2 Libcurl

Criado por Daniel Stenberg no segundo semestre de 1997 a libcurl vem se atualizando e integrando mais funções até hoje, mais foi no ano de 2000 que os desenvolvedores do PHP integraram na versão 4.0.2 suporte a libcurl. (CURL, 2010)

A libcurl é uma biblioteca que proporciona realizar a comunicação de vários tipos de servidores com uma gama variada de protocolos como, por exemplo: http, https, ftp, gopher, telnet, dict, file e ldap, e também à suporte para certificados são eles: SSL, HTTP POST, HTTP PUT. Esta biblioteca também tem funções para serviços de upload com ftp, baseados em formulário http, proxies, cookies e autenticação. Ela esta presente na linguagem PHP desde a versão 4.0.2 do PHP (PHP 2010)

Uma das suas principais característica é a forma de trabalhar de modo idêntico com várias plataformas, entre as quais podemos destacar: Solaris, NetBSD, FreeBSD, OpenBSD, Darwin, Linux, Windows, Mac OS X, Novell NetWare, Dos, entre outros. (cURL, 2010)

Por ser considerada um software livre e várias empresas declararam que à utilizam em seus sistemas, como por exemplo a Adobe, Apple, Cisco, Google, IBM, Motorola, Sony, Sun, Yahoo.

2.4.3 LibChart

LibChart é uma ferramenta para geração de gráficos para a linguagem PHP. Atualmente ela esta na versão 1.2.2 e não à necessidade de adquirir licença. (LIBCHART, 2010)

Esta versão possui conjunto de dados para gráficos em Barras (horizontais e verticais), Linhas e Pizza. A Figura 6 nos mostra um exemplo de gráfico de barras.

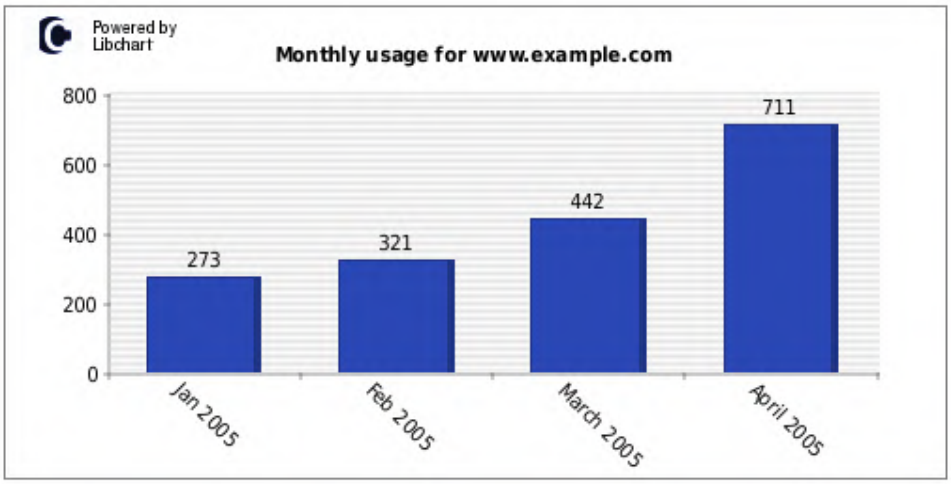


Figura 6 – Exemplo Gráfico Gerado Libchart

A montagem do gráfico é gerada através de uma biblioteca instalada no diretório, logo no código PHP deverá ser informado o caminho, ou seja, referenciando a biblioteca. A Figura 7 linha 2 ilustra isso.

```
1 <?php
2  include "libchart/classes/libchart.php"; incluem "libchart / classes / libchart.php";
3  header("Content-type: image/png"); header ("Content-type: image / png");
```

Figura 7 – Chamando a biblioteca libchart

2.5 Crontab

É indicado quando surge a possibilidade agendar tarefas em uma determinada data ou algum horário especifico

Para Julio Cezar Neves (p.70, 2008):

> Crontab é o programa para instalar, desinstalar ou listar as tabelas usadas pelo programa (*daemon*) cron para que possamos agendar a execução de tarefas administrativas.

O programa cron é o que mantém o agendamento na tabela, pois é ele quem edita, cria uma nova entrada ou remove outra.

O crontab possui algumas funções, listadas na figura a seguir.

Opção	Função
-r	Remove o crontab do usuário
-l	Exibe o conteúdo do crontab do usuário
-e	Edita o crontab atual do usuário

Figura 8 – Funções Cron

As tabelas crontab possuem seis campos separados por espaços em branco, a Figura 9 nos mostra com mais clareza.

```
#M                                  S
#i                                  e
#n      H                           m
#u      o       D       M           a
#t      r       i       e           n
#o      a       a       s           a       Programa
#=======================================
0       0       *       *           *       backup.sh
30      2       *       *           0       bkpsemana.sh
0,30    *       *       *           *       verifica.sh
0       1       30      *           *       limpafs.sh
30      23      31      12          *       encerraano.sh
```

Figura 9 – Estrutura da tabela crontab

O símbolo (#) indica que a partir daquele ponto até a linha seguinte é um comentário.

1ª Linha – Todo dia às 00:00 h execute o programa backup.sh;

2ª Linha – Às 02:30 h de todo o Domingo execute o programa bkpsemana.sh

3ª Linha – Todas as horas exatas e meias horas execute o programa verifica.sh;

4ª Linha – Todos os dias 30 (de todos os meses) à 01:00 h execute o programa limpafs.sh;

5ª Linha – No dia 31/12 às 23:30 h execute o programa encerraano.sh.

Como podemos ver, cada linha do crontab corresponde a um comando a ser executado em uma data definida, sempre ao final da linha, está o arquivo a ser executado.

2.6 Protocolos de Internet

No inicio os protocolos de internet foram desenvolvidos para suportar aplicativos remotos simples, como por exemplo, transferência de arquivos e correio eletrônico, envolvendo uma comunicação com latência muito alta entre computadores, mas apesar disso mostrou-se uma eficiência muito grande em redes de computadores locais e a longa distância. (George Coulouris, 2005)

Mas este trabalho destina-se a se aprofundar em um protocolo mais especifico que é o HTTP.

2.6.1 HTTP – Hypertext Transfer Protocol

O HTTP é um protocolo de aplicação responsável pelo tratamento de pedidos e respostas entre clientes da *internet*. Atualmente o Hypertext Transfer Protocol encontra-se na versão 1.1. (WIKIPEDIA, 2010)

Wikipedia 2010 segue afirmando que o HTTP utiliza o modelo cliente-servidor, como na maioria dos protocolos de rede, baseando-se no conceito de requisição e resposta.

As comunicações entre clientes de servidor são feitas através de mensagens. O cliente envia uma mensagem de requisição e o servidor envia uma resposta ao cliente. (George Coulouris, 2005)

Uma seção HTTP consiste basicamente em um comando GET que especifica um URL, seguido por um numero de linhas cuja a forma que ela é apresentada faz lembrar cabeçalhos de mensagens de correio eletrônico.

As respostas de retorno contém códigos, tais códigos podem ser da seguinte forma:

- 1xx (Informação) – informa para o cliente de que sua requisição foi recebida e esta sendo processada;
- 2xx (Sucesso) – Informa que a requisição foi bem sucedida;
- 3xx (Redirecionamento) – Informa a ação adicional que deve ser tomada para completar a requisição;

- 4xx (Erro no cliente) – Avisa que o cliente fez uma requisição e não pode ser atendida;
- 5xx (Erro Servidor) – Ocorreu erro no servidor ao cumprir uma requisição válida.

2.6.1.1 Códigos de respostas HTTP

Os códigos de respostas mais comuns são:

- 200 OK – Informa que a requisição teve sucesso;
- 301 Moved Permanently – O recurso foi permanentemente movido para outro local;
- 302 FOUND – Descreve um redirecionamento temporário;
- 304 Not Modificad – Informa ao cliente que o recurso não foi modificado desde a ultima requisição;
- 401 Unauthorized – O recurso pode ser acessado caso haja autorização;
- 404 Not Found – Informa ao cliente que o recurso não foi encontrado no servidor. (Bruno Torres, 2010)

3 METODOLOGIA

Nesta seção, serão descritos os passos desde a analise até a faze de teste do software.

3.1 Descrição da Ferramenta

O *software* de monitoramento propõe-se auxiliar o usuário a saber com algum detalhamento dados como por exemplo se o servidor está respondendo alguma requisição e em quanto tempo é respondida a requisição.

Esta ferramenta é denominada MonitorE, um monitor que será programado para testar softwares *web* por um tempo pré-definido, gerando e armazenando as respostas das solicitações.

Tais respostas são armazenadas para análise posteriores em forma de log.

Os logs são compostos pelos os seguintes dados:

a) Data;

b) Hora;

c) Detalhes de conexão.

Esses logs podem ser analisados apenas abrindo o arquivo que o software atualiza automaticamente, onde contém o histórico de testes realizando até o momento. Neste caso o usuário deverá estar onde a aplicação de monitoramento esta instalada.

Uma segunda opção para acesso aos logs é através da página *web* que possui acesso restrito para usuários devidamente cadastrados, onde conterá os arquivos e será possível refinar os dados e também obter uma visualização em forma de gráfico. Detalhes mais específicos são tratados a seguir.

3.2 Desenvolvimento

No desenvolvimento de programas, existem vários métodos para representar e resolver um problema. Podendo ser representado em variados modelos de algoritmos, causando uma eficiência variada (Koscianski e Soares 2007).

Para o desenvolvimento do software foi utilizado o processo sequencial. Ao decorrer das seções apresenta-se as tarefas de cada etapa.

3.2.1 Analise

Para Pressman (2002, p. 266), “[...] resultam na especificação das características operacionais do software, indicam a *interface* do software com outros elementos do sistema e estabelecem restrições que o software deve satisfazer”.

A seguir discute-se detalhes de cada artefato, que são eles: Casos de Uso descritivo e o Diagrama de Casos de Uso.

3.2.1.1 Caso de Uso Descritivo

Para Larman (2005) casos de uso são artefatos textuais redigidos. A UML define um diagrama de caso de uso para ilustrar os casos de uso e seus atores.

Larman (2005) ainda nos diz que o caso de uso descritivo não faz uma descrição interna do sistema e tão pouco seus componentes e projetos. Ao invés disso descreve as responsabilidades do sistema, pois cada elemento do sistema possui a sua, e colaboram com outros elementos que também possuem responsabilidades.

```
Uc001 – Caso de uso Entrada de Dados

Descrição: Permite a entrada de dados pelo usuário, através da
tela.

Atores: Usuário, Sistema.

Escopo: Receber, listar dados.

Cenários principais:

     1. O usuário define qual arquivo de log listar.
     Caso o usuário opte por visualizar gráficos.
     2. Especifica o intervalo entre as datas que será
listada.
     3. Especifica a data de intervalo.
     4. Encerra o processo. Remete ao cenário 1.
```

Quadro 1: Caso de uso descritivo do processo listar datas

```
Uc002 – Caso de uso Grava Logs

Descrição: Permite a execução de arquivos e logo após gravar
logs.

Atores: Sistema.

Escopo: Executar Arquivos, Gravar dados.

Cenários principais:

     1. Cron executa arquivo.
     2. As informações coletadas são armazenadas em arquivos
de logs.
     4. Encerra o processo. Remete ao cenário 1.
```

Quadro 2: Caso de uso descritivo do processo listar datas

3.2.1.2 Diagrama de Caso de Uso

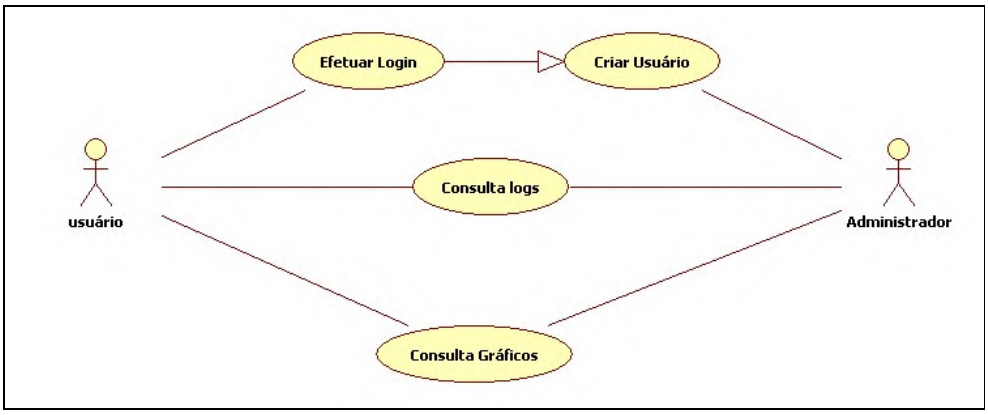
Larman (2005, p. 115):

> A UML fornece a notação de diagramas de casos de uso para ilustrar os nomes dos casos de uso e dos atores, bem como os relacionamentos entre eles.

O autor ainda relata que os diagramas de caso de uso e os seus relacionamentos são secundários na definição dos casos de uso, ou seja fazer a

definição dos casos de usos significa especifica-los em forma textual.

A imagem do quadro 2, nos mostra dois atores, onde o administrador será a pessoa que realizará os cadastros de usuários, esta é a única diferença entre o administrador e o usuário do sistema.



Quadro 3: Diagrama de caso de uso

3.2.2 Projeto

Segundo Eric Braude (2004, p. 28):

> O “projeto” de uma aplicação expressa como a aplicação deve ser construída. Ele descreve as partes envolvidas e como elas devem ser montadas. Um projeto consiste em um conjunto de documentos: em geral, esses conjuntos são diagramas com explicações com o significado desses diagramas.

Eric Braude, ainda segue dizendo, que um projeto é produzido a partir de requisitos, deixando de lado o código, uma forma muito útil para documentar um projeto é a Linguagem Unificada de Modelagem (UML).

Para Craig Larman (2005) a linguagem UML para documentar artefatos dos sistemas usa uma forma visual, usando uma notação diagramática padrão para representar figuras.

Com o intuito de mostrar as funcionalidades do sistema onde serão mostrados os diagramas UML, que é uma linguagem gráfica. Nas próximas seções são esclarecidas as especificações das funcionalidades do sistema, através de:

a) Casos de uso;

- diagramas de casos de uso;
- diagramas de atividade;

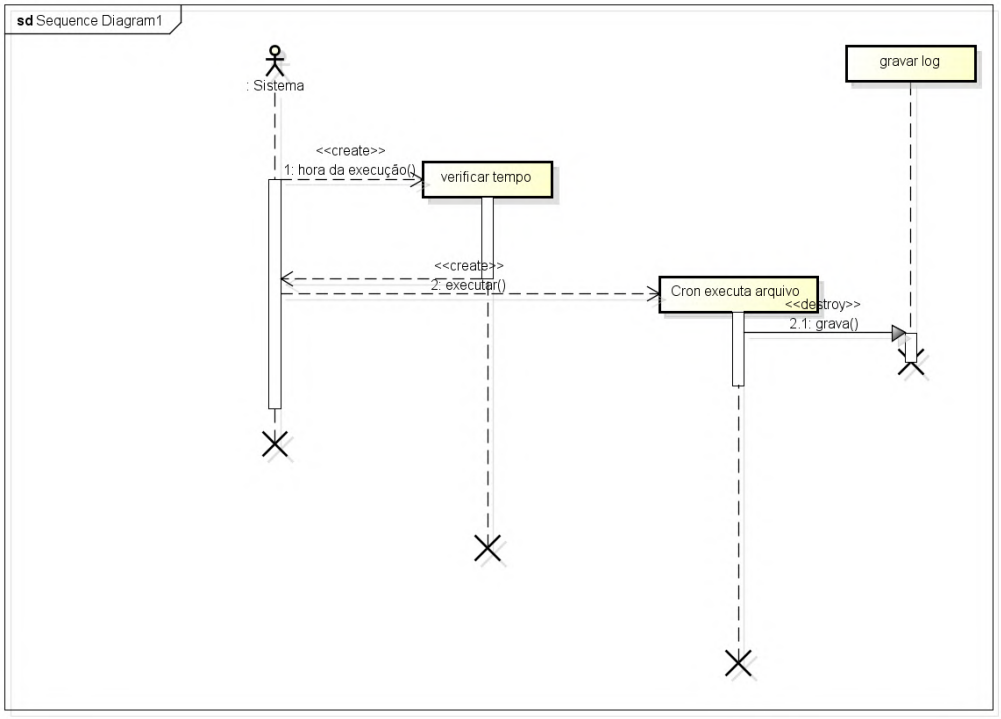
b) Diagrama de Sequência

3.2.2.1 Diagrama de sequência

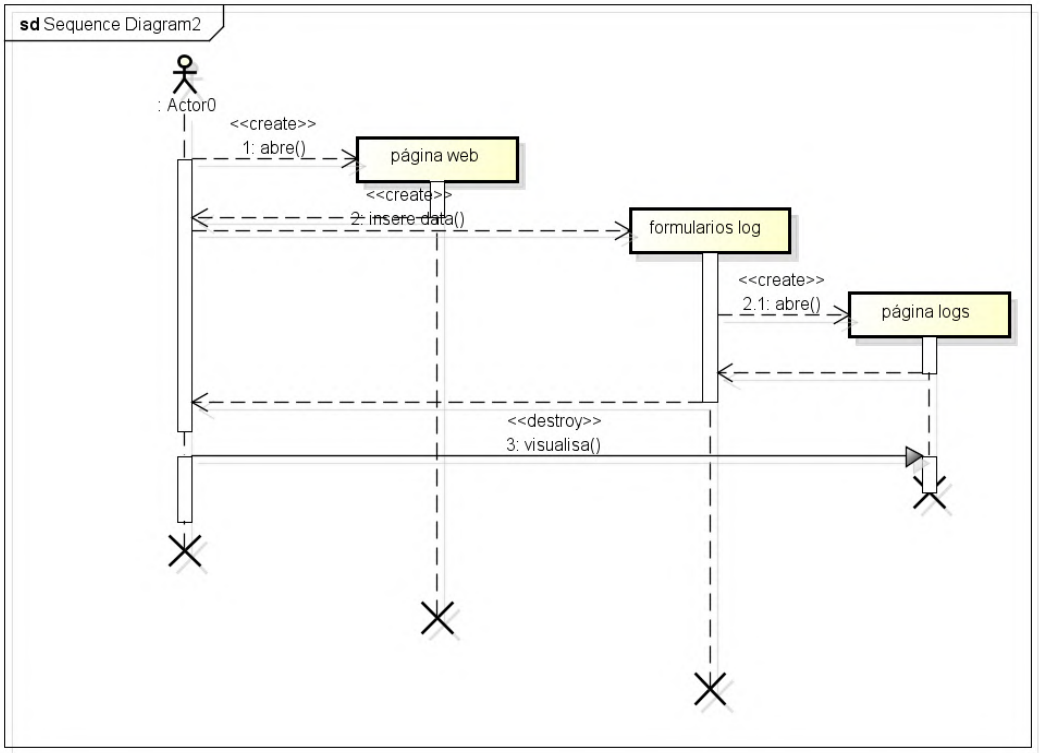
Segundo Craig Larman (p.198, 2005):

> Um diagrama de sequência do sistema é uma figura que mostra, para um cenário especifico de um caso de uso, os eventos que os atores externos geram, sua ordem e os eventos entre sistemas.

Craig Larman (2005) segue definindo que, o UML que inclui diagrama de sequência, serve para mostrar como os atores interagem com as operações iniciadas por eles.



Quadro 4: Diagrama de sequência gravar log

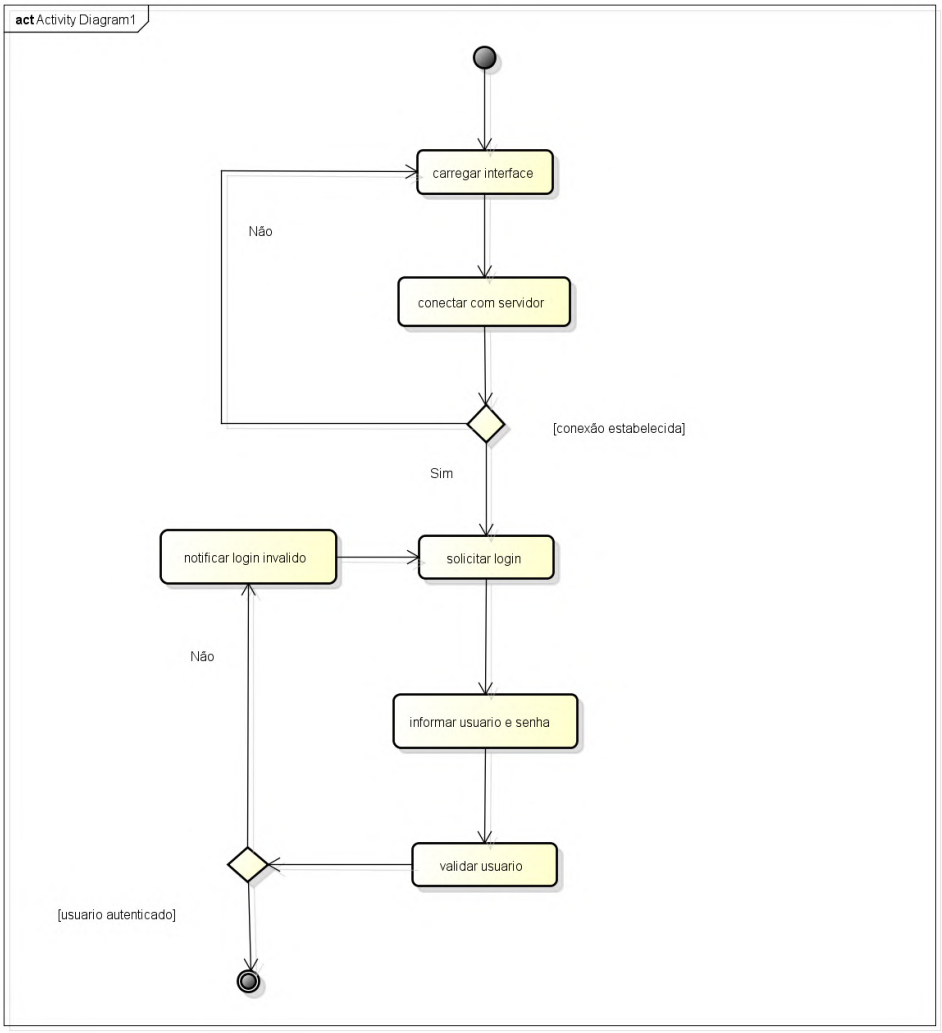


Quadro 5: Diagrama de sequência abrir logs por datas

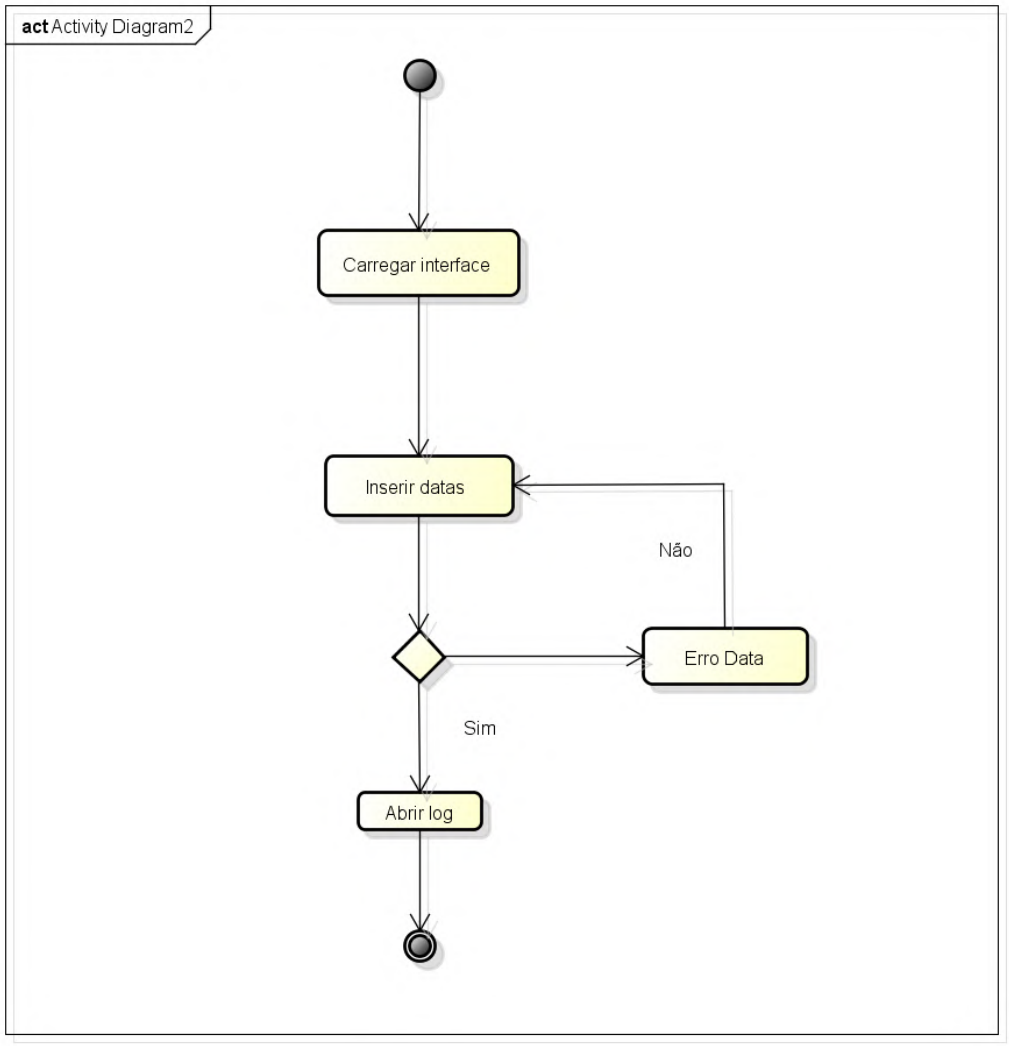
3.2.2.2 Diagrama de atividade

Um diagrama de classes da UML nos mostra uma sequência de atividades, incluindo atividades paralelas, a sua principal função é visualizar fluxos de trabalho e processos de negócio, além de casos de uso (Craig Larman, p. 484, 2005).

Segue os diagramas de atividade:.

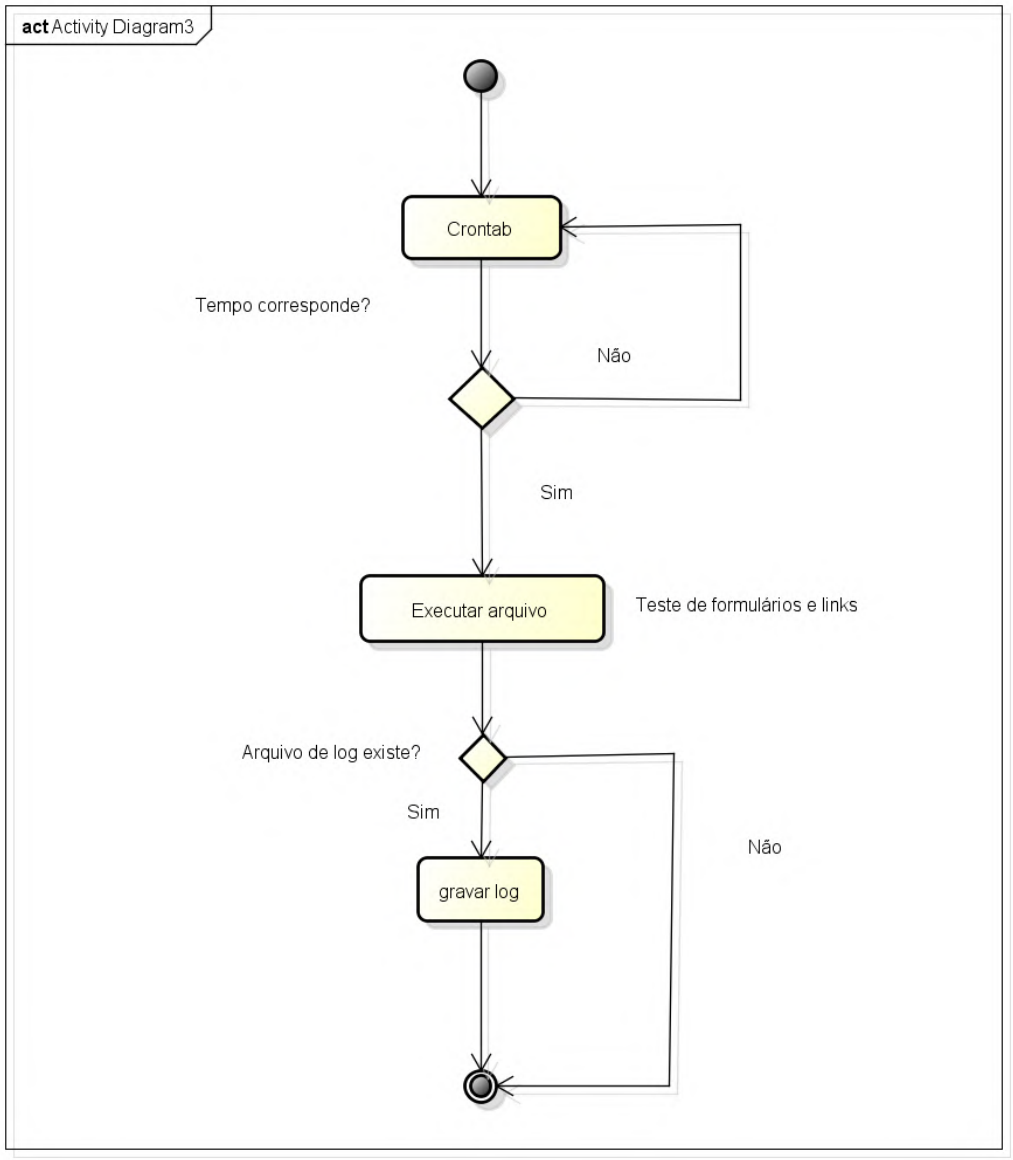


Quadro 6: Diagrama de atividade do processo de login



Quadro 7: Diagrama de atividade do processo selecionar logs por data

O diagrama acima mostra a validação da busca de datas na página principal.



Quadro 8: Diagrama de atividade do processo executar e gravar arquivos

3.2.3 Codificação

Para o desenvolvimento deste software, foi utilizada a linguagem de programação PHP versão 5.0, com o banco de dados Mysql, tal banco foi usado somente para cadastro de usuários, para acesso via *web*.

O servidor usado na aplicação é o Apache 2.2, instalado em ambiente linux Ubuntu versão 10.4.

Os resultados dos testes foram armazenados em arquivos de texto .txt, podendo ser visualizados com qualquer editor de texto.

3.3 Como Usar a Ferramenta MonitorE

O software de monitoramento MonitorE, para cada software que deseja ser monitorado deverá ser personalizado, abaixo segue os detalhes que o administrador deve ter para o uso do monitor.

O administrador do sistema deve atentar para os seguintes itens:

a. É necessário instalar o monitor em ambiente Linux ();
b. É necessário modificar o código do programa para incluir os *links* dos sites que serão monitorados;
c. Para verificar o funcionamento (SLAs e funcionalidade) da aplicação é necessário implementar no código do programa o acesso e as medições que deseja-se fazer.

3.3.1 Personalização

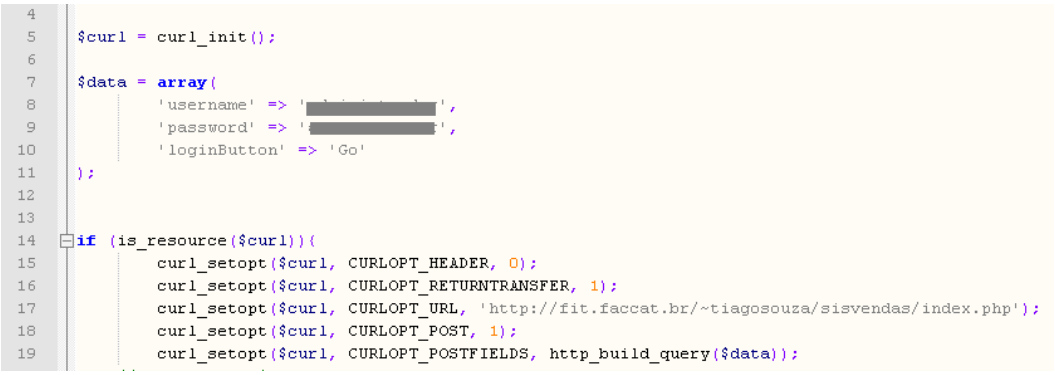
Nas seções seguintes serão mostrados os passos para a preparar o ambiente para a instalação do software, pois para cada sistema que deseja ser instalado é necessária uma personalização.

3.3.1.1 Inserindo URLs e formulários no PHP

O sistema MonitorE, não é um programa que após a instalação funcionará com qualquer sistema *web*. Ele precisa ser personalizado, adaptado para cada sistema a ser monitorado.

Particularidades como nomes e quantidade de URLs, formulários, não são

iguais em todos os softwares, sendo assim, será necessário inseri-los dentro do código, como mostra a Figura 10.



```
4
5   $curl = curl_init();
6
7   $data = array(
8           'username' => '[redacted]',
9           'password' => '[redacted]',
10          'loginButton' => 'Go'
11  );
12
13
14  if (is_resource($curl)){
15          curl_setopt($curl, CURLOPT_HEADER, 0);
16          curl_setopt($curl, CURLOPT_RETURNTRANSFER, 1);
17          curl_setopt($curl, CURLOPT_URL, 'http://fit.faccat.br/~tiagosouza/sisvendas/index.php');
18          curl_setopt($curl, CURLOPT_POST, 1);
19          curl_setopt($curl, CURLOPT_POSTFIELDS, http_build_query($data));
```

Figura 10 – Trecho de código arquivo teste_login.php

Como podemos observar na linha 7 existe um *array* recebendo os dados que compõe a tela de login. Mas nem todos os sistemas serão nesta sequência e também os nomes de usuários e senha são diferentes. Por isto é necessário para cada formulário a ser monitorado inserir no código de teste todos os campos que compõe este formulário.

Na linha 17, é passada o endereço *web* da página que possui o formulário de login.

A seguir é mostrado de que forma será utilizado o software MonitorE, como visualizar os arquivos de log, e a interação com a interface *web*.

3.3.1.2 Inserindo arquivos no Cron do Linux

Como foi explicado na seção 2.4.3 o Cron do Linux serve para agendar tarefas para serem executadas num período de tempo pré-estabelecido. Com o MonitorE será necessário que cada script que realiza testes de URLs e formulários sejam inseridos nele, para que possam ser auto executados, e os resultados sejam armazenados nos arquivos de log.

A Figura 12 nos mostra o Cron do Linux onde o MonitorE esta instalado, ao digitar no *shell* o comando crontab -e irá abrir o programa para ser editado.

Cada linha do arquivo corresponde a uma tarefa a ser agendada, utilizando o

wget que é um programa utilitário para baixar arquivos da internet utilizando protocolos HTTP, é chamado os arquivos para serem executados. (PHPBRASIL, 2010)

Após o arquivo de crontab ser criado o mesmo pode ser editado a qualquer momento.

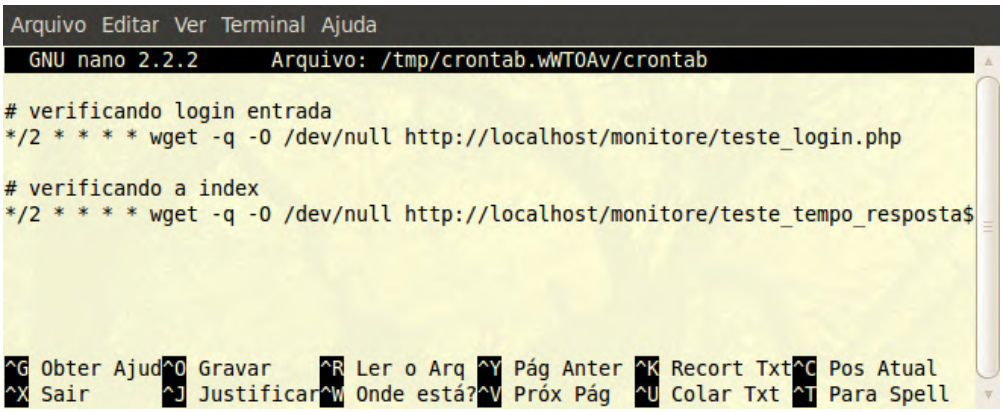


Figura 11 – Crontab Linux

Logo após a criação ou edição do crontab, digitando o comando *contab –l* no *shell* do Linux é listado todas as tarefas agendadas, como mostra a Figura 12.

Figura 12 – Listando tarefas agendadas no Crontab Linux

3.3.2 *Interface* com o usuário

O sistema poderá ser usado de duas formas, a primeira opção será quando o usuário optar por apenas analisar os arquivos de logs dentro do diretório onde o software MonitorE esta instalado. A segunda opção é ver os resultados em um ambiente *web* de forma gráfica.

Mas para isso o usuário dever ser devidamente cadastrado pelo administrador

para ter acesso ao sistema MonitorE via *web*.

Com o usuário cadastrado o mesmo acessa o endereço *web* onde o sistema está instalado, logo após ele é submetido a uma tela de login conforme a Figura 13.

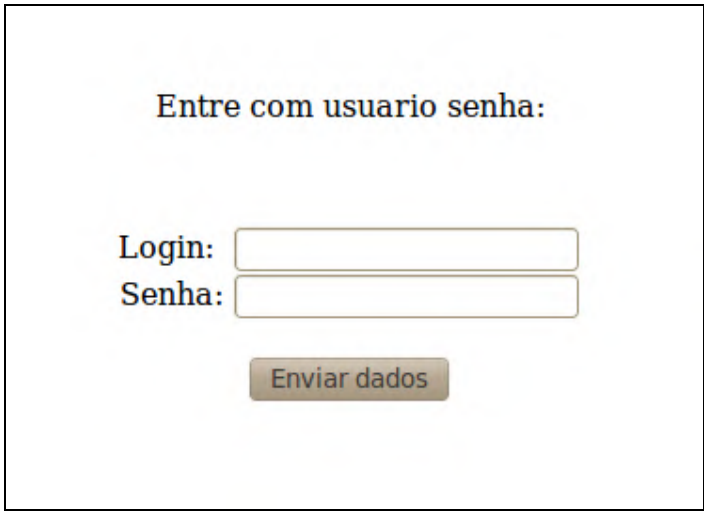


Figura 13 – Tabela de login

Logo após a autenticação do usuário, é mostrada para o mesmo uma página *web*, detalhes do sistema que esta sendo monitorado remotamente e funções que poderão ser executadas para melhor detalhamento dos arquivos de log.

A Figura 14 nos mostra a parte inicial da página *web*, onde a seta indica que no momento em que a página *web* do MonitoE foi carregado, automaticamente foi feito um teste na URL principal no sistema que esta sendo monitorado.

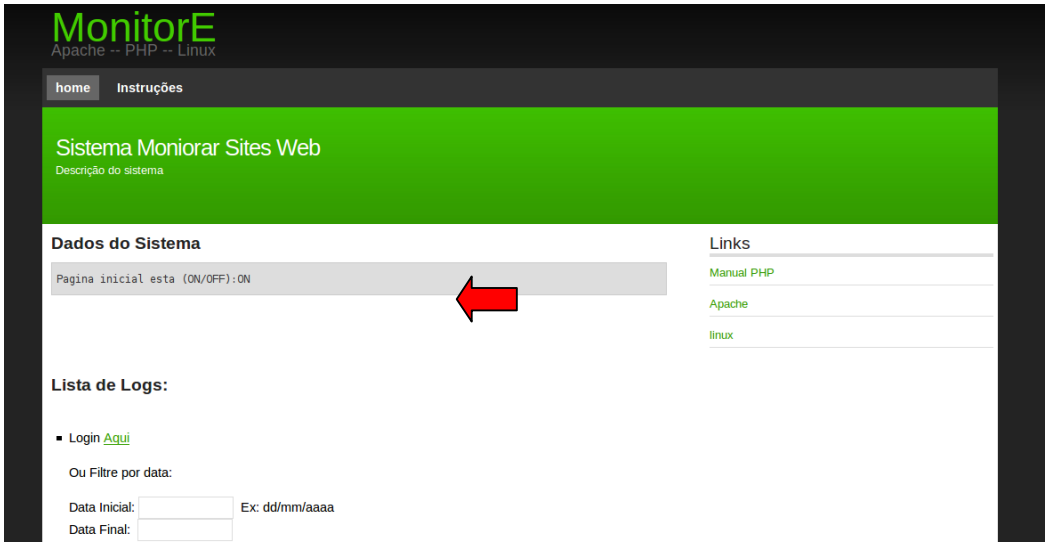


Figura 14 – Página *web* tela principal

Como os arquivos de log ao longo do tempo possuem muitos registros, pois os testes são feitos periodicamente, o usuário poderá optar por ver o arquivo completo ou especificar um intervalo inicial e final nas datas. A Figura 15 mostra o formato da página que é utilizada para fazer a seleção dos logs.

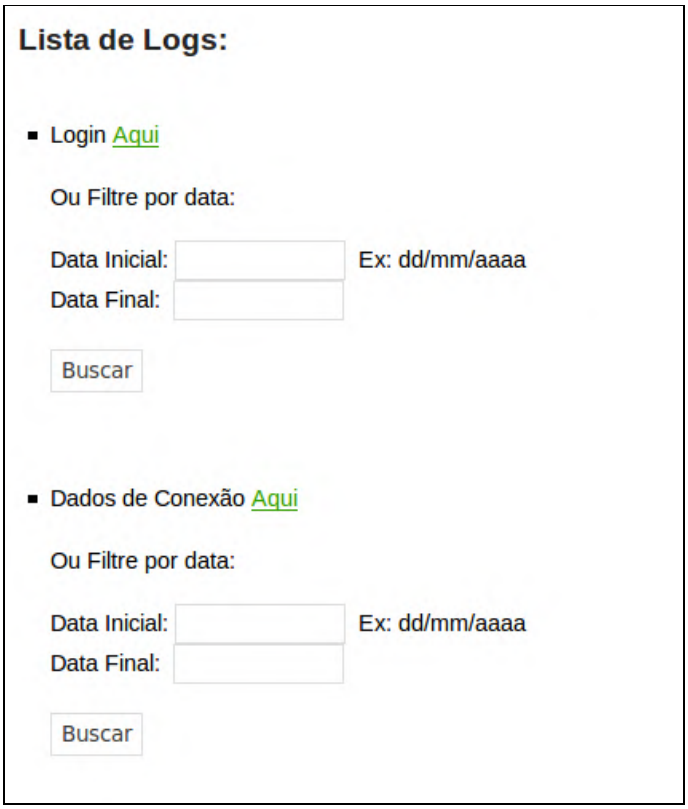


Figura 15 – Tela que abre os arquivos de logs

Na seção logs, o usuário deverá digitar no campo Data Inicial a data que deseja que comece a listagem dos logs, seguindo o padrão de dd/mm/aaaa, no campo Dara Final, o procedimento será da mesma forma que o anterior, mas o usuário colocara a data de término do log. Como mostra o Figura 16.

Figura 16 – Inserindo datas

Logo após é aberta uma nova janela do navegador contendo a sequência das linhas de logs respeitando o limite das datas inicial e final especificada anteriormente. Segue Figura 17.

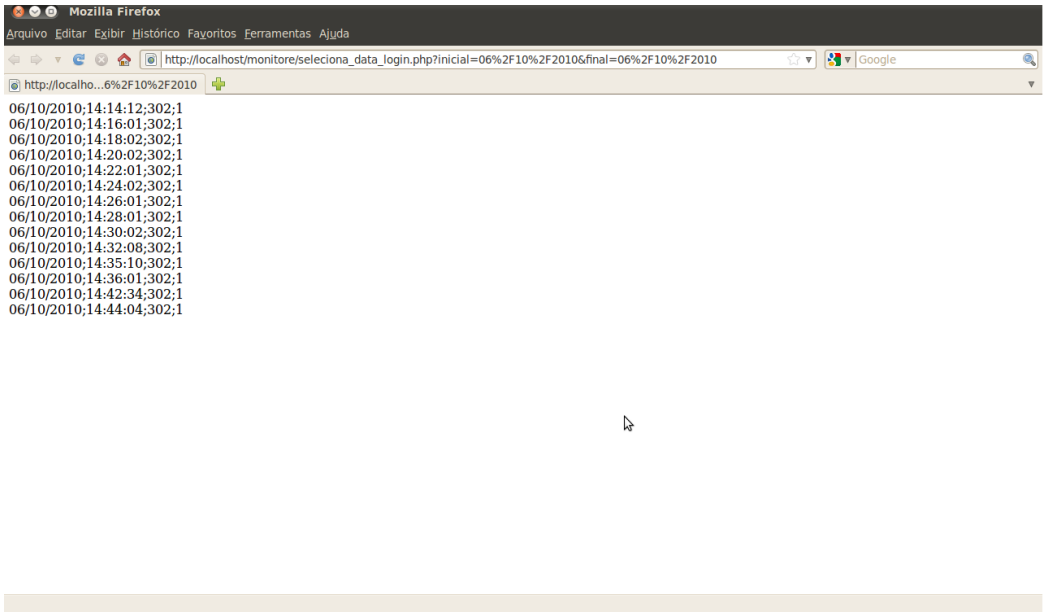


Figura 17 – Resultado da busca por data nos logs

Cada linha corresponde a uma sequência de teste programado anteriormente no Crontab do linux, os elemento das linhas estão separados por “;”, que no caso da Figura 18, o primeiro elemento é a data em que o teste foi realizado, o segundo é o horário o terceiro é a resposta HTTP, e por último é a resposta do login, onde o número 0 representa login falhou e o numero 1 representa sucesso na autenticação.

3.3.3 Analise de Gráficos

Para cada log existe a possibilidade de visualiza-los de forma de gráficos, a biblioteca usada é a LibChart conforme descrita na seção 2.4.3. Através de desta forma de visualização surge a possibilidade de uma melhor compreensão dos logs.

4 EXPERIMENTOS

Nesta seção, são demonstradas as técnicas de monitoramento e os resultados das simulações.

O cenário de monitoramento foi montado da seguinte forma:

- Servidor de Aplicação: Onde esta instalado o software que será monitorado. Este programa esta hospedado no servidor Apache, e a aplicação é em PHP.
- Agente de Monitoramento: É o programa que irá fazer o monitoramento, também foi instalado no servidor Apache e foi desenvolvido em PHP.

4.1 Simulação

Para a realização dos testes foi utilizado como sistema a ser monitorado o software PHP Point Of Sale na versão 10.1, desenvolvido por Chris Muench. (PHP POS, 2010).

O PHP Point Of Sale é direcionado para pequenas empresas, com acompanhamento de clientes, produtos e inventário, gerar relatórios baseados em vendas.

Este software foi desenvolvido em PHP que utiliza banco de dados MySQL e é utilizado em servidores.

Com o objetivo identificar e mostrar os erros ocorridos no sistema e o tempo e resposta, foram criados alguns cenários de testes, que serão detalhados ao decorrer desta seção.

As simulações foram feitas em duas situações, a primeira realizando teste em um formulário e a outra testando uma URL. Mas o sistema a ser monitorado é para ser testado todas as URLs e todos os formulários, para que o MonitorE tenha mais precisão.

O tempo que as simulações foram realizadas era a cada 2 minutos.

4.1.1 Ambiente de teste

As simulações foram feitas no computador, que possui as seguintes configurações:

- Processador Intel® Celeron® M4440 1.86 GHz, com 1GB de RAM, *Motherboard* Campal, modelo HGL3X, sistema operacional Ubuntu, versão 10.04;
- Servidor Apache, versão 2.2.16;
- PHP, versão 5.3.3;
- *Internet* ADSL, OI, velocidade de 1Mbps.

4.1.2 Teste login

Para realizar este experimento, foi configurado um script que simula a realização de uma autenticação no sistema PHP Point Of Sale.

A primeira parte da simulação foi abrindo o arquivo de log, e verificando se o usuário estava autenticando. A Figura 18 demonstra esse caso.

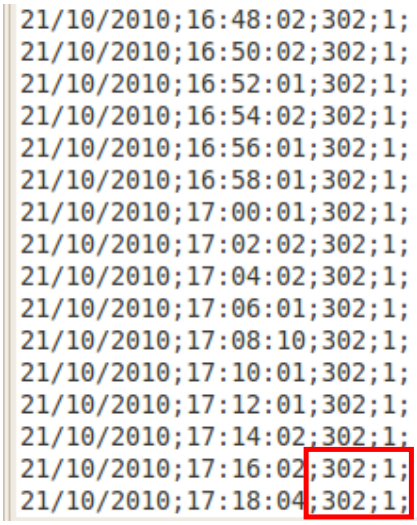


Figura 18 – Parte do arquivo de log que autentica usuário

Ao observar a Figura 18, nas últimas linhas, a resposta 302, que significa que o endereço existe e foi redirecionado. Logo após o numero 302 há o número 1 que

significa que o usuário logou no sistema.

Na sequência, há um cenário onde existe uma desconexão do servidor (ou rompimento de *link*) fazendo com que o sistema fique indisponível. Para simular esse cenário foi retirado o cabo da *internet* do computador, interrompendo a conexão com o sistema. As duas últimas linhas da imagem abaixo nos mostra a resposta.

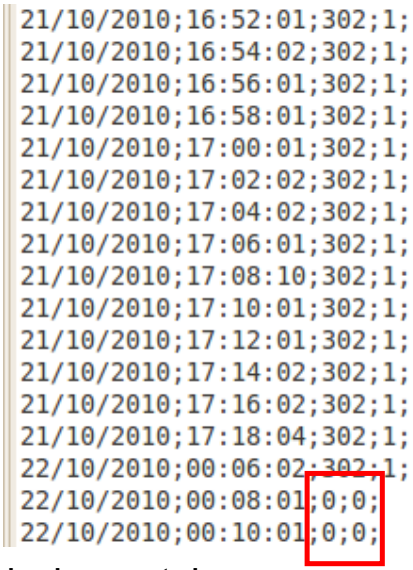


Figura 19 – Resposta log com cabo desconectado

Neste caso podemos concluir que ao desconectar a *internet* do computador, não foi registrado nenhuma resposta HTTP e também não ocorreu o login do usuário. Demonstrando que o sistema é capaz de detectar falhas de ruptura de rede nos sistemas sendo monitorados

Outro caso foi trocado apenas o endereço da página do sistema que esta sendo monitorado. A Figura 20 nos mostra o resultado.

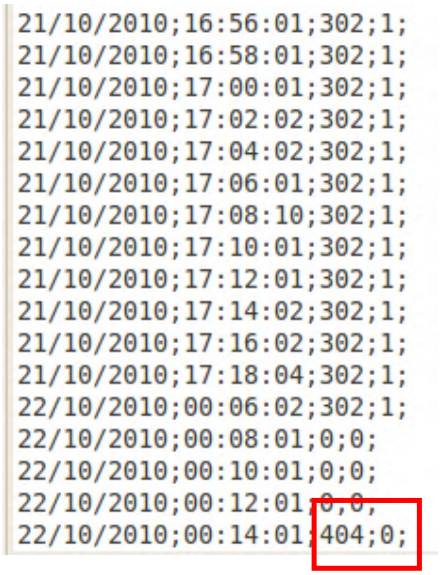


Figura 20 – Resposta log erro URL

Este cenário de teste teve como objetivo simular situações onde são feitas novas instalações e que podem fazer com que páginas sejam removidas ou mudadas de localização.

A última linha nos mostra que ao trocar o endereço, o erro HTTP foi 404 que significa, página não existe, e logo após na mesma linha o 0 corresponde a não autenticação do usuário.

E por fim com o intuito de criar um cenário onde há problemas de autenticação de usuário, criou-se uma situação onde a senha do usuário foi modificada, que estava dentro do código PHP que executa os testes. A Figura 21 apresenta os resultados

```
21/10/2010;16:50:02;302;1;
21/10/2010;16:52:01;302;1;
21/10/2010;16:54:02;302;1;
21/10/2010;16:56:01;302;1;
21/10/2010;16:58:01;302;1;
21/10/2010;17:00:01;302;1;
21/10/2010;17:02:02;302;1;
21/10/2010;17:04:02;302;1;
21/10/2010;17:06:01;302;1;
21/10/2010;17:08:10;302;1;
21/10/2010;17:10:01;302;1;
21/10/2010;17:12:01;302;1;
21/10/2010;17:14:02;302;1;
21/10/2010;17:16:02;302;1;
21/10/2010;17:18:04;302;1;
22/10/2010;00:06:02;302;1;
22/10/2010;00:08:01;0;0;
22/10/2010;00:10:01;0;0;
22/10/2010;00:12:01;0;0;
22/10/2010;00:14:01;404;0;
22/10/2010;00:16:01;200;0;
```

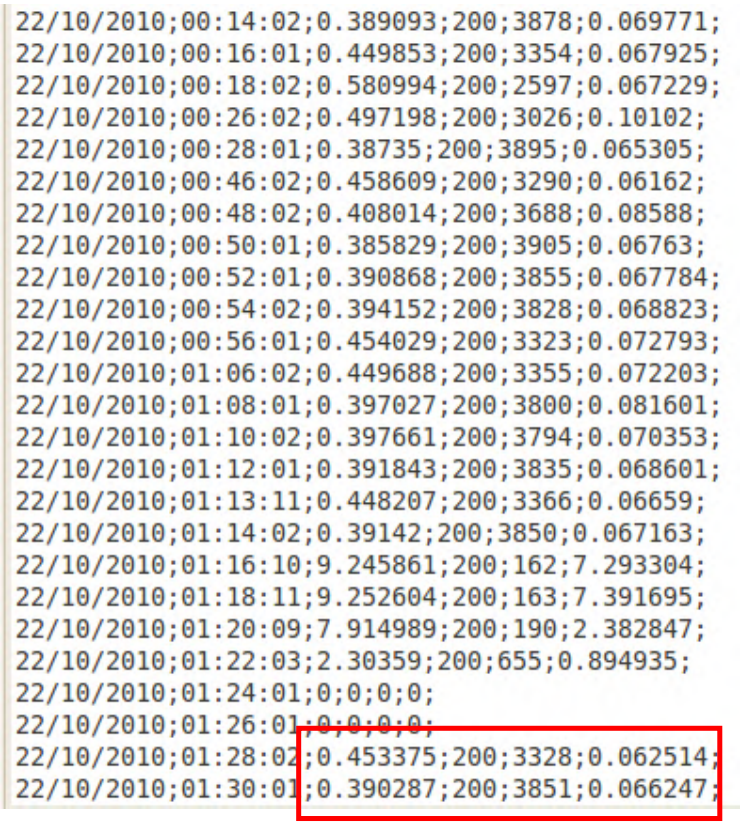
Figura 21 – Resposta log erro senha

Nesta resposta podemos observar que a resposta HTTP foi 200, o endereço foi acessado mas o número 0 após nos mostra que não ocorreu a autenticação.

4.1.3 Teste trafego de rede

Outro teste realizado foi o de observar o tempo de resposta em diferentes situações. O endereço testado foi a URL principal do sistema PHP Point Of Sale.

No primeiro caso, foi observado quais os valores do tempo de resposta, sem trafego algum na rede. Segue a Figura 22.



```
22/10/2010;00:14:02;0.389093;200;3878;0.069771;
22/10/2010;00:16:01;0.449853;200;3354;0.067925;
22/10/2010;00:18:02;0.580994;200;2597;0.067229;
22/10/2010;00:26:02;0.497198;200;3026;0.10102;
22/10/2010;00:28:01;0.38735;200;3895;0.065305;
22/10/2010;00:46:02;0.458609;200;3290;0.06162;
22/10/2010;00:48:02;0.408014;200;3688;0.08588;
22/10/2010;00:50:01;0.385829;200;3905;0.06763;
22/10/2010;00:52:01;0.390868;200;3855;0.067784;
22/10/2010;00:54:02;0.394152;200;3828;0.068823;
22/10/2010;00:56:01;0.454029;200;3323;0.072793;
22/10/2010;01:06:02;0.449688;200;3355;0.072203;
22/10/2010;01:08:01;0.397027;200;3800;0.081601;
22/10/2010;01:10:02;0.397661;200;3794;0.070353;
22/10/2010;01:12:01;0.391843;200;3835;0.068601;
22/10/2010;01:13:11;0.448207;200;3366;0.06659;
22/10/2010;01:14:02;0.39142;200;3850;0.067163;
22/10/2010;01:16:10;9.245861;200;162;7.293304;
22/10/2010;01:18:11;9.252604;200;163;7.391695;
22/10/2010;01:20:09;7.914989;200;190;2.382847;
22/10/2010;01:22:03;2.30359;200;655;0.894935;
22/10/2010;01:24:01;0;0;0;0;
22/10/2010;01:26:01;0;0;0;0;
22/10/2010;01:28:02;0.453375;200;3328;0.062514;
22/10/2010;01:30:01;0.390287;200;3851;0.066247;
```

Figura 22 – Resposta log tempo de resposta

Em cada linha as informações são separadas por “;” onde na primeira sequência corresponde a data, em segundo vem a hora, em terceiro esta os números do tempo de resposta, logo após vem códigos http, depois velocidade de transferência e por último o valor mostrado significa o tempo em que a resolução tenha sido completada.

Analisando a última linha do log, ele nos mostra que o tempo de resposta com a rede sendo pouco usada é de 0.390287, 200 é a resposta HTTP que indica a página foi acessada.

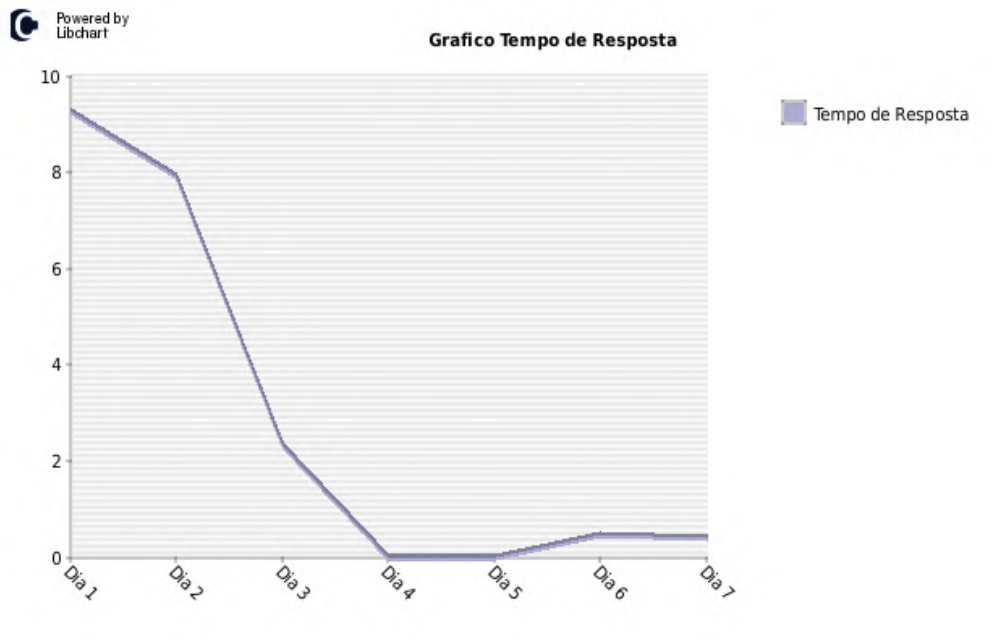


Figura 23 – Gráfico tempo de resposta

A figura 23 serve para uma melhor vizualização do tempo de resposta, tomando como base os 7 últimos registros do log da Figura 23, tais informações são baseadas no log na Figura 22.

No caso a seguir a rede estava com alto trafego, provocado para simular um alto tempo de resposta. Foi utilizado um *download* a 95 Kbps, a Figura 25 nos mostra com mais clareza o resultado.

```
22/10/2010;00:14:02;0.389093;200;3878;0.069771;
22/10/2010;00:16:01;0.449853;200;3354;0.067925;
22/10/2010;00:18:02;0.580994;200;2597;0.067229;
22/10/2010;00:26:02;0.497198;200;3026;0.10102;
22/10/2010;00:28:01;0.38735;200;3895;0.065305;
22/10/2010;00:46:02;0.458609;200;3290;0.06162;
22/10/2010;00:48:02;0.408014;200;3688;0.08588;
22/10/2010;00:50:01;0.385829;200;3905;0.06763;
22/10/2010;00:52:01;0.390868;200;3855;0.067784;
22/10/2010;00:54:02;0.394152;200;3828;0.068823;
22/10/2010;00:56:01;0.454029;200;3323;0.072793;
22/10/2010;01:06:02;0.449688;200;3355;0.072203;
22/10/2010;01:08:01;0.397027;200;3800;0.081601;
22/10/2010;01:10:02;0.397661;200;3794;0.070353;
22/10/2010;01:12:01;0.391843;200;3835;0.068601;
22/10/2010;01:13:11;0.448207;200;3366;0.06659;
22/10/2010;01:14:02;0.39142;200;3850;0.067163;
22/10/2010;01:16:10;9.245861;200;162;7.293304;
22/10/2010;01:18:11;9.252604;200;163;7.391695;
22/10/2010;01:20:09;7.914989;200;190;2.382847;
```

Figura 24 – Log tempo de resposta com trafego na rede

Neste caso ouve um aumento considerável no tempo de resposta, que antes era de 0.390287 agora subiu para 7.914989. A Figura 25 mostra a curva provocada por esse aumento na resposta da solicitação.

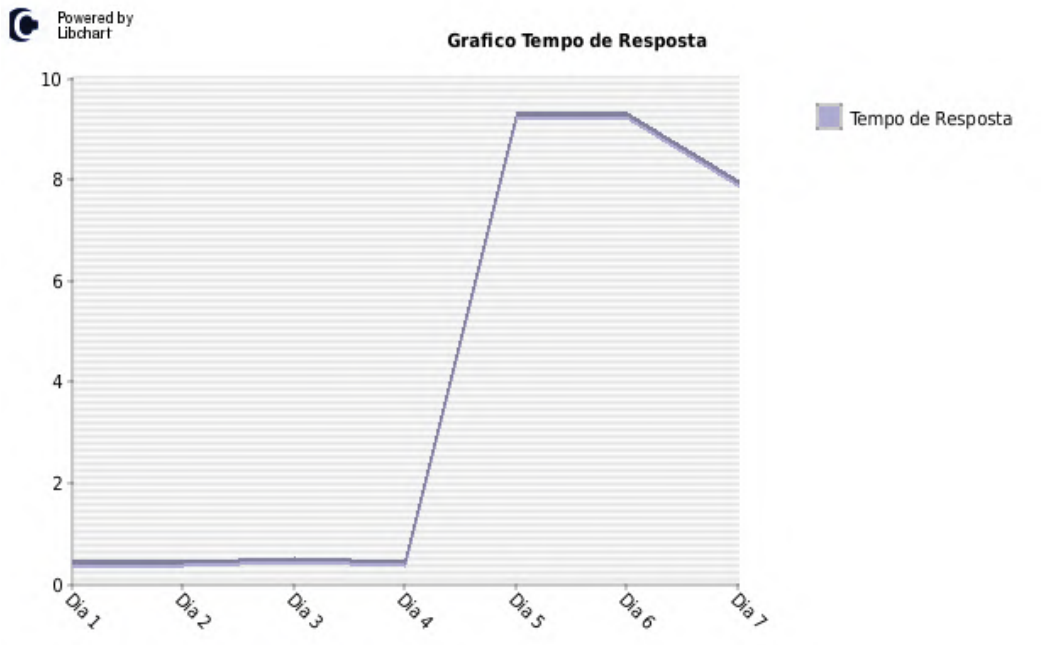


Figura 25 – Gráfico tempo de resposta com trafego na rede

4.2 Resultado das experiências

Nesta subseção, será tratado sobre o ambiente onde foram produzidos os experimentos e os resultados obtidos nos testes realizados.

4.2.1 Resultados Obtidos

Resumindo os resultados obtidos nas seções 4.1.1 e 4.1.2, onde as simulações foram as seguintes:

a. analise da situação atual do log;
b. retirada do cabo de rede;
c. troca do endereço onde o formulário de login se encontra;
d. troca da senha do usuário, provocando erro no login;
e. analise do tempo de resposta do *link* da página principal.

Todos os testes realizados foram programados e direcionados para que as situações ocorressem.

Os resultados finais da ferramenta foram satisfatórias, pois os resultados das simulações foram conforme o esperado.

4.3 Análises dos resultados

4.3.1 Análise final do *software* MonitorE

O software de monitoramento MonitorE ao realizar os testes, mostrou resultados satisfatórios, visto que a ferramenta visa uma percepção do usuário diante de falhas no sistema. O usuário ao acessar os logs e perceber que existem erros como ocorreram nas simulações, tomará medidas para solucionar os mesmos.

O acesso a uma página *web* mostrando histórico dos resultados e filtros para

os arquivos de log foi de grande valia, pois descarta a necessidade do usuário logar remotamente no servidor e acessar os arquivos por linha de comando onde o programa monitor está instalado, tornando possível a visualização dos resultados em qualquer lugar, bastando ter acesso a *internet*.

Com o auxilio de gráficos, a visualização dos resultados dos testes torna a ferramenta mais atrativa, fazendo com que a comparação de resultados e a forma em que eles mudam fica melhor o entendimento.

Uma das limitações do software é no quesito de personalização, onde o programador que ira instalar o MonitorE deverá inserir no código fonte todos os formulários e *links* que o programa a ser monitorado terá, dependendo do tamanho do sistema isso poderá se tornar um serviço que demanda tempo.

5 CONCLUSÃO

Este trabalho apresentou uma solução para softwares *web* que necessitam estar sempre com seus *links* ativos, formulários funcionando, e respeitando os SLAs contratados. Neste caso ao usuário analisar um arquivo de log gerado pelo monitor e se deparar com alguma resposta de erro, imediatamente poderá ser tomada às medidas necessárias para solucionar o problema.

Uma possibilidade para soluções de problemas identificados e registrados nos logs seria integrar o MonitorE a softwares de alertas via email ou SMS, com isso ao monitor identificar algum problema estes softwares de alerta enviariam mensagens para o usuário do sistema.

A possibilidade de poder analisar os resultados dos logs via *internet*, através de uma página *web* é um dos principais pontos positivos, pois através dela esta verificação não se restringe apenas onde a aplicação se encontra instalada, e sim a outros locais onde estiver um computador com acesso a *internet*.

As linhas dos arquivos de logs estão organizados por data, possibilitando uma melhor visualização e compreensão dos dados capturados dos testes.

Este aplicativo de monitoramento, mostrou-se útil para realizar analises em sites *web*, tornando-se uma alternativa para empresas ou qualquer usuário que possuem seus sistemas direcionados para *internet* estejam apresentando o mínimo de transtorno. Conforme os resultados apresentados foi demonstrado que os objetivos deste trabalho foram alcançados.

6 REFERÊNCIAS

TAYI, G. K.; BALLOU, D. P. Examining data quality. **Communications of the ACM.** New York, v. 41, n.2, p. 54-57, Fev. 1998.

eCommerceOrg. **Evolução da internet e do e-commerce**. Disponível em <http://www.e-commerce.org.br/stats.php>. Acesso em 21/07/2010.

eCommerceOrg. **Quanto custa montar um negócio na internet?**. Disponível em < http://www.e-commerce.org.br/artigos/custo_comercioeletronico.php>. Acesso em 21/07/2010.

PUC-SP. **SLA - Como Estabelecer e Garantir Níveis de Qualidade**. Disponível em <http://www.pucsp.br/>. Acesso em 21/09/2009.

GOMES, Silva Bogéa, Ricardo de Almeida Falbo, Crediné Silva de Manezes – **Um Modelo para Acordo de Nível de Serviço em TI**. Disponível em: <http://www.scribd.com/doc/37149539/SLA>. Acesso em: 26/08/2010.

PHP. Libcurl – **Introdução**. Disponível em <http://www.php.net/manual/pt_BR/intro.curl.php>. Acesso 28/09/2010.

IBM. **Conversação pela internet com cURL e libcurl**. Disponível em <http://www.ibm.com/developerworks/br/opensource/library/os-curl/>. Acesso em 29/09/2010.

KOSCIANSKI, André; SOARES, Michel dos S. **Qualidade de software**. 2 ed. São Paulo: Novatec, 2007.

BRAUDE, Eric. **Projeto de Software**. Da programação à arquitetura: uma abordagem baseada em java. 1 ed. São Paulo: Bookman, 2004.

LARMAN, Craig. **Utilizando UML e Padrões**: Introdução à analise e ao projeto orientados a objeto e ao desenvolvimento interativo. 3 ed. São Paulo: Bookman, 2005.

NEVES, José Cezar. **Programação Shell Linux**. 7 ed. Rio de Janeiro: Brasport, 2008.

ZUFFO, João Antonio. **A Sociedade e a Economia no Novo Milênio**. Livro I – A Tecnologia e a Infossociedade. 1 ed. Tambaré: Manole, 2003.

REVISTA ENGENHARIA, **Arquitetura de rede IPTV – Com Acesso baseado em**

tecnologia ADSL, Disponível em: <http://www.brasilengenharia.com.br/ed/595/Art_eletrica.pdf>. Acesso em: 03/12/2009.

AMORIN, Ana Amélia, Alcino Simões, João Paulo da Silva, **Indicadores de Qualidade e de Confiança de um Site**. Disponível em: <http:// repositorium.sdum.uminho.pt/bitstream/1822/.../05AnaAmelia.pdf >. Acesso em: 11/10/2010.

CANHETTE, C. C. **Análise das menções à qualidade da informação em teses e dissertações que relatam impactos do uso em sistemas ERP**. Dissertação de Mestrado. Faculdade de Economia, Administração e Contabilidade da Universidade de São Paulo, 2004.

STURM, R, Morris, W. Jander, M. (2000). **Foundations of Service Level Management**. Estados Unidos.

GOVERNO DO ESTADO DO CEARÁ. **Acordo de Nível de Serviço**. Disponível em: <http://wiwiki.pge.ce.gov.br/index.php/Acordo_de_Nível_de_Serviço>. Acesso em: 11/10/2010.

cURL. **Libcurl – The multiprotocol file transfer library**. Disponível em: <http://curl.haxx.se/libcurl/>. Acesso em: 11/10/2010.

IMASTERS. **Usando a biblioteca CURL do PHP**. Disponível em: < http://www.imasters.com.br/artigo/4140/php/usando_a_biblioteca_curl_do_php >. Acesso em 11/10/2010.

cURL. **History – How did this happen**. Disponível em: <http://curl.haxx.se/docs/history.html>. Acesso em: 11/10/2010.

Libchart. **Instruction**. Dinsponível em: <http://naku.dohcrew.com/libchart/pages/introduction/>. Acesso em: 12/10/2010

PHP POS. **PHP Point Of Sale (POS)**. Disponível em: <http://www.phppointofsale.com/>. Acesso em: 17/10/2010.

WIKIPEDIA. **Hypertext Protocol Transfer**. Disponível em: <http://pt.wikipedia.org/wiki/HTTP>. Acesso 12/10/2010

COULOURIS, Gorge, Jean Dollimore, Tim Kindberg. **Sistemas Distribuídos**. 4 ed. Porto Alegre: Artmed, 2005.

TORRES, Bruno. **Protocolo http: códigos de resposta mais comuns e seus significados**. Disponível em: <http://www.obasicodaweb.com/http-codigos-de-resposta-mais-comuns-e-seus-significados/>. Acesso em: 12/10/2010.

PHPBRASIL. **Utilizando Crontab/agenda de tarefas com o PHP**. Disponível em: <http://phpbrasil.com/artigo/QckB67Mux3na/utilizando-crontabagenda-de-tarefas-com-o-php>. Acesso em; 12/10/2010.