UNIVERSIDADE DE SÃO PAULO
Escola de Artes, Ciências e Humanidades

Marcus Vinicius Cerrato

Utilização do *ErgoParser* em Arquivos de *Log* para Determinação de Métricas de Usabilidade de *Websites*

São Paulo

Dezembro 2012

Marcus Vinicius Cerrato

Utilização do *ErgoParser* em Arquivos de *Log* para Determinação de Métricas de Usabilidade de *Websites*

Dissertação apresentada como parte da avaliação do programa de mestrado em Sistemas de Informação EACH - Escola de Artes, Ciências e Humanidades.

Orientador: Prof. Dr. Marcelo Morandini

VERSÃO CORRIGIDA
(original disponível na EACH-USP para consulta)

São Paulo

Dezembro 2012

Autorizo a reprodução e divulgação total ou parcial deste trabalho, por qualquer meio convencional ou eletrônico, para fins de estudo e pesquisa, desde que citada a fonte.

CATALOGAÇÃO-NA-PUBLICAÇÃO
Biblioteca
Escola de Artes, Ciências e Humanidades da Universidade de São Paulo

Cerrato, Marcus Vinicius
Utilização do ergoparser em arquivos de log para determinação de métricas de usabilidade de websites / Marcus Vinicius Cerrato ; orientador, Marcelo Morandini. – São Paulo, 2012.
72 f. : il.

Dissertação (Mestrado em Ciências) – Programa de Pós-Graduação em Sistema de Informação, Escola de Artes, Ciências e Humanidades da Universidade de São Paulo.

1. Web sites - usabilidade. 2. Word Wide Web. 3. Interação usuário-computador . 4. Interface homem-computador. 5. Servidores de rede. I. Morandini, Marcelo, orient. II. Título.

CDD 22.ed. – 004.678

Dissertação de mestrado sob o título “Utilização do *ErgoParser* em Arquivos de *Log* para Determinação de Métricas de Usabilidade de *Websites*", defendida por Marcus Vinicius Cerrato e aprovada em 17 de dezembro de 2012, em São Paulo, Estado de São Paulo, pela banca examinadora constituída pelos doutores:

Prof. Dr. Marcelo Morandini
Orientador

Profa. Dra. Lucia Filgueiras
Universidade de São Paulo

Prof. Dr. Marcelo Fantinato
Universidade de São Paulo

Dedico este trabalho a Deus e à minha família.

Agradeço a Deus, que me deu sabedoria e dedicação para realizar e concluir este trabalho. Agradeço ainda a meus falecidos pais, que sempre me incentivaram nos caminhos do conhecimento. Agradeço também a minha esposa e minhas filhas, pelo apoio e pela motivação que me fortalecem. Agradeço a meus irmãos, pelo companheirismo que me ampara. Agradeço pela minha família e amigos, por terem colaborado ao aceitarem minhas ausências para me dedicar a este trabalho. Agradeço, por fim, a todos que já atuaram em algum momento como meus professores, pelo conhecimento e formação proporcionados.

“Diga ao povo que marche!”
(Texto do Livro de Êxodo da Bíblia)

Resumo

A usabilidade é um requisito valioso para os produtos de Sistemas de Informação. A avaliação de usabilidade está cada vez mais se tornando uma parte importante do processo de desenvolvimento de software. O objetivo deste trabalho é propôr uma abordagem para suportar avaliação de usabilidade de sistemas *web* por meio da análise de arquivos de *log* dos servidores *web*. A metodologia desenvolvida neste trabalho consiste em filtrar arquivos de *log*, para capturar as tarefas realizadas, e comparar com um modelo descritivo das tarefas possíveis de serem desempenhadas em um *website*. As tarefas realizadas são descritas pelos recursos *web* essenciais para o cumprimento dessas tarefas. As métricas de usabilidade obtidas deste processo podem indicar possíveis problemas de usabilidade. Métodos de avaliação de usabilidade mais detalhados podem ser aplicados para verificar ou não se estes problemas realmente ocorrem. Além disso, a comparação de métricas permite identificar o quanto positivas foram algumas mudanças com relação à eficácia e à eficiência na interação com o website em avaliação. Este trabalho resultou no desenvolvimento de uma ferramenta que recebe um arquivo de *log* referente a um *website*, filtra seu conteúdo, de acordo com páginas *web* relacionadas a tarefas previamente cadastradas, e calcula métricas que quantifiquem a usabilidade do *website* avaliado.

Abstract

Usability is a valuable requirement for Information Systems' products. Usability evaluation is increasingly becoming an important part of software development process. The objective of this work is to propose an approach for supporting usability evaluation of web systems by analyzing log files of web servers. The methodology developed in this work is to filter log files, to capture the tasks performed, and compare with a model describing the possible tasks to be performed on a website. The tasks performed are described by web resources essential to the fulfillment of these tasks. The usability metrics obtained by this process can indicate possible usability problems. More detailed usability evaluation methods may be applied to verify whether or not these problems do occur. Furthermore, comparison of metrics identifies how positive some changes were with respect to the effectiveness and the efficiency of the interaction with the website in evaluation. This work resulted in the development of a tool that receives a log file relating to a website, filter content, according to web pages related tasks previously registered, and calculate metrics that quantify the usability of the evaluated website.

Sumário

Lista de Figuras xi

Lista de Tabelas xii

1 Introdução 1

1.1 Considerações Iniciais 1

1.2 Justificativa e Objetivos 6

1.3 Objetivos Específicos e Metodologia 9

1.4 Limitações do Trabalho 10

2 Revisão Bibliográfica 12

2.1 Usabilidade 12

2.2 Atributos de Usabilidade 14

2.3 Avaliação de Usabilidade 18

2.3.1 Métodos de Avaliação de Usabilidade 20

2.3.2 *Heurísticas Propostas por Jakob Nielsen* 24

2.3.3 Critérios Ergonômicos Propostos por Scapin e Bastien 25

2.4 Técnicas de Co-Inspeção Avaliativa 27

2.5 Automatização do Processo de Avaliação 29

2.6 A Proposição do ErgoMonitor 31

2.7 Automatização e Atributos de Usabilidade 33

3 Estruturas Computacionais Empregadas 34

3.1 Arquitetura Cliente/Servidor 34

3.2 Arquivos de *Log* 36

3.3 Arquivos de L*og* no Acesso ao Servidor *Web* 39

3.4 Reconhecimento da Estrutura dos Arquivos de *Log* 40

4 A Implementação de um Parser para Auxiliar na Determinação de Taxas e Métricas de Usabilidade 43

4.1 Desenvolvimento da Ferramenta 44

4.1.1 Mapeamento das Tarefas 47

4.1.2 Armazenamento em Banco de Dados 50

4.1.3 *Upload* de Arquivos de *Log* para o Servidor *Web* 54

4.2 Validação da Ferramenta 58

4.2.1 Exemplo de Avaliação 59

4.2.2 Definição das Taxas e Métricas Calculadas e Análise do Resultado da Avaliação 61

5 Conclusões 65

5.1 Considerações Finais 65

5.2 Trabalhos Futuros 67

Referências 70

Apêndice A 76

Lista de Figuras

FIGURA 1 Visão Geral do ErgoMonitor 31

FIGURA 2 Estrutura do ErgoMonitor 46

FIGURA 3 Tela para Cadastro de Nova Tarefa. 50

FIGURA 4 Modelo Entidade-Relacionamento 51

FIGURA 5 Colocação de Novo Usuário no Banco de Dados 53

FIGURA 6 Colocação das Informações Básicas do Website no Banco de Dados . 54

FIGURA 7 Tela da Ferramenta de *Upload* de Arquivo de *Log* 57

FIGURA 8 *Upload* de Arquivo de *Log* 58

Lista de Tabelas

TABELA 1 **Heurísticas de Nielsen** **24**

TABELA 2 **Critérios Ergonômicos por Scapin e Bastien** **25**

TABELA 3 **Subcritérios Ergonômicos por Scapin e Bastien** **26**

TABELA 4 **Ferramentas que Avaliam Usabilidade** **33**

TABELA 5 **Exemplo de Entrada de Arquivo de Log** **36**

TABELA 6 **Formato de Arquivo de Log no Apache** **36**

TABELA 7 **Códigos de Status Devolvidos pelo Servidor Web** **37**

TABELA 8 **Alguns Códigos de Staus Comentados** **38**

TABELA 9 **Função preg_match() Usada em Arquivos de Log** **42**

TABELA 10 **Conceito de Tarefa** **47**

TABELA 11 **Trecho de Arquivo de Log com Tarefa** **48**

TABELA 12 **Comandos Linux para Filtrar IP** **55**

TABELA 13 **Comandos Windows para Filtrar IP** **55**

TABELA 14 **Trecho de Arquivo de Log para Filtrar IP** **56**

TABELA 15 **Uso do Comando para Filtrar IP** **56**

TABELA 16 **Trecho de Arquivo de Log com IP Filtrado** **56**

TABELA 17 **Exemplo de Avaliação** **60**

TABELA 18 **Resultado de Avaliação** **63**

1 Introdução

Nas seções iniciais deste trabalho são introduzidos os conceitos usados no desenvolvimento dele. Após uma discussão inicial sobre usabilidade, é apresentada uma justificativa sobre a necessidade deste trabalho, seus objetivos, sua metodologia e suas limitações.

1.1 Considerações Iniciais

A Rede Mundial de Computadores, também denominada Internet, pode ser visualizada por meio de um programa conhecido como navegador. Ela é composta por diversos documentos ligados entre si. Um *website* é um local nesta Rede que contém informações sobre pessoas, organizações ou outros; e geralmente consiste de diversas páginas *web* agrupadas [Mairs, 2012].

Na Rede ocorre uma alternância entre quem está comunicando e quem está recebendo a comunicação, interpretando e reagindo a ela. Esse processo de comunicação entre pessoas e sistemas é denominado interação [Mairs, 2012]. Para permitir essa comunicação, os sistemas devem possuir uma interface de usuário, a qual deve ser entendida como a parte de um sistema computacional com a qual uma pessoa entra em contato de maneira física, perceptiva ou conceitual [Moran, 1981].

Os *websites* são construídos tendo basicamente como recursos utilizados texto, imagens e alguns recursos tecnológicos. As novidades nessa área são constantes e os *websites* tendem a necessitar de atualizações no modo como apresentam as informações. Além disso, percebe-se que os *websites* são a interface por meio da qual as pessoas interagem com a Internet

Nesse contexto, uma Interface Humano-Computador (IHC) de *websites* pode não estar clara ou agradável para determinados usuários e precisar eventualmente ser reavaliada, em

alguns momentos, para atender aos padrões de uso esperados por esses usuários. Isso se deve ao fato de o desenvolvimento das interfaces desses *websites* ser incremental, originando-se de interações com usuários; e ainda devido ao fato de novas ferramentas tecnológicas surgirem constantemente.

Quando ocorrer a avaliação do *website*, no entanto, os problemas de usabilidade já podem estar sendo apresentados aos usuários. Portanto revela-se a necessidade de um programa contínuo de pesquisas e considerações sobre a usabilidade desses ambientes. Pode ser empregada a análise de arquivos de *log* para tal tarefa que inicialmente é uma técnica que não envolve a participação direta dos usuários em ensaios de interação ou preenchimento de formulários. Porém, trata-se de uma atividade de difícil realização, pois nem sempre nos arquivos há dados suficientes ou ainda há muitos dados não diretamente relacionados com a avaliação.

Percebe-se que um importante atributo de qualidade em uma IHC é a sua usabilidade, que estabelece o quanto os sistemas são projetados de forma a serem facilmente utilizados e aprendidos [Nielsen, 2000]. Quem projeta um *website* deve ter a preocupação que os usuários deste *website* consigam realizar o conjunto de atividades disponíveis nele de forma agradável e prática, e com o mínimo de erros possível.

Os projetistas de um *website* sempre trabalham para ter disposto seu conteúdo da melhor maneira possível. Assim, eles consideram que as informações podem ser encontradas com facilidade e que o *website* que desenvolveram pode ser utilizado sem dificuldades. No entanto, os usuários do *website* precisam comprovar esta afirmação.

Ao interagirem com o *website*, os usuários realizam ações com base no contato obtido com a interface desse e interpretam as respostas transmitidas pelo *website* por meio de sua interface. Uma disposição de conteúdo que pareça clara para o projetista pode fazer com que algumas informações não sejam facilmente encontradas pelos usuários, e isso pode provocar insatisfação ou até mesmo a desistência de acesso ao *website*, uma vez que um usuário tende a usar cada vez menos um *website* que não o satisfaça, até que abandone seu uso por completo [Nielsen, 2000]. Por isso, desenvolver um *website* que atenda aos requisitos de usabilidade

estabelecidos deve ser uma das preocupações de seus projetistas.

De acordo com a norma ISO/IEC 9241, a usabilidade pode ser definida como um recorte de métricas derivadas das interações com que cada usuário específico alcança suas metas definidas em ambientes particulares. Essas métricas estão baseadas nas características da eficácia (precisão, perfeição), da eficiência (recursos gastos) e da satisfação (conforto e aceitação) [ISO 9241].

A usabilidade é importante tanto para os usuários, que conseguem ter um melhor desempenho em suas atividades [Jordan, 2002], quanto para os projetistas dos *websites*, que possuem mais visitantes no *website* que elaboraram e, consequentemente, maior visibilidade devido aos acessos que são realizados às informações divulgadas. Portanto, a usabilidade deve ser uma preocupação a ser considerada no ciclo de desenvolvimento dos *websites*.

Nesse contexto, Jakob Nielsen aborda de forma clara a preocupação constante dos projetistas de *websites*: estar no "estado da arte" não é uma característica fácil de ser atingida pelos *websites* por demandar constantes atualizações [Nielsen, 2000], em especial, quando novas técnicas mais atraentes para os usuários são desenvolvidas para apresentar o mesmo conteúdo. Um usuário provavelmente não desperdiça seu tempo em *websites* confusos, lentos ou que não satisfaçam suas necessidades, quando existem outras opções que atendam às suas expectativas [Nielsen, 2012]

Então, percebe-se a validade da afirmação de Nielsen (2000) de que na *Web* é a usabilidade quem dita as regras. Esse fato coloca o poder de decisão para a *Web* nas mãos de quem “clica o botão do mouse”, pois é muito fácil ir para outro lugar, ou mais especificamente, para outro *website* [Nielsen, 2000]. Novos *websites* devem ser constantemente criados para as mais diversas finalidades de forma cada vez mais dinâmica e rápida.

Desta forma, é grande a concorrência entre os projetistas para que o *website* criado por eles seja usado. Ao se observar atentamente, a usabilidade possui uma importância comercial, pois a própria sobrevivência do *website*, seja das informações que ele divulga ou do negócio que ele abriga, fica comprometida se os usuários não se sentirem satisfeitos com seu uso ou

não conseguirem desempenhar corretamente as atividades que ele suporta.

Para se perceber a medida de usabilidade de um *website*, devem ser realizadas uma ou mais avaliações de usabilidade nele. A avaliação de usabilidade pretende relacionar resultados sobre quão satisfatório para os usuários foi o uso do sistema e quais pontos da interação eles consideram positivos de acordo com diversas características [Shneiderman, 1992]. Contudo, é importante o conhecimento do objetivo da avaliação antes que ela seja realizada.

Em Labiutil (2012) são citados alguns possíveis objetivos a serem atingidos pelas avaliações:

- relacionar problemas efetivos de usabilidade durante a interação;
- calcular métricas objetivas sobre a eficácia, eficiência e produtividade do usuário na interação com o sistema;
- prever dificuldades de aprendizado na operação do sistema;
- prever os tempos de execução de tarefas informatizadas;
- conhecer a opinião do usuário em relação ao sistema;

Existem diversas maneiras propostas para avaliação da usabilidade. Porém, durante as avaliações de usabilidade, os dados coletados são qualitativos (descrições que qualifiquem a usabilidade) ou quantitativos (medidas que a quantifiquem) [Morandini, 2002]. Em geral, métodos quantitativos produzem números; enquanto os qualitativos produzem descrições por meio de palavras, vídeos, sons ou imagens. Dados quantitativos explicam melhor o “porquê” e o “como”, enquanto dados qualitativos explicam melhor o “quê”, o “quem” e o “quando” [CCP, 2013].

A avaliação qualitativa foca a satisfação das necessidades dos usuários com relação ao uso dos *websites*, no que se refere aos padrões esperados para a categoria em que os *websites* se encontram. A comparação entre o que é observado e os padrões esperados para usabilidade conduzem ao diagnóstico, porém tais métodos demandam mais tempo e recursos.

Enquanto na avaliação qualitativa são empregados questionários ou *checklists*, na

quantitativa espera-se a coleta de informações que permitam a produção de métricas a serem analisadas. Particularmente, a avaliação quantitativa mede as interações entre os usuários e componentes do *website*, analisando algumas características desejáveis e comparando as medidas com padrões esperados de comportamento.

Em geral, dentre as metodologias que suportam avaliações qualitativas da usabilidade de sistemas interativos, as avaliações heurísticas são reveladoras de uma grande quantidade e variedade de diagnósticos de problemas (quanto ao nível de severidade). Muitas vezes, em sua aplicação, os avaliadores se valem de grades de análise que os orientam quanto aos aspectos a verificar na interface e no seu contexto de uso. Dentre as grades de heurísticas conhecidas, destacam-se os critérios ergonômicos definidos por Scapin e Bastien (1997). Tais critérios são apresentados como uma forma de serem fornecidas diretrizes para que os projetistas elaborem interfaces ergonomicamente corretas.

Uma metodologia para realizar avaliações de usabilidade que proporcionem resultados quantitativos é o emprego dos testes de usabilidade, que são técnicas baseadas na observação de interações reais. Eles conseguem identificar claramente problemas reais que se manifestam. Eles devem ser usados para comprovar o diagnóstico obtido por uma avaliação qualitativa, produzidos por avaliações heurísticas ou pelo uso de listas de verificação (*checklists*).

Segundo Morandini (2002), taxa é uma medida intermediária da qualidade da interação (taxa de sucesso, de desistência, de cancelamento, etc) e métrica representa uma medida de usabilidade, conforme estabelecida pela norma ISO 9241:11 (métrica de eficácia e de eficiência, basicamente).

Sauro (2011) apresenta algumas métricas de usabilidade que podem ser obtidas por meio desse trabalho, a saber: tarefas completadas com sucesso (conhecida como métrica fundamental de usabilidade); quantos foram os problemas de usabilidade encontrados; duração total para a realização de cada tarefa (real medida de eficiência e produtividade); erros, ou seja, cada ação não intencionada, deslize, engano ou omissão que um usuário fizer; e quantidade de visualizações de uma página.

Este trabalho foca avaliação quantitativa, remota e assíncrona de usabilidade, de modo que pode obter dados que não necessitem do contato direto com o usuário do *website*. Portanto, métricas como tarefas completadas, duração total, erros, e quantidade de visualizações são passíveis de serem calculadas pela abordagem apresentada por este trabalho.

Ao se realizar uma avaliação de usabilidade, métricas de usabilidade como as relacionadas acima devem ser obtidas por meio de testes de usabilidade. Durante esses testes, usuários típicos realizam tarefas típicas em uma plataforma típica. No entanto, alguns cuidados precisam ser tomados ao serem aplicados testes de usabilidade: é difícil e custoso reunir um grupo grande de usuários em um laboratório para testar um determinado *website;* precisa ficar claro para os usuários que o foco do teste é o *website* e não eles próprios; e os avaliadores devem também estar bem treinados no que estão fazendo.

Uma abordagem prática e de baixo custo é o emprego dos arquivos de *log* (arquivo de texto onde ficam armazenadas as interações ocorridas por meio de um *website*.) para apoiar a coleta e análise de métricas de usabilidade, técnica inserida nos métodos de avaliação remota de usabilidade [Ivory & Hearst, 2001]. Essa técnica possui algumas vantagens, não só em termos de custo, como também no modo de avaliação. Por meio desta abordagem se conseguem obter algumas métricas para avaliar a usabilidade de *websites.*

A seção a seguir apresenta a justificativa e objetivos deste trabalho, enfocando o tema das tarefas e das taxas e métricas.

1.2 Justificativa e Objetivos

A avaliação de usabilidade de *websites* é o principal foco deste trabalho e, de forma mais específica, a avaliação remota de usabilidade. O objetivo deste trabalho é descrever como os arquivos de *log* fornecem dados significativos para a obtenção de taxas e métricas objetivas que quantifiquem a usabilidade de interfaces, ao serem realizadas algumas tarefas determinadas. Desta forma, podem ser possibilitadas avaliações confiáveis, objetivas, rápidas e de baixo custo.

Para que seja possível a avaliação de usabilidade de um *website*, é necessário o conhecimento de todas as páginas web que façam parte do conjunto de páginas *web* a que os usuários do *website* tenham acesso. Além disso, talvez existam outros recursos *web*, como vídeos, fotos, áudios entre outros, que estejam contidos nas páginas *web* e que também sejam acessados pelos usuários. Doravante será usado o termo recurso *web* para se referir tanto às páginas *web* quanto aos demais recursos *web*.

Os recursos *web*, quando acessados numa ordem específica, a partir de um recurso inicial, passando por alguns recursos, e chegando a um recurso final, compõem uma tarefa. A proposta deste trabalho é basear a avaliação de usabilidade de um *website* no conjunto de tarefas relativas ao *website* em questão.

O conjunto de tarefas de um *website* representa a visão do projetista ou dos responsáveis pelo *website* dos passos que um usuário deve seguir para atingir os recursos *web* correspondentes aos últimos passos de cada tarefa e, desta forma, realizando tarefas específicas. Isso será denominado modelo de acesso para a tarefa.

Quanto mais abrangente for o conjunto de tarefas elencado, melhor será descrito o *website* e mais precisas serão as taxas calculadas pela ferramenta desenvolvida por este trabalho. Além disso, para uma determinada tarefa, não existe necessariamente apenas um caminho. Quanto mais caminhos o projetista identificar para a tarefa, maiores as possibilidades de um desses caminhos aparecer no arquivo de *log*.

As informações coletadas dos usos reais do *website* ao serem realizadas algumas tarefas estabelecidas são comparadas com modelos de acesso para a realização destas tarefas. Desta comparação devem ser identificados os recursos *web* percorridos, as tarefas completadas, as tarefas abandonadas, dentre outras situações. A partir deste levantamento, são obtidas taxas e calculadas métricas da usabilidade do *website*. Estes valores devem ser confrontados com outras avaliações realizadas para o mesmo *website* ou para *websites* na mesma categoria do avaliado.

Dentre as informações que são coletadas, encontram-se:

- quantidade de acessos ao *website*;
- quantidade de acessos a recursos *web* ou atividades específicas;
- tempo de acesso para o *website* e para cada recurso *web* que o compõe;
- quantidade de mensagens de erro apresentadas por recurso *web* (e geral);
- quantidade de páginas *web* de ajuda consultadas.

Algumas taxas podem ser obtidas a partir destas informações coletadas, as quais auxiliam na composição das características de usabilidade do *website*:

- taxa de pedidos de ajuda;
- taxa de mensagens de erro;
- taxa de repetições de tarefas (i.e., uma proporção que apresente quantas vezes a mesma tarefa, tal como o preenchimento de um formulário, teve que ser repetida);
- taxa de desistências de acesso;
- taxa de cancelamentos de acesso;

O método de avaliação proposto neste trabalho deve interferir o mínimo possível nas interações do usuário com o *website*. Os *websites* são classificados em: informativos ou transacionais. Os informativos são aqueles que permitem basicamente consulta a informações; e os transacionais são aqueles em que existe algum tipo transação como comércio eletrônico ou acesso a banco. Tal método deve funcionar com ambos os tipos de *website*, desde que tenham tarefas bem delineadas, com início e fim bem definidos; e não deve gerar ruídos para os usuários, pois é executado remota e assincronamente.

As informações usadas como base para a obtenção das taxas e para o cálculo das métricas se encontram armazenadas nos arquivos de *log* dos servidores *web*, permitindo ao usuário interagir normalmente com o *website* em seu local rotineiro de trabalho. Consequentemente, o foco da avaliação de usabilidade empregada não é desviado do *website* para o usuário.

O método de avaliação apresentado neste trabalho não pretende substituir outros métodos de avaliação, em especial os métodos qualitativos, mas pode servir como avaliação preliminar. Assim, são identificadas as partes do *website* que poderiam ser posteriormente avaliadas por outros métodos mais custosos e demorados. Desta forma, pode-se focar a avaliação em possíveis indícios de problemas de usabilidade.

A seguir são apresentados os objetivos específicos, detalhando assim, o que se pretende com sua realização. Além disso, a metodologia adotada é mencionada

1.3 Objetivos Específicos e Metodologia

Os objetivos específicos deste trabalho estão relacionados a seguir. Eles foram definidos de modo a auxiliar na obtenção dos objetivos gerais delineados acima.

- Propôr um método para avaliar remota e automatizadamente a usabilidade de *websites*, por meio de uma ferramenta que analisa os arquivos de *log* relativos a estes *websites*.
- Produzir as seguintes métricas de usabilidade com base em arquivos de *log* de *websites* e em tarefas específicas a serem realizadas por meio desses *websites*:
  - tarefas completadas;
  - tempo gasto nas tarefas completadas.
- Sugerir alternativas para evitar avaliações demoradas e custosas em tarefas que não sejam candidatas a problemas de usabilidade.
- Indicar possíveis problemas de usabilidade ao serem realizadas determinadas tarefas em um *website*, com base nas métricas calculadas para este *website*.

Para o funcionamento da ferramenta desenvolvida nesse trabalho são necessárias as seguintes etapas:

1) Separar trecho do arquivo de *log* por período a ser analisado;
2) Filtrar o trecho separado para alguns endereços IP;
3) Fazer upload do arquivo de *log* separado e filtrado para o servidor da ferramenta;
4) Cadastrar *website* e suas informações básicas (nome do arquivo de log e dados do usuário);
5) Mapear tarefas correspondentes ao *website*;
6) Cadastrar tarefas pelo uso da ferramenta; e
7) Realizar avaliação do *website* com a ferramenta.

A metodologia adotada consistiu em coletar os arquivos de *log* do servidor, adaptar um recurso que analisasse as informações contidas no arquivo de *log*, elaborar uma ferramenta que usasse o recurso acima e calculasse taxas e métricas de usabilidade, e fazer a ferramenta funcionar para os arquivos de *log* coletados.

No capítulo que trata sobre o desenvolvimento da ferramenta, existe uma explicação mais detalhada sobre esta sequência de etapas.

1.4 Limitações do Trabalho

Para apresentar adequadamente os objetivos deste trabalho, são apresentadas, a seguir, algumas restrições e limitações do mesmo, as quais devem ser levadas em consideração ao se avaliar seu resultado geral.

É importante destacar que os arquivos de *log* não armazenam todas as informações a respeito das interações do usuário com o *website*. Informações como cliques com o "mouse", foco dos olhos, intenção de procura, atividade que se quer realizar, dentre outras não estão relacionadas nos arquivos de *log*. É possível capturar destes arquivos apenas os recursos *web* visitados pelos usuários e os parâmetros de acesso destas visitas (por exemplo, como opções da página *web*, hora do acesso, navegador usado).

Considere-se ainda que o *website* deve ser avaliado em seu contexto regular de uso, sem a necessidade de qualquer alteração no *website*. Isso facilita o emprego da ferramenta, pois o contato necessário pode se restringir apenas ao administrador do servidor *web* em que o *website* está hospedado.

No próximo capítulo, será apresentada a revisão bibliográfica. Em especial, serão destacados aspectos relacionados à usabilidade, abordando seus atributos, métodos para sua avaliação, e a automatização do processo de avaliação. Além disso, serão abordadas as tecnologias envolvidas na ação de navegação pela *Web* e sua relação com usabilidade.

2 *Revisão Bibliográfica*

Este capítulo apresenta um levantamento de conceitos que foram abordados na literatura, relacionados ao tema da usabilidade, focando os métodos de avaliação de usabilidade e destacando a análise de arquivos de *log*. Por fim, são abordados alguns tópicos que auxiliam o entendimento do ambiente onde os *websites* funcionam e que mostram as tecnologias usadas neste trabalho.

2.1 Usabilidade

Usabilidade é o termo que define o quanto a interface de um sistema é fácil de ser utilizada e aprendida [Nielsen, 2000]. Ela é uma qualidade desejável e muito importante para um sistema, porque os usuários tendem a não usar o sistema, se não conseguirem atingir seus objetivos com ele.

Um sistema que possui o atributo da usabilidade possibilita aos seus usuários a exploração de seu conteúdo, sem deixá-los desorientados ou frustrados por não conseguirem realizar satisfatoriamente esta tarefa exploratória [Garzotto et al., 1998]. Considere-se ainda o agravante que, normalmente, não se conhece quem serão os usuários de um sistema. Portanto deve existir maior preocupação com relação à usabilidade dos sistemas a serem desenvolvidos.

Ademais, os usuários não estão dispostos a gastar tempo aprendendo a interagir com um sistema [Nielsen, 2000]. Na impossibilidade ou dificuldade de se conseguir fazer isso, vão acabar procurando outros sistemas similares e mais fáceis de usar [Nielsen, 2012]. E um sistema que é pouco usado não compensa o esforço realizado para colocá-lo em funcionamento.

Ampliando esta idéia, segundo a norma ISO 9241 da Organização Internacional de Padronização, parafraseada por Ivory & Hearst (2001), “usabilidade é a extensão na qual um

sistema de computador permite aos usuários, em um dado contexto de uso, atingir objetivos específicos de maneira efetiva e eficiente, enquanto promove sentimentos de satisfação".

Essa Norma traz ainda algumas definições [Brajnik, 2000] [ISO 9241]:

- eficácia é a precisão e a perfeição com que as metas desejadas em um ambiente são alcançadas pelos usuários;
- eficiência são os recursos gastos em relação à precisão e perfeição de metas alcançadas; e
- satisfação é o quanto os usuários estão confortáveis e aceitam o sistema.

Recentemente ocorreu uma alteração no padrão de qualidade ISO 9126, o qual foi substituído pelo padrão ISO 25010. A definição de usabilidade sofreu um ajuste na maneira de expressar o conceito. Usabilidade ficou definida como uma medida pela qual um produto pode ser usado por usuários específicos para alcançar metas específicas com eficácia, eficiência e satisfação em um contexto específico de uso. [ISO25010, 2012]

Ao serem adotadas metodologias para medir os aspectos de usabilidade das interfaces de usuário e serem encontrados problemas nelas, está sendo realizada uma atividade de avaliação da usabilidade. A avaliação de usabilidade tem sido cada vez mais utilizada no projeto de *websites*.

Embora não exista um consenso sobre uma metodologia única a ser empregada para se realizar qualquer avaliação de usabilidade de IHCs, existem diferentes tipos de avaliação empregados nas diversas fases do desenvolvimento de uma interface [Rocha e Barnauskas, 2000].

Em uma avaliação de usabilidade devem ser levados em conta aspectos relacionados à realização de tarefas pelas quais os usuários pretendem atingir seus objetivos. A facilidade de uso de um sistema estará diretamente relacionada ao desempenho do usuário, seja em condições controladas ou em casos de uso real, considerando que as condições controladas devem estar o mais próximas do uso real do sistema [Bevan et al., 1991].

Por fim, é importante notar que uma avaliação deve ser uma fase embutida no próprio processo de desenvolvimento do sistema, não devendo ser negligenciada, pois pode ajudar a melhorar alguns aspectos do sistema em questão [Fernandez & Hussmann, 2009] [Brajnik, 2000].

Bevan et al. (1991) afirmam que usabilidade está relacionada à interação do usuário com o sistema e só pode ser medida adequadamente através do desempenho, da satisfação e da aceitação do usuário com relação ao sistema em questão. Esta afirmação está diretamente relacionada com a definição de usabilidade citada por Ivory & Hearst (2001) da norma ISO 9241.

Brajnik (2000) deixa claro que um *website* é um sistema interativo, pois ao menos os usuários finais e os *webdesigners* têm interação com ele. Ainda segundo Bevan et al. (1991), mudanças nas características do sistema afetarão a usabilidade. Um sistema não pode ser classificado como usável ou não, mas contém atributos que determinam a usabilidade dele para um usuário, uma tarefa e um ambiente em particular. Em outras palavras, usabilidade depende do contexto.

Durante o desenvolvimento de um sistema em que a usabilidade é pretendida, deve-se tentar encontrar o equilíbrio entre a complexidade da estrutura do sistema e os requisitos de usabilidade. [Garzotto et al., 1998]

Na próxima seção o termo usabilidade será mais bem descrito ao se apresentar uma classificação de seus atributos.

2.2 Atributos de Usabilidade

Alguns atributos de usabilidade comuns são citados a seguir, acompanhados por uma pergunta motivadora para ajudar a descobrir se a IHC possui ou não cada atributo [Cleardesign 2011] [ISO 9126]:

- Facilidade de uso: É fácil para os usuários aprenderem como usar seu *website*?

- Eficiente para usar: Os usuários conseguem encontrar e obter informação rapidamente e efetivamente do *website*?
- Memorizável: Um usuário que retorne ao seu *website* conseguiu reter alguma experiência de aprendizado por ter visitado anteriormente?
- Erros dos usuários: É uma falha do programador ou projetista do *website* quando o usuário experimenta um erro?

No padrão ISO 25010, o novo modelo definido para usabilidade, segurança e compatibilidade foram adicionadas como características de qualidade. Algumas subcaracterísticas foram adicionadas ao modelo e um número delas foi renomeado para termos considerados mais apropriados.

No ISO 25010, o modelo de qualidade do software descreve medidas internas e externas de qualidade de software. Medidas internas descrevem um conjunto de atributos internos e estáticos que podem ser medidos. Já as medidas externas focam mais em software como uma caixa preta e descrevem atributos externos que podem ser medidos. Além disso, é definido um modelo de qualidade de software em uso. [ISO25010, 2012]

Jordan (2002) faz uma avaliação das características do usuário e seus efeitos na usabilidade. Ele comenta sobre a experiência do usuário, o conhecimento do usuário sobre o domínio do sistema, o histórico cultural do usuário, uma possível limitação do usuário, e ainda o gênero e a idade do usuário.

No entanto, ressalta que tais características deverão mudar pouco com o tempo, uma vez que o desempenho do usuário com um sistema tende a melhorar em relação a tarefas que sejam realizadas com repetição ao longo do tempo. Apesar disso, algumas já serão bem realizadas e outras serão melhor realizadas após um tempo sem serem desempenhadas. Desta forma, percebemos que a discussão é ampliada.

Apresentam-se abaixo os componentes descritos por Jordan (2002):

1. **Previsibilidade, Capacidade de Suposição** (*Guessability*): “A eficácia e eficiência e

satisfação com a qual usuários específicos completam tarefas especificadas em um produto particular pela primeira vez." [Jordan, 2002] Refere-se ao desempenho de uma tarefa pela primeira vez; quanto mais fácil deduzir como fazê-la, maior a previsibilidade. Como dificilmente os usuários serão treinados para utilizar um novo *website*, esta característica é muito importante neste contexto. Ademais, previsibilidade pode se referir ao realizar de uma nova tarefa com base em uma tarefa já realizada, diminuindo assim a curva de aprendizagem para o *website* completo.

2. **Facilidade de ser Aprendido** (*Learnabillity*): "A eficácia, eficiência e satisfação com a qual usuários específicos atingem um nível de desempenho competente em tarefas específicadas em um produto, tendo já previamente realizado aquelas tarefas." [Jordan, 2002] Se o usuário conseguiu ter uma boa lembrança de como realizar a tarefa após a ter completado anteriormente, o produto mostrou-se ser grandemente aprensível. Esta característica se aplica muito mais a interfaces gráficas do que a interfaces linha de comando, portanto faz muito sentido para *websites*.
3. **Desempenho de Usuário Experiente** *(Experienced User Performance):* "A eficácia, eficiência e satisfação com a qual usuários experientes específicos atingem tarefas específicas em um particular produto." [Jordan, 2002] Um usuário que já usou o produto por diversas vezes volta a usar sem perder desempenho. No caso de *websites* não é um atributo significativo, pois a tendência é que ocorram visitas rápidas ao *website* e por um curto período de tempo.
4. **Potencial do Sistema** *(System Potencial):* "O nível ótimo de eficácia, eficiência e satisfação com o qual seria possível completar tarefas específicas com o produto." [Jordan, 2002] Para esta característica fazer sentido para um *website*, o usuário precisa gastar tempo com ele, explorando as diversas formas de realizar uma determinada tarefa para ver qual delas é a melhor possível. Com o tempo de uso ou com algum aconselhamento isso pode ser atingido, mas a maioria dos usuários realizará as tarefas sempre do mesmo jeito. Para usuários de *website* esta realidade também ocorre, o que significa que essa característica não é tão significativa.

5. **Reusabilidade** *(Re-usability):* "A eficácia, eficiência e satisfação com a qual usuários específicos atingem tarefas específicas em um particular produto, após um período comparativamente longo sem desempenhar tais tarefas." [Jordan, 2002] Refere-se à queda de desempenho após o usuário não usar o produto por algum tempo. Note-se que, no primeiro período de uso, o usuário pode não ter necessitado realizar tarefas que no segundo período necessitará realizar. Esta característica para usuários de *websites* vai variar de acordo com o assunto ou tema do *website*.

Segundo Garzotto et al. (1998) os atributos de usabilidade são qualidades esperadas de um sistema bem projetado e bem implementado. Os autores se baseiam na facilidade de um usuário novato aprender a usar um sistema e na possibilidade de um usuário atingir seus objetivos (eficiência), para suas definições.

A facilidade de aprender o uso é decomposta em consistência (elementos conceitualmente similares são tratados visualmente de forma similar) e previsibilidade (o usuário consegue elaborar um modelo previsível de como o conteúdo está organizado e como o sistema se comporta). Eficiência é descrita por acessibilidade (quão fácil o usuário localiza a informação que está procurando) e orientação (habilidade de o usuário entender sua localização no sistema e os movimentos que faz) [Garzotto et al., 1998].

Segundo a análise feita por Fernandez & Hussmann (2009), as variáveis mais frequentemente usadas para medir os atributos de usabilidade foram tempo (para uma determinada tarefa), eventos (cliques de botão) e navegação (número de passos e quais componentes são acessados para se atingir um destino).

Se compararmos tal análise com os atributos de usabilidade acima, vemos que há grande correlação entre ambos. Com relação ao tempo, se o usuário localiza rapidamente o que estiver procurando, gastará menos tempo em cada etapa; quanto a eventos, se o usuário associa elementos similares e aprende o uso do sistema facilmente, dará menos cliques para percorrer o mesmo; sobre a navegação, se o usuário está orientado e prevê o comportamento do sistema, atingirá seu destino em menos passos.

Hartson et al. (2000) nos apontam o problema do critério de avaliação. São estabelecidos critérios últimos para um sistema, mas deve ser percebida a relevância destes critérios para as aplicações do sistema em uma situação real, para se adequarem os critérios últimos aos critérios reais. Esses autores relacionam medidas de desempenho, ao aplicarem critérios reais. Eles descrevem que os avaliadores de um método de avaliação de usabilidade querem que os resultados dos métodos sejam consistentes, independente de quem estiver desempenhando a avaliação; completos, ou seja, que a maior quantidade possível de problemas de usabilidade existentes seja encontrada; corretos, onde apenas problemas que realmente existem sejam encontrados. Estas características, segundo eles, parecem ser as principais medidas de critério que os pesquisadores deveriam continuar a investigar.

A seção seguinte discorrerá sobre avaliação de usabilidade, mostrando seu desenvolvimento ao longo dos anos, até descrever o que é praticado atualmente.

2.3 Avaliação de Usabilidade

Neste trabalho haverá ênfase na avaliação heurística de usabilidade, pois este é um dos focos do método que está sendo proposto como fruto deste trabalho. Veremos sobre heurísticas mais para frente nesta seção. A seguir aborda-se um histórico sobre avaliação de usabilidade, mostrando as preocupações que motivaram os avaliadores ao longo das últimas décadas.

Hartson et al. (2000) nos fornecem este histórico de como a avaliação de usabilidade vem sendo conduzida nos últimos anos. Na década de 80, os testes de avaliação de usabilidade feitos em laboratório eram o principal meio de examinar modificações nas interfaces. Os usuários que participavam dos testes ajudavam na avaliação da velocidade, acurácia e erros, além de fornecer opiniões próprias em certos casos.

Na década seguinte, para reduzir os custos de testes e para antecipar o momento em que os testes aconteciam, os desenvolvedores passaram a conduzir métodos de inspeção com especialistas analisando passo a passo as heurísticas ou a usabilidade.

Scholtz (2004) confirma esta idéia, mostrando que métodos como avaliação por heurísticas, percurso cognitivo e usabilidade remota surgiram na década de 90. São justamente os métodos onde entra a participação de um avaliador.

O autor ainda mostra que a principal desvantagem de avaliações focadas no usuário é o custo e afirma que o método de passo a passo cognitivo é o único que deveria ser usado em estágios iniciais de desenvolvimento do sistema.

Ainda segundo Hartson et al. (2000), um processo intermediário passou a surgir em seguida, chamado avaliação formativa, a qual procura resolver problemas de usabilidade em um momento intermediário, quando o projeto não está tão imaturo, mas quando ainda não está finalizado. Para facilitar este processo, métodos de inspeção combinando a participação dos usuários e dos especialistas foram desenvolvidos.

De qualquer maneira, independente de como se conduz a avaliação de usabilidade, como em qualquer processo de avaliação são necessárias as seguintes etapas: coleta de dados, interpretação destes dados para identificar problemas e sugestão de soluções. Os métodos existentes para avaliação de usabilidade fazem uma ou mais destas etapas automaticamente, caso necessário [Ivory & Hearst, 2001].

Uma avaliação realizada com dados extraídos de forma aleatória busca conseguir informações que representem melhor a realidade de visitação das páginas avaliadas, permitindo inclusive que a avaliação seja feita de novo com outros dados, para validar os resultados obtidos, sem custo adicional.

A avaliação de usabilidade pode ser realizada remotamente também. Um dos propositores desta perspectiva observa que, para tanto, basta haver uma separação relacionada a espaço ou a tempo entre o avaliador e o usuário. Isso permite que os usuários estejam em seu ambiente de trabalho [Castillo et al., 1997].

Os autores comentam que a tecnologia à nossa disposição permite que este processo seja realizado sem prejuízo e incluem uma avaliação em laboratório como um *benchmark* para os resultados obtidos. Consideram como atributos dos métodos remotos: tipos de usuários envolvidos, tempo de avaliação, localização dos usuários, tipos de dados coletados, tipos de

tarefas, custos e qualidade para coletar e analisar os dados.

Eles concluem que, apesar de ser um método remoto, a abordagem que pareceu melhor para eles foi a que pergunta ao usuário sobre problemas encontrados. Neste caso, percebemos que a vantagem de o método ser remoto está na flexibilidade, pois a avaliação de usabilidade continua focada no usuário, como já foi abordado por Bevan et al. (1991).

Na seção a seguir, será abordada uma classificação dos métodos de avaliação de usabilidade.

2.3.1 Métodos de Avaliação de Usabilidade

Segundo Hartson et al. (2000) independente do método usado, o objetivo dos métodos de avaliação de usabilidade é essencialmente o mesmo, a saber, produzir descrição de problemas de usabilidade observados ou detectados, com o propósito de analisar e projetar novamente a interface.

Apesar de os métodos de avaliação convergirem para um mesmo objetivo, Ivory & Hearst (2001) propõem uma taxonomia de métodos de avaliação de usabilidade, onde estabelecem quatro dimensões para agrupar os métodos:

a) Classe do Método: Descreve o tipo de avaliação conduzida em um alto nível.

- teste, onde um avaliador examina usuários a interagirem com uma interface para determinar problemas de usabilidade;
- inspeção, onde um avaliador faz uso de critérios e heurísticas para identificar problemas potenciais de usabilidade em uma interface;
- questionário, onde o usuário dá retorno sobre uma interface através de entrevista ou avaliação;
- modelagem analítica, onde um avaliador emprega modelos de usuário e de interface para gerar predições de usabilidade; e
- simulação, em que um avaliador emprega modelos de usuário e de interface para mimetizar interações do usuário com a interface e relatar resultados desta interação.

b) Tipo do Método: Descreve como a avaliação é conduzida em cada classe do método.

- Testing (Classe do Método = Teste)
  - Protocolo *Thinking-Aloud* - usuário fala durante o teste;
  - Protocolo *Question-Asking* - quem testa faz perguntas ao usuário;
  - Método *Shadowing* - o *expert* explica ações do usuário para quem testa;
  - Método *Coaching* - usuário pode perguntar a um *expert*;
  - Método *Teaching* - usuários experientes ensinam usuários novatos;
  - *Co-discovery Learning* - dois usuários colaboram;
  - Medição de Desempenho - quem testa registra dados de uso durante teste;
  - Análise de Arquivos de *Log* - quem testa analisa dados de uso;
  - Teste Retrospectivo - quem testa revê vídeo com usuário;
  - Teste Remoto - quem testa e o usuário não estão no mesmo lugar durante o teste;
- Inspection (Classe do Método = Inspeção)
  - *Guideline Review* - um *expert* verifica conformidade com uma *guideline*;
  - *Cognitive Walkthrough* - um *expert* simula a resolução do problema do usuário;
  - *Pluralistic Walkthrough* – diversas pessoas conduzem o *Cognitive Walkthrough*;
  - Avaliação Heurística - um *expert* identifica violações de heurísticas;
  - Inspeção Baseada em Perspectiva - um *expert* conduz de perto avaliação heurística focada;
  - Inspeção de Características - um *expert* avalia características do produto;
  - Inspeção de Usabilidade Formal - um *expert* conduz avaliação heurística formal;
  - Inspeção de Consistência - um *expert* verifica consistência entre produtos;
  - Inspeção de Padrões - um *expert* verifica conformidade com padrões;
- Inquiry (Classe do Método = Questionário)
  - Questionário Contextual - entrevistador questiona usuário em seu ambiente;
  - Observação de Campo - entrevistador observa o uso do sistema no ambiente do usuário

- *Focus Groups* - diversos usuários participam de uma sessão de debate;
- Entrevistas - um usuário participa de uma sessão de debate;
- *Surveys* - um entrevistador faz perguntas específicas para o usuário
- Questionários - usuário provê respostas para questões específicas;
- *Self-Reporting Logs* - usuário registra operações da interface;
- Instantâneos da Tela - usuário captura instantâneos da tela;
- *Feedback* do usuário - usuário envia comentários;

- Analytical Modeling (Classe do Método = Modelagem Analítica)
  - Análise GOMS - predizer execução e tempo do usuário;
  - *Análise* UIDE – análise GOMS conduzida com UIDE;
  - Análise de Tarefa Cognitiva - predizer problemas de usabilidade;
  - Análise Ambiente-Tarefa – avaliar mapeamento dos objetivos do usuário em tarefas na interface;
  - Análise de Conhecimento - predizer aprendizado;
  - Análise de *Design* - avaliar a complexidade do *design*;
  - Modelos de Usuário Programáveis - escrever programas que agem como usuário;
- Simulation (Classe do Método = Simulação)
  - Modelagem do Processo de Informação - mimetizar ações do usuário;
  - Modelagem por Rede de Petri - mimetizar ações do usuário pelos dados de uso;
  - Modelagem por Algoritmos Genéticos - mimetizar ações de um usuário novato;
  - Modelagem por Rastro de Informação - mimetizar a navegação em um *website*;

c) Tipo de Automatização: Descreve o aspecto da avaliação que é automatizado.

- nenhuma automatização dos aspectos da avaliação, pois o avaliador conduz todos os aspectos do método de avaliação;
- automatização da captura de dados;
- automatização da análise para identificar problemas potenciais de usabilidade; e
- automatização crítica, onde melhorias são sugeridas.

d) Nível de Esforço: Descreve o tipo de esforço humano requerido para executar o método.

- esforço mínimo; desenvolvimento de modelo de usuário ou de interface da parte do avaliador para empregar o método;
- uso informal, onde é requerido que se completem tarefas escolhidas livremente; e
- uso formal, onde é requerido que se completem tarefas especificamente selecionadas.

Pela taxonomia apresentada por Ivory e Hearst (2001), a ferramenta desenvolvida neste trabalho se encaixa nas dimensões apresentadas por eles da seguinte forma:

- Classe do Método: teste (observação da interação com interface indiretamente)
- Tipo do Método: teste (análise de arquivos de log)
- Tipo de Automatização: da captura de dados
- Nível de Esforço Humano Exigido: mínimo (para executar o método)

Após a coleta de informações que medem a usabilidade do sistema, um estudo mais detalhado tentará diagnosticar problemas de usabilidade. O aspecto relacionado à classe dos métodos na taxonomia apresentada por Ivory & Hearst (2001) é abordado por Bevan et al. (1991), ao mostrarem o uso das seguintes técnicas para detectar problemas de usabilidade: opinião de especialistas sobre o problema observados, análise detalhada de interações dos usuários, e análise das interações dos usuários ou de especialistas por lista de verificações.

A abordagem de Bevan et al. (1991) pode ser detalhada como segue: na análise das interações dos usuários, os usuários explicam as dificuldades encontradas em atingir seus objetivos; na análise das interações dos usuários por lista de verificações, os usuários respondem sobre a aceitação de vários aspectos da interface e identificam problemas; e na análise da interação de um especialista por lista de verificações, o especialista abordará tarefas pré-definidas do sistema.

De uma maneira mais resumida, Brajnik (2000) concorda com a idéia exposta acima ao relatar que as avaliações de usabilidade de um sistema adotam os métodos de inspeção,

conduzidos por avaliadores especialistas, e os testes de usuários.

No que se refere aos métodos de inspeção, uma técnica bastante usada são as avaliações heurísticas, onde os avaliadores usam padrões de qualidade para julgar o valor de usabilidade de uma interface e de seus componentes [Scholtz, 2004]. Seu uso pode levar os projetistas a elaborarem interfaces ergonomicamente corretas, evitando que o usuário tenha dificuldades ou erros de utilização, sinta falta de retorno do sistema ou simplesmente desista de utilizar o sistema. Apesar de muito utilizados, os métodos de avaliação centrados nos especialistas, segundo Scholtz (2004), também apresentam desvantagens.

A avaliação heurística pode ser vista como um modelo de baixo custo na Engenharia da Usabilidade. Assim, pode ser considerado como um método barato, fácil e rápido de avaliar um projeto de interface do usuário, uma vez que os custos de uma avaliação de usabilidade utilizando outros métodos é algo consideravelmente caro, o que acaba muitas vezes sendo deixado de lado no processo de desenvolvimento de um sistema que será utilizado por pessoas.

A avaliação heurística é o mais popular dos métodos de inspeção de usabilidade. O objetivo da avaliação heurística é encontrar os problemas de usabilidade no projeto para que eles possam ser atendidos como parte de um processo de design interativo [Nielsen, 2012].

As próximas duas seções destacam dois métodos de especialistas em IHC: as heurísticas de Nielsen e os critérios ergonômicos de Scapin e Bastien.

2.3.2 Heurísticas Propostas por Jakob Nielsen

Nielsen (2010) foi um dos pioneiros na área de usabilidade e formulou heurísticas de usabilidade, brevemente descritas a seguir:

Tabela 1 *- Heurísticas de Nielsen [Nielsen, 2010]*

1. *Visibilidade do Estado do Sistema:* deve haver retorno apropriado para o usuário em tempo razoável;
2. *Compatibilidade entre o Sistema e o Mundo Real:* as informações ao usuário devem ser compreensíveis na linguagem dele;

3. *Controle e Liberdade do Usuário:* o usuário deve conseguir alternar entre o estado anterior e o próximo sem dificuldades;
4. *Consistência e Padrão:* palavras, ações ou situações devem significar sempre a mesma coisa em diferentes contextos;
5. *Prevenção de Erros:* melhor que relatar bem um erro e evitar que ele aconteça;
6. *Reconhecimento:* os objetos, ações e opções devem estar visíveis entre etapas e as instruções devem estar acessíveis;
7. *Flexibilidade e Eficiência de Uso:* os usuários experientes devem poder acelerar suas ações sem que os novatos sejam afetados;
8. *Projeto Estético:* as informações devem ser diretas, para que o foco não se perca;
9. *Ajuda ao Reconhecimento, Diagnóstico e Recuperação de Erros:* mensagens de erro devem ser claras, indicando problema e sugerindo solução;
10. *Ajuda e Documentação:* documentação simples e objetiva deve estar disponível.

As heurísticas descritas acima são regras gerais para descrever propriedades comuns e interfaces utilizáveis e devem ser consideradas para todos os elementos.

2.3.3 Critérios Ergonômicos Propostos por Scapin e Bastien

Scapin & Bastien (1997) afirmam que os Critérios Ergonômicos representam as características mínimas que uma IHC deve ter. A seguir, estão relacionados os critérios principais levantados pelos autores:

Tabela 2 *- Critérios Ergonômicos por Scapin e Bastien [Scapin & Bastien, 1997]*

1. **Condução** (*Guidance*): Mecanismos para advertir, orientar, informar, instruir e guiar os usuários durante suas interações.
2. **Sobrecarga de Trabalho** (*Workload*): Relacionado aos elementos da IHC que servem para reduzir a carga de percepção do usuário e melhorar a eficiência de diálogo com o usuário.
3. **Controle Explícito** (*Explicit Control*): Refere-se ao processamento do sistema por meio de ações explícitas do usuário, bem como ao controle pelos usuários sobre o processamento dessas ações.
4. **Adaptabilidade** (*Adaptability*): Refere-se à capacidade do sistema se comportar de

acordo com o contexto e com as necessidades do usuário.

5. **Gerenciamento de Erros** (*Error Management*): Refere-se aos mecanismos do sistema para reduzir erros e se recuperar deles quando ocorrerem.
6. **Consistência** (*Consistency*): Refere-se à forma como as opções da interface permanecem iguais em contextos similares e são diferentes se aplicadas em contextos diferentes.
7. **Significado dos Códigos** (*Significance of Codes*): Refere-se ao uso de códigos e nomes com forte relação com os itens ou ações a que se referem.
8. **Compatibilidade** (*Compatibility*): Refere-se ao relacionamento entre as características do usuário e as características da tarefa a ser realizada.

Alguns destes critérios se subdividem em outros critérios elementares, conforme denominados pelos autores. As subdivisões para os critérios são apresentadas a seguir:

Tabela 3 - *Subcritérios Ergonômicos por Scapin e Bastien*

1) Condução:
1.1)Presteza: Meios usados para levar usuário a realizar certas ações.
1.2)Agrupamento/Distinção de Itens: Organização visual dos ítens de informação relacionados entre si de alguma maneira (por localização ou por formato).
1.3)Feedback Imediato: Reposta do sistema às ações do usuário.
1.4)Legibilidade: Características da informação apresentada na tela que facilitam ou não sua leitura e entendimento.
2) Sobrecarga de Trabalho:
2.1) Brevidade: Objetivo de limitar a leitura e sobrecarga de entrada de dados bem como o número de passos necessários (divide-se em concisão e ações mínimas).
2.2) Densidade Informacional: Sobrecarga imposta ao usuário relacionada a todo o conjunto de informações apresentadas.
3) Controle Explícito:
3.1) Ação Explícita do Usuário: Apenas ações solicitadas pelo usuário são processadas.
3.2) Controle do Usuário: Os usuário devem sempre estar no controle do processamento do sistema.
4) Adaptabilidade:
4.1) Flexibildade: O sistema deve permitir que usuários personalizem sua interface.

4.2) Experiência do Usuário: O sistema deve respeitar o nível de experiência do usuário.
5) Gerenciamento de Erros:
5.1) Proteção Contra Erros: Mecanismos disponíveis para detectar e prevenir entradas de dados com erros, ou ainda ações não recuperáveis.
5.2) Qualidade das Mensagens de Erro: Legibilidade e exatidão da informação dada ao usuário devido a um erro cometido.
5.3) Correção de Erros: Mecanismos disponíveis para os usuários corrigirem os eventuais erros cometidos.
6) Consistência - não apresenta sub-critérios.
7) Significado dos Códigos - não apresenta sub-critérios.
8) Compatibilidade - não apresenta sub-critérios.

Na seção seguinte, será apresentada a Técnica de Co-Inspeção Avaliativa e sua relação com o ErgoMonitor.

2.4 Técnica de Co-Inspeção Avaliativa

Cybis e Scapin (2002) desenvolveram a Técnica de Co-Inspeção Avaliativa, que serve como uma ferramenta de suporte à avaliação da usabilidade. Ela é um exemplo de uma técnica de avaliação qualitativa-diagnóstica de usabilidade de *websites* fortemente baseada em tarefas. A técnica permite a classificação dos problemas por critérios ergonômicos.

O nome da técnica revela duas facetas que são explicadas a seguir. A inspeção é realizada por meio de *checklists* aplicadas aos atributos de qualidade que compõem a ergonomia das interfaces *web*. Da inspeção devem participar os usuários e os projetistas, por isso percebe-se o aspecto colaborativo. A avaliação é realizada nas inspeções que necessitam do julgamento dos avaliadores.

Esta técnica se aplica à usabilidade e está associada ao ErgoMonitor, uma vez que este ambiente define a implementação de uma combinação entre técnicas comprobatórias e diagnósticas. A ferramenta desenvolvida neste trabalho pode servir de apoio à aplicação dessa técnica.

A técnica tem como uma de suas características principais a racionalização das atividades de avaliação ou inspeção. Desta forma, são apenas coletadas informações relacionadas às avaliações, no que diz respeito ao contexto de utilização do software onde a técnica será aplicada. Além disso são examinadas apenas as interações com o sistema relativas às tarefas mais importantes, críticas ou frequentes. Por fim, só serão apresentadas ao avaliador questões dos *checklists* aplicavéis aos componentes técnicos da interface e a seu contexto operacional.

O método de uso desta técnica é composto pelas seguintes etapas:

- Análise do contexto de operação do *website* em avaliação, por meio da busca de informações pertinentes à avaliação de usabilidade:
  - ✓ características do *website*: reconhecimento e descrição assistida dos componentes globais do *website* e de sua *homepage*;
  - ✓ entrevista com os projetistas: captura da intenção deles com o *website*, mostra cenário prescrito, identificação das sequências de páginas associadas às tarefas possíveis;
  - ✓ entrevista com os usuários: obter a opinião deles sobre o *website*, mostra cenário real, identificação das sequências de páginas associadas às tarefas mais frequentes e importantes.
- Integração dos cenários descritos pelos projetistas e usuários, para definir páginas *web* alvo das atividades de reconhecimento local de componentes, ou seja, compor o elenco de cenários de operação (série de páginas *web* e ficha de descrição) a serem considerados nas inspeções avaliativas.
- Armazenamento dos resultados das atividades em uma base de conhecimentos sobre o contexto de operação.
- Tratamento das respostas fornecidas e elaboração dos relatórios finais de acordo com

várias perspectivas.

- Configuração do material de avaliação e aplicação de *checklists* compostos apenas de questões aplicáveis por nível de importância.

As fichas de inspeções definidas e ponderadas são o instrumento de realização da técnica. São realizadas três tipos de inspeções avaliativas: global do *website*, local da página de acesso e locais relativas às páginas ligadas às principais tarefas. O avaliador precisa percorrer o *website* seguindo os caminhos relacionados às tarefas principais, para ter uma boa visão geral. Então ele deve procurar responder as questões propostas sobre as fichas de verificação.

A seção seguinte irá abordar o assunto da automatização do processo de avaliação e suas vantagens.

2.5 Automatização do Processo de Avaliação

A automatização do processo de avaliação apresenta vantagens que são citadas a seguir: redução de tempo e de custos, consistência crescente de erros não cobertos, redução da necessidade de conhecimento específico para a avaliação, aumento da cobertura das características avaliadas, possibilidade de comparação entre modelagens alternativas, incorporação da avaliação na fase de modelagem [Ivory & Hearst, 2001].

A automatização das avaliações de usabilidade torna o processo mais ágil e menos custoso, dependendo menos da intervenção humana e, portanto, das variações a que estão sujeitas as pessoas que participarão desta avaliação [Fernandez & Hussmann, 2009].

Essa vantagem é ampliada pelo fato de que os dados dos arquivos que registram as atividades do servidor *web* (arquivos de *log*) permitem acompanhar o uso do *website*, em condições de trabalho naturais, além de representar vários usuários em um longo período de tempo [Tec-Ed, 1999].

Embora em uma avaliação de páginas de internet seja necessária a consideração de

atributos internos e externos das páginas, a automatização completa só pode ser realizada para os atributos internos. Para os externos, um passo de avaliação humana é requerido [Brajnik, 2000].

A avaliação por meio da análise de *log* pode ser usado por empresas, as quais analisam seus arquivos de *log* para detalhes mais apurados sobre a usabilidade de seus *websites*, em outras palavras, como os visitantes bem-sucedidos desses *websites* encontram informações específicas que necessitam ou atingem seus objetivos transacionais lá [Tec-Ed, 1999].

Um arquivo ideal de *log* deve permitir a reconstrução de todo o episódio da visita do usuário ao seu *website*, fornecendo quem fez a visita, o caminho percorrido, quanto tempo ficou, e por onde saiu [Tec-Ed, 1999]. Devido às informações armazenadas pelos arquivos de *log* e à estrutura de funcionamento da arquitetura cliente/servidor, normalmente não se consegue um arquivo de *log* com todas as informações desejadas, principalmente porque a maioria dos *websites* não identifica seus visitantes nem suas intenções com a visita. Além disso, o arquivo de *log* também não contém informações que permitam responder com certeza se o usuário encontrou a informação que procurava.

Morandini (2002) amplia a discussão, identificando dificuldades existentes nos softwares das ferramentas automatizadas existentes, a saber:

- não oferecem grande apoio para análise qualitativa;
- estão restritos à utilização da técnica de medição de desempenho dos usuários durante a interação ao automatizarem os procedimentos para análise dos dados;
- não possibilitam a comparação de interações de diferentes usuários em um mesmo *website*;
- não permitem ao avaliador filtrar o que interessa nas interações;
- dependem muito da habilidade do avaliador para conclusões; e
- não usam taxas basicamente relativas às usabilidades dos *websites*.

Morandini (2002) propõe um sistema chamado ErgoMonitor para tentar abordar a

maioria dos aspectos mencionados como lacunas nas ferramentas automatizadas de apoio à avaliação de usabilidade existentes na ocasião.

Na seção a seguir, será descrito com mais detalhes o sistema ErgoMonitor.

2.6 A Proposição do ErgoMonitor

O ErgoMonitor busca monitorar a usabilidade de *websites* pela coleta seletiva e análise de entradas de arquivos de *log* correspondentes a interações reais. São calculadas métricas que quantificam a usabilidade do *website* como fruto da comparação entre essas entradas e modelos de comportamento esperado.

O ErgoMonitor é capaz de identificar, por meio de análises quantitativas, problemas nas interações que se estabelecem entre usuários e websites. As análises qualitativas estão além do escopo desta ferramenta.

A Figura 1 a seguir mostra uma visão geral do ErgoMonitor:

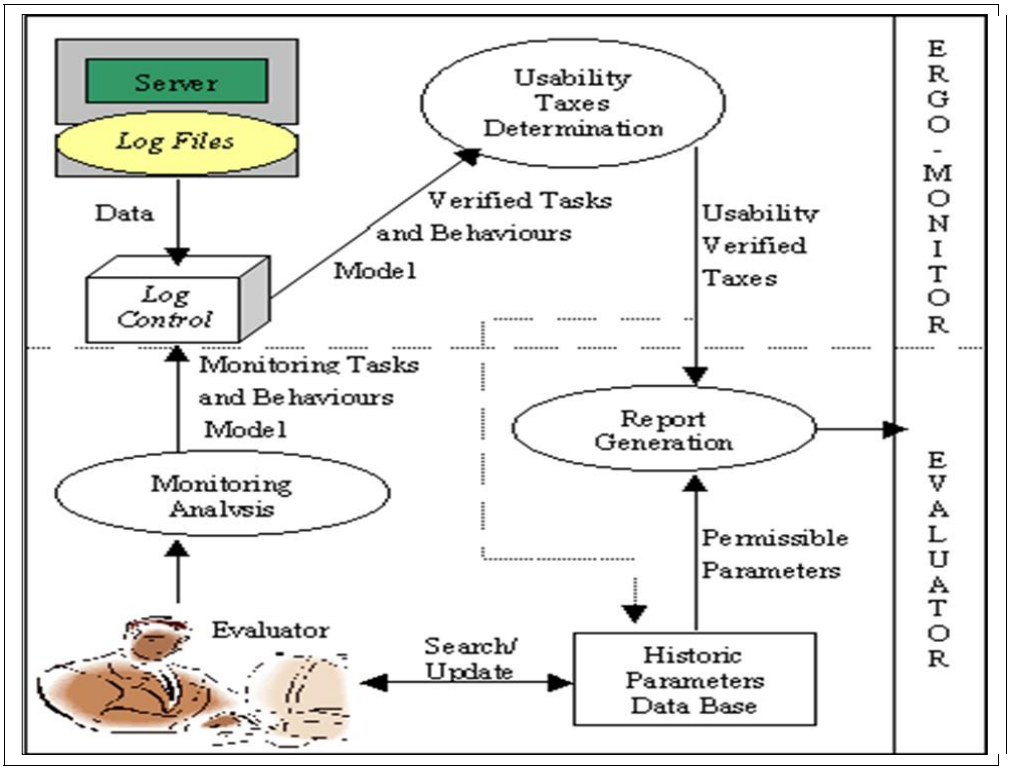

Figura 1 *– Visão Geral do ErgoMonitor extraída de [Morandini, 2002]*

O usuário da ferramenta, avaliador do *website*, participa em revisões do *website* monitorado, em configurar os parâmetros para a avaliação de seu *website*, em modelar as tarefas a serem monitorados, em interpretar a usabilidade do *website*, e em montar uma base de medidas históricas de usabilidade.

Pela sua forma imperceptível para o usuário de atuar e ainda pelo fato de avaliar dados de interações ocorridas em situações reais de uso do *website*, a ferramenta pretende reduzir a distância entre o que se espera e o que se pode obter em uma avaliação de usabilidade de *websites* baseada na observação da interação.

Morandini (2002) relata que, após a coleta e filtragem das entradas do arquivo de *log*, são determinadas taxas (medidas intermediárias da qualidade da interação; como sucessos, desistências e cancelamentos) e calculadas métricas (medidas de usabilidade; ou seja, de eficácia e eficiência) de usabilidade. Os valores verificados são confrontados com parâmetros admissíveis estabelecidos por especialistas. Então o avaliador elabora um laudo sobre os problemas de usabilidade relatados pelo processo.

Ainda segundo Morandini (2002), as atividades que devem ser implementadas para que o ErgoMonitor não seja mais tão dependente de suas restrições são:

- necessidade de ter um endereço IP exclusivo e único para cada usuário
- necessidade de registro de todas as páginas *web* no arquivo de *log*, inclusive páginas de erro e de ajuda
- necessidade de realização de mais experimentos para auxiliar na definição de parâmetros admissíveis para a tomada de decisão
- necessidade de automatizar o cálculo das taxas e análises, com relação aos parâmetros admissíveis

Na seção a seguir, o tema da automatização de um método de avaliação será descrito, abordando como medir a usabilidade através de seus atributos. Serão ainda apresentadas ferramentas que automatizam a avaliação de usabilidade.

2.7 Automatização e Atributos de Usabilidade

A automatização da coleta e análise das informações sobre as interações dos usuários com o sistema é tratar estas informações, para facilitar seu uso para avaliar o que elas representam [Fernandez & Hussmann, 2009]. No entanto, uma avaliação de usabilidade mais detalhada de sistemas específicos deve considerar os atributos de usabilidade, fatores de qualidade em que a usabilidade pode ser decomposta, para ser analisada e medida de uma forma melhor.

Existem algumas ferramentas disponíveis que buscam realizar de maneira automatizada a avaliação de usabilidade. Dentre tantas a serem encontradas, citamos algumas:

Tabela 4 *- Ferramentas que Avaliam Usabilidade*

- **Google Analytics** - gratuito, informa quantos visitantes e visualizações por página entre outras coisas; fornece diversos relatórios
- **AWStats** - gratuito, provê informação detalhada sobre visitantes, navegadores e sistemas operacionais; além de estatísticas como contar visitantes e visualizações de página, instalado no servidor web
- **Piwik** - código aberto, instalado no servidor web
- **Grape Web Statistics** - gratuito, código aberto, estatísticas como números de visitantes e fontes de referência
- **Trace Watch** - gratuito, fornece estatísticas básicas, permite configurar dados, oferece análise de caminhos percorridos pelos visitantes
- **GoingUp!** - estatísticas básicas, tendência de tráfego, mapa de calor, monitoramento de link
- **StatCounter** - gratuito até 250 mil visitas a páginas por mês, estatísticas básicas incluindo referência a fontes, páginas populares, páginas de saída, estatística de navegador e sistema operacional, dentre outras

No próximo capítulo, serão abordados alguns conceitos e ferramentas que auxiliaram no desenvolvimento da ferramenta proposta neste trabalho e na compreensão do funcionamento de *websites*.

3 *Estruturas Computacionais Empregadas*

Neste capítulo são apresentadas as características da arquitetura cliente/servidor, importante para o funcionamento de *websites*, e são detalhados os recursos a serem usados para o desenvolvimento da ferramenta a ser introduzida neste trabalho. Para manipular melhor os arquivos de *log*, foi usado o conceito de expressão regular por um *parser* (programa que recebe uma entrada e identifica suas partes) de arquivo de *log*, desenvolvido na linguagem PHP.

3.1 Arquitetura Cliente/Servidor

Na arquitetura cliente/servidor, o servidor provê serviços que são consumidos por um ou mais clientes, não necessariamente ao mesmo tempo. Os clientes iniciam o diálogo requisitando um serviço. Os servidores ficam passivamente esperando as requisições dos clientes. O software cliente/servidor deve ser independente de plataforma, seja hardware ou Sistema Operacional.

Orfali et al. (1996) descrevem a relação cliente/servidor para a *Web* da seguinte forma:

> "Servidores *web* e clientes *web* são um tipo especial de cliente/servidor. Um servidor *web* retorna um documento quando o cliente o solicita pelo nome. Os navegadores *web* são clientes *web* universais. Através deles especificamos um URL (*Unified Resource Locator*), que é usado para dar nomes e acessar globalmente todos os recursos *web* (documentos, imagens, clipes de som ou vídeo e programas). As URLs descrevem totalmente onde reside um recurso e como obtê-lo. O protocolo HTTP (*Hypertext Transfer Protocol*) é usado para acessar recursos no espaço URL, estabelecendo uma conexão cliente/servidor, transmitindo e recebendo parâmetros e encerrando a conexão. Os programas do servidor geraram arquivos de texto HTML (*Hypertext Markup Language*), a língua franca dos documentos *web*, em resposta a requisições dos clientes. Um documento *web* é também chamado página. Os navegadores interpretam os comandos HTML e mostram textos e imagens da forma apropriada."

O servidor *web* desta arquitetura é um servidor no qual funciona um programa que recebe as requisições dos navegadores e as processa. Existem basicamente dois programas para servidores *web* largamente utilizados: o Apache e o IIS da Microsoft.

Um ponto importante em nossas considerações é o fato que, ao se consultar o *website* da Netcraft, percebe-se que, de um total de quase 526 milhões de *websites*, cerca de 342 milhões usam em seu servidor *web* o software desenvolvido pela Apache, enquanto apenas 82 milhões usam o software desenvolvido pela Microsoft. Em outras palavras temos 65% do total de navegações feitas na *Web* fazendo acesso a servidores onde o software Apache está instalado [Netcraft 2011].

Este trabalho foi desenvolvido com base nos arquivos de *log* gerado pelos servidores *web* com Apache instalado. Não foram considerados os arquivos de *log* dos servidores *web* com o software da Microsoft (IIS) instalado, por abarcar um público significativamente pequeno em relação ao concorrente, e também para limitar o escopo do atual trabalho [Netcraft 2011]. Williams e Lane (2004) afirmam que, apesar das possíveis preferências particulares, o fato de o software Apache fazer baixa requisição de recursos, possuir código aberto, ser rápido e escalável pode ter contribuído para ser largamente utilizado.

Quando o Apache é executado em uma máquina, torna-a um servidor *web* e lhe permite lidar com requisições simultâneas de navegadores [Williams & Lane, 2004]. O Apache é projetado para funcionar em Sistemas Operacionais multitarefas como o Linux, Mac OS X e Microsoft Windows.

O arquivo de configuração do Apache controla como ele “escuta” as requisições da rede e as atende. Entre outras coisas, estabelece como ele lida com requisições HTTP e relaciona nos arquivos de *log* as requisições ocorridas.

A seguir será visto um descritivo dos arquivos de *log*, para permitir uma melhor compreensão sobre as informações que eles disponibilizam para os servidores *web* e sobre como realizar a identificação de tarefas.

3.2 Arquivos de *Log*

Os *logs* são escritos em um formato configurável. O formato pode ser *common* e *combined*. As diretivas são colocadas em um arquivo de configuração e representam as informações a serem armazenadas no arquivo de *log*. Como exemplo de uma suposta entrada de um arquivo de *log*, temos a linha abaixo:

Tabela 5 - *Exemplo de Entrada de Arquivo de Log*

127.0.0.1 - frank [10/Oct/2000:13:55:36 -0700] "GET /apache_pb.gif HTTP/1.0" 200 2326 "http://www.example.com/start.html" "Mozilla/4.08 [en] (Win98; I ;Nav)"

Seguem abaixo duas linhas possíveis de um arquivo de configuração com as diretivas correspondentes e as devidas explicações. A fonte para tais informações foi a documentação do próprio Apache em [Apache 2011 b].

Tabela 6 - *Formato de Arquivo de Log no Apache*

LogFormat "%h %l %u %t \"%r\" %»s %b" common
LogFormat "%h %l %u %t \"%r\" %»s %b \"%{Referer}i\" \"%{User-agent}i\"" combined

%...h Servidor remoto --» IP da máquina que fez o acesso
%...l *Logname* remoto --» é para redes internas fortemente controladas, quase não é usado
%...u Usuário remoto --» autenticação HTTP (proteção por senha),
para erro 401 deve ser desconsiderado
%...t Momento em que a requisição foi recebida --» pode-se configurar o formato desejado
%...r Primeira linha da requisição
%...s Status (%...»s for the last) --» para redirecionamento, é o status do pedido original
sucesso (2--) / redirecionamento (3--) / erro do cliente (4--) / erro no servidor (5--)
%...b Tamanho da resposta em bytes, excluindo os cabeçalhos HTTP
%...{Foobar}i --» conteúdo da linha de pedido Foobar enviada ao servidor

Percebe-se que a diferença entre os dois formatos é a presença das informações "Referer" e "User-agent" no formato *combined*. Este formato permite saber, além das

informações já contidas no formato *common*, o endereço *web* acessado pelo cliente ("Referer") e de qual navegador e Sistema Operacional o cliente fez esta requisição ("User-agent").

De acordo com o resultado da requisição feita ao servidor *web*, um código é retornado para o navegador, que permite saber se houve algum erro, ou ainda qual foi o retorno do servidor. Seguem abaixo os códigos de status devolvidos pelos servidores *web* conforme a situação e que são armazenados nos arquivos de *log*.

Tabela 7 - *Códigos de Status Devolvidos pelo Servidor Web*

"100" Continue	*"400" Bad Request*
"101" Switching Protocols	*"401" Unauthorized*
"200" OK	*"402" Payment Required*
"201" Created	*"403" Forbidden*
"202" Accepted	*"404" Not Found*
"203" Non-Authoritative Information	*"405" Method Not Allowed*
"204" No Content	*"406" Not Acceptable*
"205" Reset Content	*"407" Proxy Authentication Required*
"206" Partial Content	*"408" Request Time-out*
"300" Multiple Choices	*"409" Conflict*
"301" Moved Permanently	*"410" Gone*
"302" Found	*"411" Length Required*
"303" See Other	*"412" Precondition Failed*
"304" Not Modified	*"413" Request Entity Too Large*
"305" Use Proxy	*"414" Request-URI Too Large*
"307" Temporary Redirect	*"415" Unsupported Media Type*
	"416" Requested range not satisfiable
	"417" Expectation Failed
	"500" Internal Server Error
	"501" Not Implemented
	"502" Bad Gateway
	"503" Service Unavailable
	"504" Gateway Time-out
	"505" HTTP Version not supported

Alguns destes códigos são comentados com maiores detalhes abaixo, por serem mais frequentes ou relevantes [BasicoWeb, 2012]:

Tabela 8 *- Alguns Códigos de Status Comentados*

- 200 - OK - A requisição teve sucesso e está tudo Ok. Junto com este código o servidor deve enviar, acompanhado de alguns headers, o conteúdo do recurso requisitado.
- 302 - Found - Este é o código de redirecionamento mais comum. Ele descreve um redirecionamento temporário, ou seja, pode ser que numa próxima requisição esse redirecionamento não seja necessário.
- 301 - Moved Permanently - O recurso foi permanentemente movido para um outro local.
- 304 - Not Modified - Esse código é usado quando o cliente faz uso de caching, ou seja, guarda cópias locais dos recursos acessados. Ele informa ao cliente que o recurso não foi modificado desde a última requisição e que a versão guardada em cache pode ser usada.
- 404 - Not Found - Este código informa ao cliente que o recurso não foi encontrado no servidor. Pode ser que o recurso realmente não exista ou apenas esteja temporariamente indisponível.
- 410 - Gone - O código 410 informa que o recurso requisitado não existe mais, ou seja, foi intencionalmente removido do servidor e não há endereço para redirecionamento. De acordo com a especificação, esse estado deve ser considerado permanente.
- 403 - Forbidden - Esse código informa que você simplesmente não pode acessar o recurso requisitado. Nem mesmo é possível acessar o recurso por meio de autenticação.
- 401 - Unauthorized - Nesse caso o recurso pode ser acessado, desde que você possua as informações de autenticação corretas (usuário e senha) para acessá-lo.
- 500 - Internal Server Error - Esse erro é bastante comum. Ele informa que algo de inesperado aconteceu no servidor e a sua requisição não pôde ser satisfeita.
- 502 - Bad Gateway - Significa que o servidor que você acessou atua como um proxy ou gateway e que o servidor “acima” dele reportou algum erro ao tentar completar a requisição.
- 503 - Service Unavailable - O recurso está temporariamente indisponível. Este erro pode ser causado por sobrecarga no servidor ou por alguma operação de manutenção.

A seguir é apresentado como os arquivos de log participam no desempenho da funcionalidade do servidor *web*.

3.3 Arquivos de *Log* no Acesso ao Servidor *Web*

Um servidor w*eb* é uma máquina com um programa instalado, que é responsável por receber as requisições de recursos disponíveis na *Web* feitas pelos navegadores, ou seja, pelos clientes *web* [TechTerms, 2012].

Basicamente um arquivo de *log* de um servidor *web* é um arquivo de texto que relaciona todas as ações que ocorreram no servidor. Em outras palavras, cada requisição de um recurso *web*, seja de uma página, de uma figura, de um arquivo de estilo, de uma música, de um vídeo, ou de algum outro recurso é armazenada neste arquivo.

Além de armazenar o recurso que foi solicitado, o arquivo *web* armazena o momento exato em que o pedido ocorreu (data, horário e fuso horário), o status do pedido (se foi ou não bem sucedido, por exemplo), o navegador utilizado para fazer o pedido e o comando que foi executado pelo navegador.

Servidores *web* mantêm uma listagem de cada requisição feita para o servidor nos arquivos de *log*. Toda a comunicação entre o usuário, por intermédio do navegador, e o servidor *web* é registrada nestes arquivos. Com o uso de ferramentas de análise para arquivos de *log*, é possível se ter uma boa idéia sobre: de onde os visitantes vêm, quão frequentemente eles retornam, e como eles navegam através de um *website*.

No entanto essa possibilidade é interferida pelo "cacheamento dinâmico de páginas" realizado pelos clientes *web*. Esta foi uma forma encontrada de agilizar o serviço de entrega de recursos *web* pelos navegadores, pois eles armazenam localmente alguns dos recursos e os usam em momentos apropriados, não os carregando sempre dos servidores *web* [RNP, 2012]. Consequentemente, esta requisição não será listada no arquivo de *log*, pois ela é realizada localmente.

Outra interferência pode ocorrer quando o computador onde está o cliente *web* estiver configurado para receber IP dinamicamente. Desta forma, dois acessos vindos do mesmo computador em ocasiões distintas devem ser considerados como vindos de computadores diferentes, no caso de em cada acesso o computador estar com um IP diferente. Então não é

possível relacionar com precisão quantas vezes o mesmo cliente *web* acessou determinado recurso *web*.

Com o objetivo de testar a ferramenta que será desenvolvida com um *website* real em funcionamento e viabilizar o acesso aos arquivos de *log* do servidor *web* onde este *website* está hospedado foram feitos contatos com o Centro de Computação Eletrônica da Universidade de São Paulo (CCE | USP). A utilidade deste teste é validar a ferramenta que foi desenvolvida.

O motivo para este procedimento é que atualmente o servidor *web* que abriga o *website* da Escola de Artes, Ciências e Humanidades (EACH | USP), onde esta pesquisa está sendo desenvolvida, está sendo administrado pelo CCE | USP. A EACH | USP não tem permissões de administrador neste servidor. Então era necessária permissão ao menos para a leitura dos arquivos de *log* deste servidor e a realização dos testes de usabilidade no *website* da EACH | USP.

Tendo sido concedida a permissão de leitura aos arquivos de *log* mencionados acima, este trabalho passou a ser desenvolvido tendo por base estes arquivos. Desta forma, as funcionalidades da ferramenta já realizadas até o momento foram testadas com estes arquivos de *log*.

Na próxima seção, será mostrado o processo pelo qual o arquivo de *log* é analisado: por meio do reconhecimento de sua estrutura.

3.4 Reconhecimento da Estrutura de Arquivos de *Log*

Um *parser* é um programa que recebe uma entrada e identifica as diversas partes que a compõem [SearchSOA, 2012]. Para esta ação ser favorecida, é útil que sempre as mesmas partes estejam presentes e numa ordem já determinada. *Log Parsers*, ou *parsers* de arquivos de *log*, são beneficiados com o fato de os arquivos de *log* terem apenas dois formatos, sendo que um está contido no outro [Apache, 2011].

Uma expressão regular (ou abreviadamente regex ou ainda regexp) é um conjunto

especial de textos para descrever um padrão de busca. Pode ser comparada com uma expressão onde alguns caracteres não possuem significado, mas estão ali para representar um conjunto de outros caracteres com significado. Um exemplo clássico é a expressão “*.txt” que representa todos os arquivos com extensão “txt” e que tem como equivalente a expressão regular “.*\.txt$.” [Regexp, 2011].

O termo expressão regular pode ainda ser entendido como um padrão descrevendo certa quantidade de texto. Percebe-se que as expressões regulares ajudam a reconhecer padrões em conjuntos de caracteres. Os *log parsers* fazem uso das expressões regulares para identificar as partes de cada entrada do arquivo de *log*, servindo como auxílio para filtrar os dados de acesso ao servidor *web* armazenados nestes arquivos e estabelecendo quando os critérios desejados são atingidos.

Para o desenvolvimento da ferramenta, foi escolhida a linguagem PHP, cujo nome é um acrônimo recursivo para “PHP: Hyperetext Preprocessor”. A motivação para o uso de PHP é que ela é a linguagem de *script* (linguagem de programação executada no interior de programas ou de outras linguagens de programação) da *Web* mais vastamente mantida e utilizada, além de ser de código aberto e mantida e melhorada por esforços de uma comunidade.

PHP geralmente é embutida e combinada com o HTML de uma página *web*. Quando a página é requisitada, o servidor executa o *script* PHP e substitui o resultado desta execução dentro da página. Além disso, realiza os acessos a Banco de Dados contidos no código. Então retorna toda a saída do *script* para o navegador exibir com o texto HTML [Williams & Lane, 2004].

Outro detalhe importante para o uso de PHP é que a ferramenta proposta neste trabalho deve ser acessível pela Internet para ser utilizada por quem desejasse. O uso de PHP para seu desenvolvimento torna isso possível.

Para auxiliar na análise do conteúdo dos arquivos de *log*, e servir como um *Log Parser* para o PHP foi utilizada a classe "PHP apache *log* parser class", criada por Sam Clarke, em 2005. Ela é gratuita e está disponível para acesso em <http://www.phpclasses.org/package/2596-PHP-

Parse-Apache-log-files-in-the-common-log-format.html>. Ela faz uso de expressões regulares, para coincidir a entrada do *log* com o padrão estabelecido, através da execução da função "*preg_match()*", como exemplificado abaixo:

Tabela 9 *- Função preg_match() Usada em Arquivos de Log*

```
preg_match("/^(\S+) (\S+) (\S+) \[([^:]+):(\d+:\d+:\d+) ([^\]]+)\] \"(\S+) (.*?) (\S+)\" (\S+)
(\S+) (\".*?\") (\".*?\")$/", $line, $matches); // pattern to format the line
```

Desta forma, a ferramenta desenvolvida fornece para esta classe cada entrada do arquivo de log a ser analisado e recebe cada parte contida nesta entrada para tratar e extrair as informações necessárias.

O capítulo seguinte aborda a ferramenta ErgoMonitor, seu desenvolvimento e os testes que foram feitos para validar sua implementação e obter resultados relacionados à usabilidade de um *website*.

4 A Implementação de um Parser para Auxiliar na Determinação de Taxas e Métricas de Usabilidade

Neste capítulo, será abordada a ferramenta ErgoMonitor propriamente dita: seu desenvolvimento, a forma como as estruturas computacionais envolvidas foram efetivamente utilizadas na sua elaboração, a estrutura empregada para o armazenamento das avaliações, dentre outras coisas. Serão apresentados ainda alguns resultados práticos do uso da ferramenta em arquivos de *log* reais, bem como as taxas e métricas geradas neste processo.

Inicialmente faz-se necessário esclarecer que o ErgoMonitor já foi idealizado em Morandini (2002). No entanto, até o presente momento este ambiente tem sido utilizado apenas como um protótipo, sendo que alguns dos módulos já implementados só poderiam ser utilizados em versões *desktop*. Além disso, o filtro realizado no arquivo de *log* era mais simples do que o realizado neste trabalho, identificando apenas páginas *web* cujo nome terminasse por ".php" ou ".html", o que limitava a possibilidade de ocorrerem opções no endereço da página ou ainda de se querer atingir um recurso *web* como imagem ou vídeo. Por fim, o armazenamento das avaliações é realizado diretamente num banco de dados e as métricas são efetivamente calculadas.

A etapa inicial do ErgoMonitor é justamente a que trabalha com os dados do arquivo de *log* e separa as informações contidas nele. Esta ação é desempenhada pelo ErgoParser, o qual não estava implementado de forma definitiva no trabalho inicialmente apresentado em Morandini (2002). Assim, de forma geral, o foco principal deste trabalho está no ErgoParser, um *parser* que atua sobre os arquivos de *log* no sentido de auxiliar o reconhecimento das realizações de tarefas contidas neste arquivo e, a partir destas observações, apresentar as taxas correspondentes que apoiem o cálculo de métricas de usabilidade.

A ferramenta ErgoMonitor foi planejada para ser um ambiente disponível na *Web* e para ser usada mediante um cadastro prévio. Neste cadastro, uma pessoa responsável pela

avaliação de um *website* que desejar utilizar a ferramenta, fornece o nome do *website* e um arquivo de *log*. Então recebe um nome de usuário na ferramenta para fazer as avaliações.

Para cada nome de usuário fornecido, são gerados registros iniciais no Banco de Dados, relacionando o nome de usuário do responsável pela avaliação com o *website* e o arquivo de *log* correspondente. Esta ação é detalhada neste trabalho após ser apresentada a estrutura das tabelas. A seguir o responsável pela avaliação pode usar a ferramenta para fazer o cadastro dos recursos deste *website*, vincular algumas tarefas e seus respectivos recursos e avaliar o *website* como um todo ou, se preferir, alguma tarefa específica suportada pelo *website*. Com isso, espera-se obter taxas e métricas de usabilidade para esta avaliação.

Uma maneira de expressar a medida de alguma característica é por meio do uso de taxas. Especificamente nesta ferramenta, as taxas servem como base para auxiliar na medição da usabilidade do website a ser avaliado, identificando as tarefas que foram realizadas, quantas vezes as tarefas foram interrompidas, quantas vezes houve troca do navegador que estava sendo usado, ou quantas vezes passou um tempo muito longo até que o uso do website fosse retomado.

Dado que as taxas são coletadas tendo por base um arquivo de *log* do servidor *web* e de forma automatizada, não ocorre no processo de coleta qualquer interferência no uso corriqueiro do *website*, nem algum constrangimento aos usuários do *website* que está sendo avaliado. Após a coleta das taxas, a ferramenta calcula as métricas de usabilidade do *website* avaliado.

Na próxima seção, descreve-se como ocorreu o desenvolvimento da ferramenta, as etapas percorridas e o relacionamento da teoria relacionada ao assunto com o que foi realizado na prática.

4.1 Desenvolvimento da Ferramenta

Esta seção aborda alguns aspectos técnicos que foram necessários no desenvolvimento da ferramenta. Alguns conceitos serão apresentados em detalhes, facilitando a compreensão

necessária do funcionamento da ferramenta.

PHP foi a linguagem de programação usada para o desenvolvimento desta ferramenta, uma vez que ela é fortemente apropriada para programação na *Web*. Assim sendo, os responsáveis pelas avaliações podem utilizar a ferramenta por meio de um navegador *web*. O Banco de Dados utilizado para armazenar as informações do *website* e dados associados aos resultados das avaliações foi o MySQL.

Para o desenvolvimento da ferramenta, inicialmente foram feitos alguns ajustes na classe PHP que trata arquivos de *log* para colocá-la em funcionamento dentro de um código PHP com um exemplo de arquivo de *log* real.

Então, a este código PHP foram sendo acrescentadas rotinas para, dentre outras ações:

- filtrar apenas os recursos *web* que deveriam ser analisados, conforme descritos pelo responsável pela avaliação;
- capturar o intervalo de tempo entre os acessos a dois recursos *web* distintos e sequenciais;
- destacar intervalos de tempo considerados muito grandes;
- identificar mudanças de navegador ou de Sistema Operacional;
- perceber acessos a páginas *web* de ajuda ou de erro; e
- inserir um item de segurança para os usuários necessitarem de senha para acesso.

A Figura 2 a seguir apresenta a metodologia de funcionamento do ErgoMonitor e onde se encaixa o ErgoParser. Percebe-se a necessidade de preparo do ambiente de trabalho da ferramenta, tanto da parte do administrador, como da parte dos responsáveis pelas avaliações que farão uso da mesma.

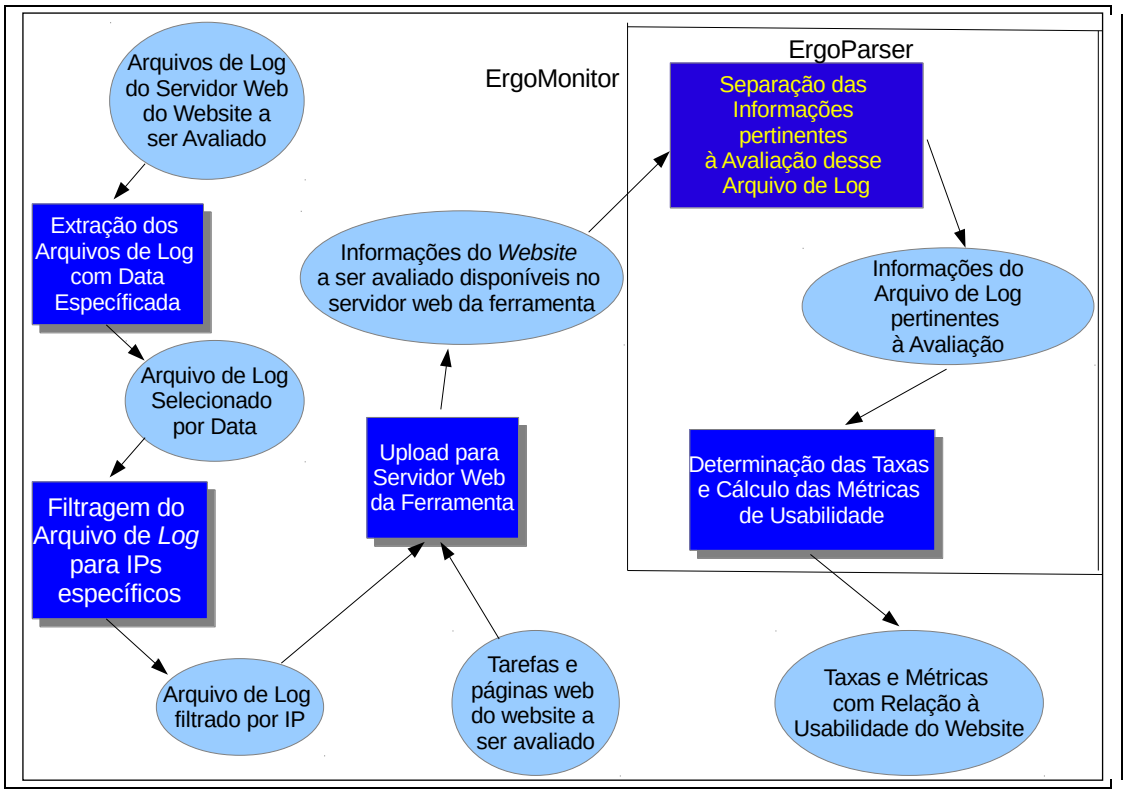

Figura 2 *– Estrutura do ErgoMonitor*

Com relação a esta ferramenta não se pretende que seja necessária qualquer alteração no *website* para ser realizada sua avaliação. No entanto, pode ocorrer de o formato do arquivo de *log* deste *website* ter sido configurado diferentemente dos padrões para arquivos de *log* destacados neste trabalho. Para tanto, foi necessária a adaptação da classe *PHP Log Parser* para que a expressão regular coincidisse com o formato de arquivo de *log* adotado pelo servidor *web* onde o *website* está hospedado.

A proposta inicial do ErgoMonitor foi descrita em Morandini (2002). Esta proposta inicial foi implementada no mesmo trabalho de forma preliminar e simplificada. No trabalho atual, esta algumas alterações e adaptações foram feitas ao que inicialmente foi proposto. A estrutura apresentada na figura 2 acima já reflete estas modificações, bem como as implementações realizadas, que resultaram na ferramenta ora apresentada.

O conceito de tarefa, o mapeamento de tarefas e sua implementação na ferramenta

serão os assuntos da próxima seção.

4.1.1 Mapeamento de Tarefas

No contexto deste trabalho, define-se tarefa como o que se pretende realizar ao ocorrer uma interação com o *website*. Mais diretamente, tarefa é uma ação delimitada com início e fim. Para uma tarefa ser realizada existem algumas possibilidades ou caminhos. Os caminhos são sequências de recursos *web*, desde o recurso *web* inicial até o recurso *web* desejado ou final. A tarefa é considerada realizada quando um dos caminhos possíveis para ela é percorrido.

Para que a ferramenta funcione, é necessário que as tarefas a serem avaliadas sejam mapeadas, para possibilitar a posterior comparação destas com as tarefas identificadas no arquivo de *log*. Mapear as tarefas, na prática, equivale a identificar os possíveis caminhos a serem percorridos pelos usuários do *website* para realizarem a tarefa em questão.

Destaca-se que um recurso web pode ser uma página *web* ou algum conteúdo disponibilizado internamente à página *web* que é considerado importante na execução da tarefa, como um arquivo de texto, um vídeo, ou uma imagem.

O esquema na Tabela 10 a seguir explicita o conceito de tarefa. Neste esquema, RW corresponde a um recurso *web* qualquer, RWi é o recurso *web* inicial da tarefa, RWf é o recurso *web* final da tarefa e os termos entre colchetes podem ou não aparecer.

Tabela 10 *- Conceito de Tarefa*

TAREFA = <RW inicial> - n * <RW intermediário> - <RW final>
TAREFA1 = RWi1 – [RWa – ... – RWn] – RWf1 TAREFA2 = RWi2 – [RWb – ... – RWp] – RWf2 TAREFA3 = RWi3 – [RWc – ... – RWq] – RWf3

Por exemplo, se uma tarefa do *website* for “Visualizar o Cardápio de Almoço do Restaurante”, esta tarefa só será considerada realizada se o recurso *web* correspondente ao cardápio estiver listado no arquivo de *log*. No trecho a seguir, extraído de um arquivo de *log*,

percebe-se a sequência de páginas *web* percorrida pelo usuário, até atingir o recurso *web* desejado.

Tabela 11 *- Trecho de Arquivo de Log com Tarefa*

```
187.101.253.34 - - [26/Aug/2012:11:47:48 -0300] "GET /site/restaurante.php HTTP/1.1" 200 6483
"http://each.uspnet.usp.br/site/instalacoes.php" "Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.1 (KHTML,
like Gecko) Chrome/21.0.1180.83 Safari/537.1"
187.101.253.34 - - [26/Aug/2012:11:47:48 -0300] "GET /site/eventos-box.php HTTP/1.1" 200 3364
"http://each.uspnet.usp.br/site/restaurante.php" "Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.1 (KHTML,
like Gecko) Chrome/21.0.1180.83 Safari/537.1"
187.101.253.34 - - [26/Aug/2012:11:47:49 -0300] "GET /favicon.ico HTTP/1.1" 404 746 "-" "Mozilla/5.0
(Windows NT 6.1) AppleWebKit/537.1 (KHTML, like Gecko) Chrome/21.0.1180.83 Safari/537.1"
187.101.253.34 - - [26/Aug/2012:11:47:52 -0300] "GET /site/restaurante.php?item=informacoes HTTP/1.1" 200
6483 "http://each.uspnet.usp.br/site/restaurante.php" "Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.1
(KHTML, like Gecko) Chrome/21.0.1180.83 Safari/537.1"
187.101.253.34 - - [26/Aug/2012:11:47:52 -0300] "GET /site/eventos-box.php HTTP/1.1" 200 3364
"http://each.uspnet.usp.br/site/restaurante.php?item=informacoes" "Mozilla/5.0 (Windows NT 6.1)
AppleWebKit/537.1 (KHTML, like Gecko) Chrome/21.0.1180.83 Safari/537.1"
187.101.253.34 - - [26/Aug/2012:11:47:53 -0300] "GET /favicon.ico HTTP/1.1" 404 746 "-" "Mozilla/5.0
(Windows NT 6.1) AppleWebKit/537.1 (KHTML, like Gecko) Chrome/21.0.1180.83 Safari/537.1"
187.101.253.34 - - [26/Aug/2012:11:47:55 -0300] "GET /site/restaurante.php?item=cardapio HTTP/1.1" 200
6337 "http://each.uspnet.usp.br/site/restaurante.php?item=informacoes" "Mozilla/5.0 (Windows NT 6.1)
AppleWebKit/537.1 (KHTML, like Gecko) Chrome/21.0.1180.83 Safari/537.1"
187.101.253.34 - - [26/Aug/2012:11:47:55 -0300] "GET /site/eventos-box.php HTTP/1.1" 200 3364
"http://each.uspnet.usp.br/site/restaurante.php?item=cardapio" "Mozilla/5.0 (Windows NT 6.1)
AppleWebKit/537.1 (KHTML, like Gecko) Chrome/21.0.1180.83 Safari/537.1"
187.101.253.34 - - [26/Aug/2012:11:47:55 -0300] "GET /favicon.ico HTTP/1.1" 404 746 "-" "Mozilla/5.0
(Windows NT 6.1) AppleWebKit/537.1 (KHTML, like Gecko) Chrome/21.0.1180.83 Safari/537.1"
187.101.253.34 - - [26/Aug/2012:11:47:57 -0300] "GET /infra/restaurante/cardapio-almoco.pdf HTTP/1.1" 200
46354 "http://each.uspnet.usp.br/site/restaurante.php?item=cardapio" "Mozilla/5.0 (Windows NT 6.1)
AppleWebKit/537.1 (KHTML, like Gecko) Chrome/21.0.1180.83 Safari/537.1"
```

O usuário faz acesso aos seguintes recursos *web*: “/site/restaurante.php”, “/site/restaurante.php?item=informacoes”, “/site/restaurante.php?item=cardapio”; até atingir seu alvo, o recurso *web* “/infra/restaurante/cardapio-almoco.pdf”.

Uma tarefa é considerada realizada se o usuário percorrer a sequência de recursos *web* determinada pelo avaliador, partindo do primeiro, indo por cada recurso *web* na ordem estabelecida, até atingir o último recurso *web*. Vale destacar que toda página *web* é um recurso *web*, mas o contrário não é verdadeiro.

Caso os usuários do *website* façam uso de IP dinâmico, dois usuários diferentes poderão usar o mesmo IP em momentos distintos. Então, independentemente de qual usuário estava realizando o acesso, a tarefa será considerada realizada, uma vez que o arquivo de *log* não fornece mecanismos para diferenciar esses dois usuários.

Além disso, deve ser levado em consideração que não se sabem quais tarefas os usuários estão tentando realizar a princípio. Este é o foco da análise. A análise dos *logs* também não deve ser a única estratégia para avaliar a usabilidade do *website*, mas uma das estratégias. O ideal é que ela seja sempre a avaliação inicial a ser realizada. Outras avaliações mais demoradas e custosas poderiam ser realizadas posteriormente para focar as partes do *website* onde foram percebidos indícios de problemas de usabilidade pela análise do arquivo de *log*.

Para ser iniciado este processo de avaliação, os *webdesigners*, ou projetistas do *website*, precisam indicar todos os recursos *web* que podem ser acessados com o uso do *website*. Desta forma ao ser identificada alguma página *web* embutida no arquivo de *log*, normalmente colocada dentro de estruturas do HTML conhecidas como *frames*, deve-se ter certeza que é um recurso *web* interno à página *web* acessada. Em outras palavras, não foi acessada outra página *web*.

Após o mapeamento das tarefas ter sido realizado, os recursos *web* relativos a ele, as páginas *web* de ajuda e erro, e os recursos *web* que descrevem as tarefas deste *website* são armazenados em um Banco de Dados. Desta forma, é possível a comparação entre estas informações já armazenadas pelo responsável pela avaliação e as informações identificadas ao se percorrer o arquivo de *log* a ser analisado.

A Figura 3 apresentada a seguir mostra a tela da ferramenta em que uma nova tarefa é registrada. Os recursos *web* que fazem parte do *website* a ser analisado são dispostas à esquerda. Elas devem ser selecionadas individualmente e movidas da esquerda para a direita, ou ao contrário, através do pressionamento dos botões centrais.

Figura 3 *– Tela para Cadastro de Nova Tarefa*

Após ser escolhido o nome para a nova tarefa, ela é armazenada no Banco de Dados através do pressionamento do botão localizado abaixo. A próxima seção abordará como foi planejado este processo na ferramenta.

4.1.2 Armazenamento em Banco de Dados

Para facilitar as avaliações, o mapeamento das tarefas de cada *website* é armazenado em um Banco de Dados. A ferramenta compara as entradas observadas no arquivo de *log* carregado com as entradas do Banco de Dados, possibilitando a realização do procedimento para qualquer *website* de posse destas duas entradas, ou seja, sem nenhuma informação específica do *website* armazenada em seu código.

No caso da ferramenta que está sendo desenvolvida, para que seja possível uma

análise histórica de avaliações, é necessário que cada avaliação seja armazenada. O uso de Banco de Dados é reforçado pela necessidade de comparação entre os históricos, para se extraírem conclusões.

O cadastro no Banco de Dados pode ser acessado e realizado pelo próprio responsável pela avaliação do *website*, por meio da ferramenta. Desta forma, apenas será necessário um cadastro inicial de um usuário para o avaliador.

A descrição dos relacionamentos presentes na ferramenta é explicitada através do Modelo Entidade-Relacionamento da mesma apresentada na Figura 4. Observa-se, nesta figura, que os usuários podem fazer a avaliação de *websites* ou tarefas, bem como as informações adicionais para cada um destes componentes ou para as avaliações em si.

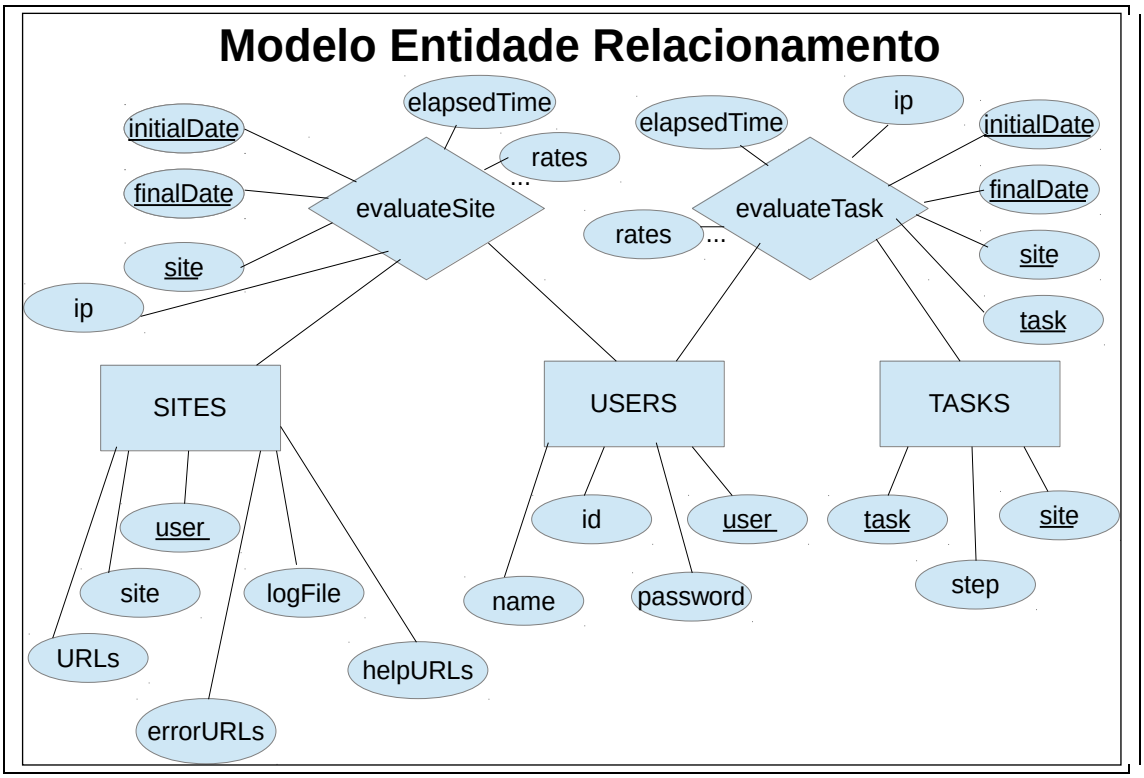

Figura 4 *– Modelo Entidade Relacionamento*

A estrutura das tabelas elaboradas para se armazenarem as informações de *websites* e tarefas, bem como das avaliações, segue abaixo. Inicialmente será apresentada cada tabela e

seu papel para o funcionamento da ferramenta. No Apêndice A, encontra-se a sequência de comandos para serem geradas novamente as tabelas.

As tabelas são as seguintes:

- tabela **evaluateSite**: armazena as taxas coletadas das avaliações dos *websites*, realizadas pelos usuários para um IP e para datas de início e fim determinadas;
- tabela **evaluateTask**: armazena as taxas coletadas das avaliações das tarefas dos *websites*, realizadas pelos usuários para um IP e para datas de início e fim determinadas;
- tabela **sites**: contém as informações relativas aos *websites*, ou seja, os recursos *web* que ele possui, suas páginas *web* de ajuda e erro, o nome de seu arquivo de *log*, o parâmetro que indica quantos segundos são um tempo muito longo e um nome de usuário que o possa avaliar;
- tabela **tasks**: contém as informações relativas às tarefas dos *websites*, ou seja, o *website* a que a tarefa pertence, o nome da tarefa, e o caminho ou sequência de recursos *web* necessária para realizar esta tarefa;
- tabela **usuários**: contém as informações sobre os usuários que poderão usar o sistema, ou seja, o nome de usuário, um identificador único, a senha e o nome da pessoa responsável pelo usuário.

As tabelas **evaluateSite** e **evaluateTask** refletem diretamente os relacionamentos entre as entidades **site**, **task** e **usuário**; sendo que estes últimos também acabam sendo descritos em tabelas. Para o cadastro inicial de um usuário de forma a lhe permitir o uso da ferramenta, precisam ser enviadas para o administrador as informações descritas na tabela **usuários**. Então é realizada a colocação manual deste conteúdo na ferramenta gerenciadora de Banco de Dados, conforme exemplo na Figura 5 a seguir.

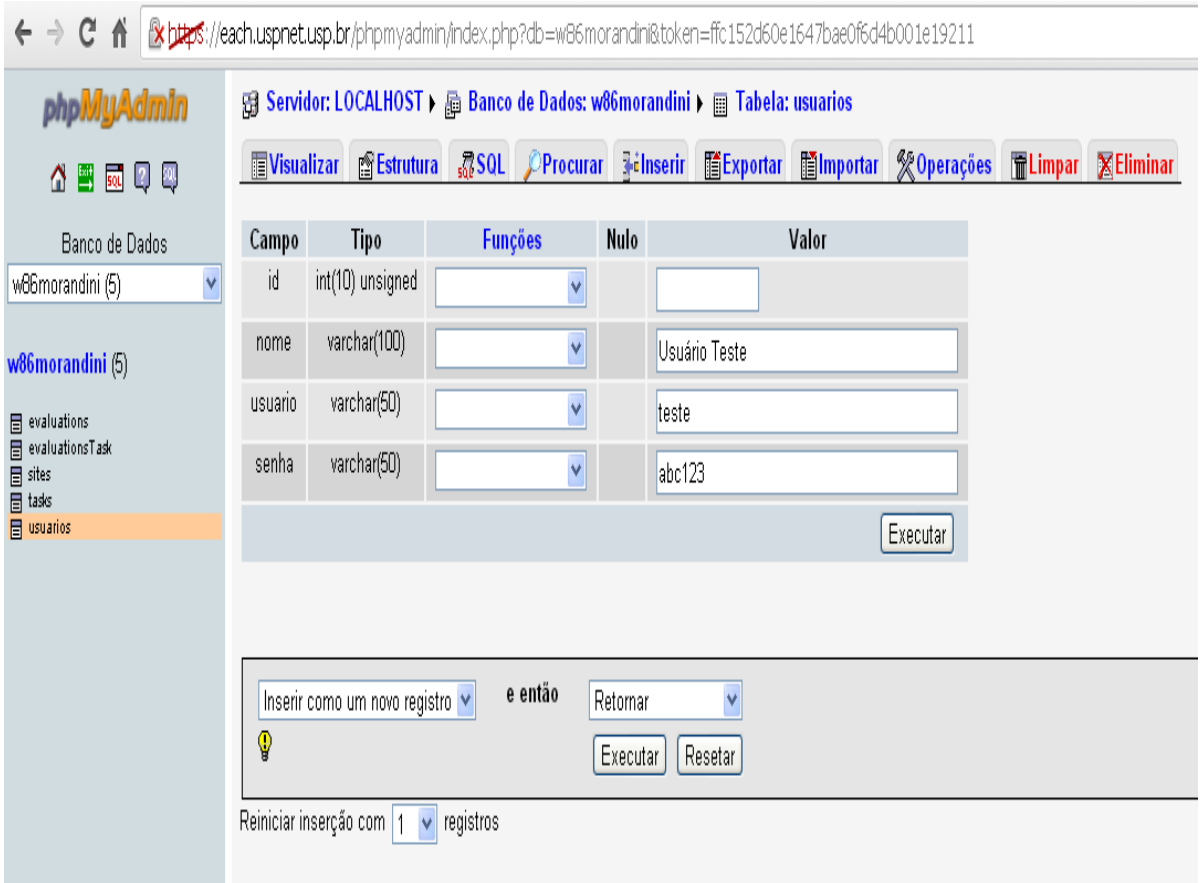

Figura 5 *- Colocação de Novo Usuário no Banco de Dados*

Além das informações acima, o administrador da ferramenta necessita de algumas das informações da tabela **sites** para que o usuário inicie o uso da mesma. A colocação destas outras informações está apresentada na Figura 6.

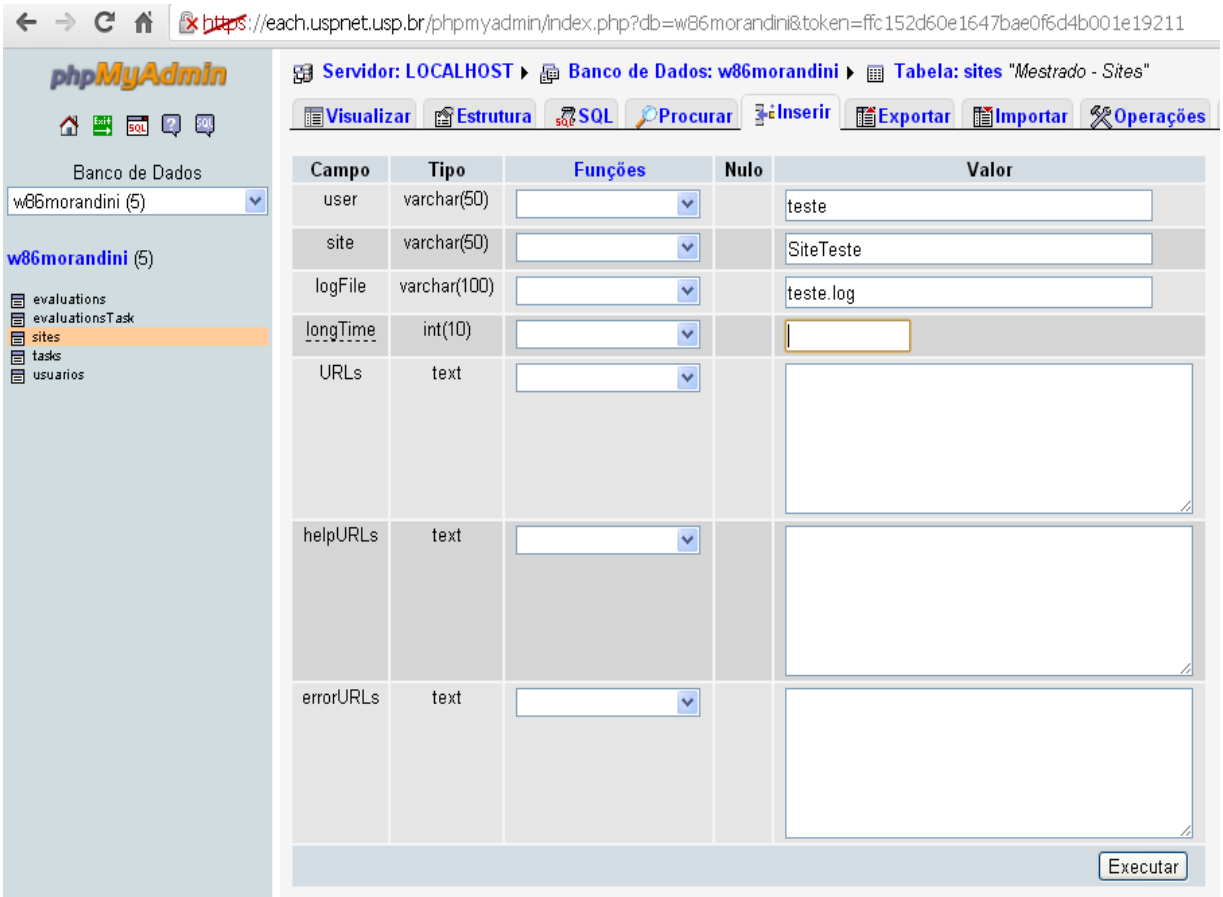

Figura 6 *– Colocação das Informações Básicas do Website no Banco de Dados*

Após estas ações, o usuário da ferramenta passa a cadastrar os recursos *web* que pertencem ao *website* e as tarefas que o descrevem, por meio da própria *interface* da ferramenta. Desta forma são complementadas as informações necessárias no Banco de Dados para permitir à ferramenta realizar as avaliações do *website* e das tarefas.

4.1.3 *Upload* de Arquivos de *Log* para o Servidor *Web*

Para poder ser usada dinamicamente, a ferramenta precisa aceitar que os arquivos de *log* sejam transferidos para o servidor, ou seja, que possa ser feito o *upload* deles. Dado o fato que os arquivos de *log* são, geralmente, grandes e com muitas entradas irrelevantes para o contexto deste trabalho, por não dizerem respeito aos recursos *web* necessários para o

emprego desta abordagem de avaliação. Mesmo se forem consideradas apenas as entradas úteis, o arquivo ainda pode ser grande e, portanto, apresentar dificuldades em sua manipulação.

Inicialmente, é necessário que se separe apenas o arquivo de *log* cujas entradas estejam dentro do período de tempo que se quer analisar. Caso seja tomado um arquivo com um período ainda mais abrangente, posteriormente a ferramenta filtra neste arquivo o período mais restritivo a ser analisado.

Para evitar que a ferramenta tenha que manipular arquivos muito grandes, pode ser aplicado um filtro no arquivo de *log* para serem analisadas apenas as entradas relativas a um determinado IP. Caso dois usuários usem o mesmo IP durante a navegação, serão considerados como se fosse um único usuário, e o foco fica nos recursos *web* acessados pelo IP em questão.

No Linux, a composição de comandos do Sistema Operacional que faz o filtro das entradas para um determinado IP e ainda para mais de um IP seguem abaixo. Ambos os comandos direcionam a saída para o arquivo teste.log.

Tabela 12 - *Comandos Linux para Filtrar IP*

```
cat default_access.log.3 | grep 127.0.0.217 » teste.log
cat default_access.log.3 | egrep '127.0.0.217|127.0.0.218' » teste.log
```

No Windows, o comando equivalente ao mencionado é o seguinte:

Tabela 13 - *Comandos Windows para Filtrar IP*

```
type default_access.log.3 | findstr 127.0.0.217 > teste.log
type default_access.log.3 | findstr “127.0.0.217 127.0.0.218” > teste.log
```

A seguir, é mostrado um exemplo de uso deste comando. As linhas apresentadas são um trecho de um arquivo de *log* para o qual será executado o código mostrado. Este trecho será gravado num arquivo de nome “trecho_type.log”.

Tabela 14 *- Trecho de Arquivo de Log para Filtrar IP*

```
66.249.71.237 - - [26/Aug/2012:08:23:39 -0300] "GET /infra/restaurante/cardapio-almoco.pdf HTTP/1.1" 304 -
"-" "Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)"
201.82.20.8 - - [26/Aug/2012:08:23:40 -0300] "GET /ppgsi/selecao.html HTTP/1.1" 200 6930
"http://www.each.usp.br/ppgsi/faq.html" "Mozilla/5.0 (Windows NT 5.1) AppleWebKit/537.1 (KHTML, like
Gecko) Chrome/21.0.1180.83 Safari/537.1"
201.82.20.8 - - [26/Aug/2012:08:23:42 -0300] "GET /favicon.ico HTTP/1.1" 404 741 "-" "Mozilla/5.0
(Windows NT 5.1) AppleWebKit/537.1 (KHTML, like Gecko) Chrome/21.0.1180.83 Safari/537.1"
41.70.155.37 - - [26/Aug/2012:08:23:39 -0300] "GET /site/imagem/fotos/21.jpg HTTP/1.1" 200 29790
"http://each.uspnet.usp.br/site/especializacao-psicologia-politica.php" "Mozilla/5.0 (compatible; MSIE 9.0;
Windows NT 6.1; Trident/5.0)"
201.81.238.240 - - [26/Aug/2012:08:23:48 -0300] "GET /estudosculturais HTTP/1.1" 301 309 "-" "Mozilla/5.0
(Windows; U; Windows NT 6.0; pt-BR; rv:1.9.2.15) Gecko/20110303 Firefox/3.6.15 ( .NET CLR 3.5.30729;
.NET4.0C) FBSMTWB"
201.81.238.240 - - [26/Aug/2012:08:23:48 -0300] "GET /estudosculturais/ HTTP/1.1" 200 5310 "-" "Mozilla/5.0
(Windows; U; Windows NT 6.0; pt-BR; rv:1.9.2.15) Gecko/20110303 Firefox/3.6.15 ( .NET CLR 3.5.30729;
.NET4.0C) FBSMTWB"
41.70.155.37 - - [26/Aug/2012:08:23:34 -0300] "GET /site/imagem/fotos/24.jpg HTTP/1.1" 200 27819
"http://each.uspnet.usp.br/site/especializacao-psicologia-politica.php" "Mozilla/5.0 (compatible; MSIE 9.0;
Windows NT 6.1; Trident/5.0)"
```

Apenas como exemplo, o comando adaptado que deverá ser executado neste trecho de arquivo de *log* para filtrar as entradas pertencentes ao IP 201.81.238.240 ficará da seguinte forma:

Tabela 15 *- Uso do Comando para Filtrar IP*

```
type trecho_type.log | findstr 201.81.238.240 > trecho_ip.log
```

O resultado da execução deste código no trecho apresentado irá ser armazenado num arquivo de nome “trecho_ip.log” e seu conteúdo é:

Tabela 16 *- Trecho de Arquivo de Log com IP Filtrado*

```
201.81.238.240 - - [26/Aug/2012:08:23:48 -0300] "GET /estudosculturais HTTP/1.1" 301 309 "-" "Mozilla/5.0
(Windows; U; Windows NT 6.0; pt-BR; rv:1.9.2.15) Gecko/20110303 Firefox/3.6.15 ( .NET CLR 3.5.30729;
.NET4.0C) FBSMTWB"
201.81.238.240 - - [26/Aug/2012:08:23:48 -0300] "GET /estudosculturais/ HTTP/1.1" 200 5310 "-" "Mozilla/5.0
(Windows; U; Windows NT 6.0; pt-BR; rv:1.9.2.15) Gecko/20110303 Firefox/3.6.15 ( .NET CLR 3.5.30729;
.NET4.0C) FBSMTWB"
```

Após a filtragem do arquivo de *log* em que são separadas as entradas específicas a

determinado IP, e durante o *upload* do arquivo de *log*, deve ocorrer a filtragem das entradas relevantes, as quais são armazenadas numa tabela do Banco de Dados. Tal filtragem tem como base a relação de recursos *web* do *website*, já armazenada no Banco de Dados. Por mais que se filtre o arquivo, para *websites* muito acessados, mesmo para um período curto de tempo, a quantidade de entradas ainda poderá ser grande.

O servidor Apache possui um limite aceitável de tempo para ser realizada uma requisição e, em determinadas situações, ao se tentar fazer o *upload* dos arquivos de *log* pela ferramenta, este tempo poderá ser excedido. Portanto, para se evitar isso, para o *upload* dos arquivos de *log* serão usados programas convencionais de transferência de arquivo, em vez de se fazer o *upload* pela própria ferramenta.

Com a finalidade de ilustrar este processo, seguem abaixo duas figuras com as telas de *upload* para um dos programas possíveis de serem utilizados.

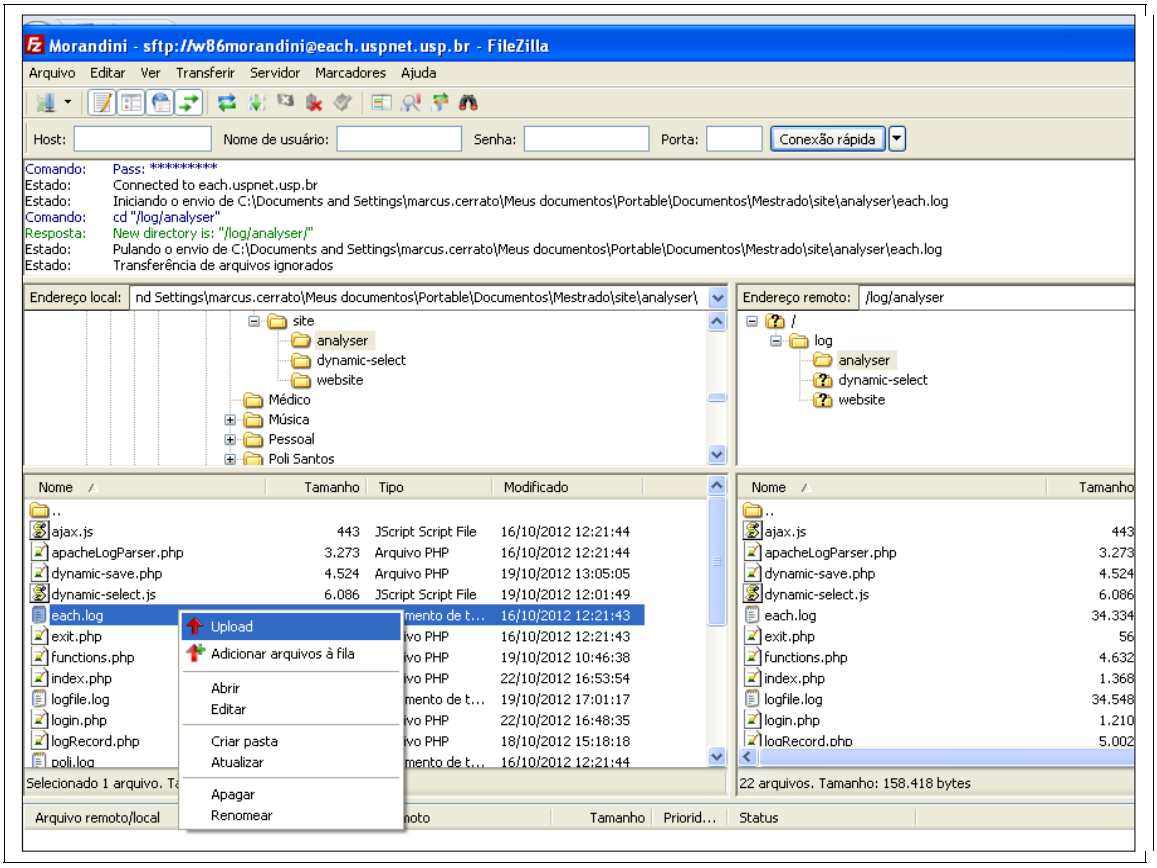

Figura 7 *– Tela da Ferramenta de Upload de Arquivo de Log*

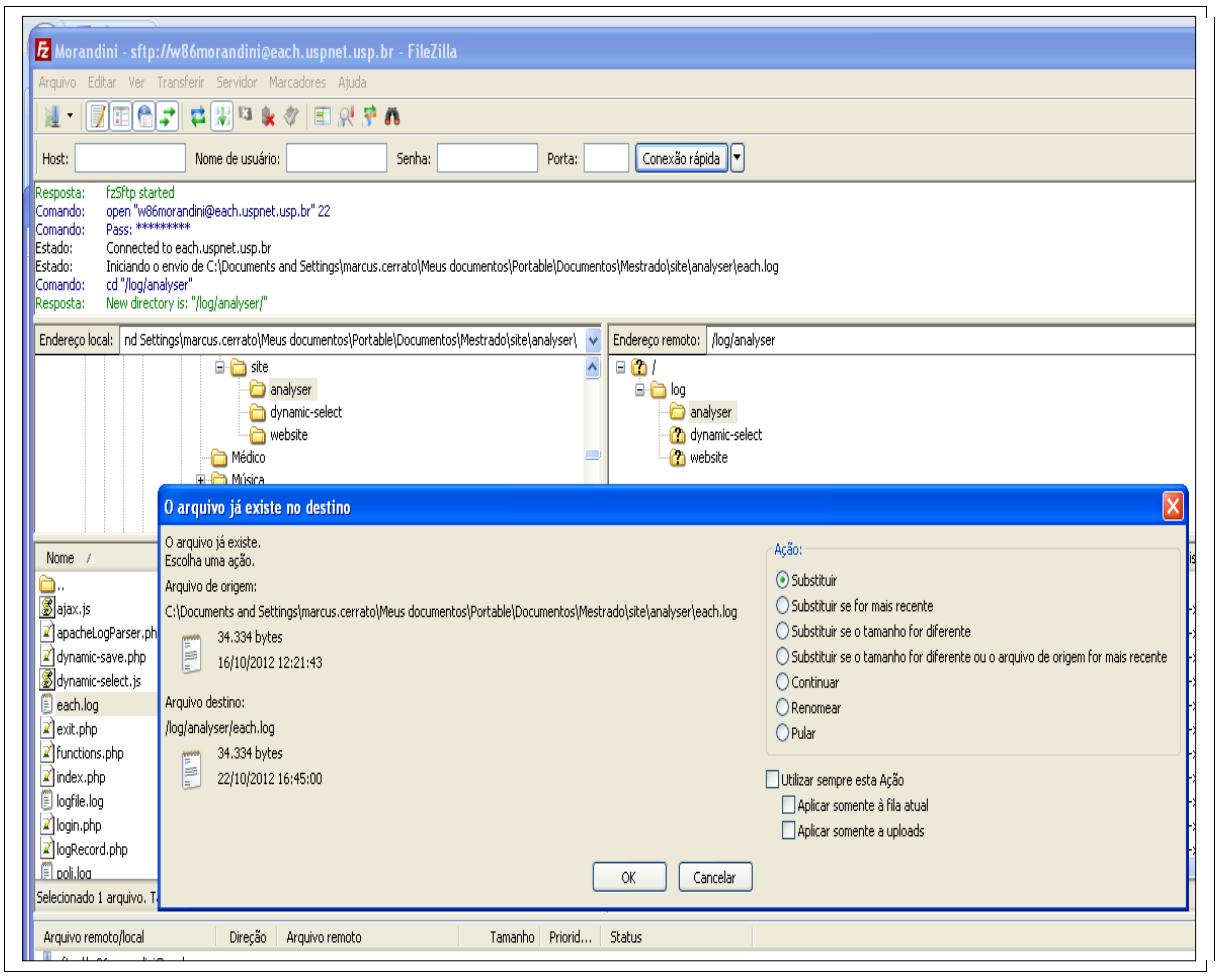

Figura 8 *– Upload de Arquivo de Log*

Na próxima seção, serão mostrados os resultados de uma avaliação realizada com a ferramenta em um *website*, bem como os arquivos de *log* correspondentes aos *websites* e as taxas coletadas e métricas calculadas neste processo.

4.2 Validação da Ferramenta

Para validar a ferramenta desenvolvida, foram realizadas algumas avaliações práticas com um arquivo de *log* real. O arquivo de *log* utilizado foi um dos arquivos do *website* da

EACH | USP, para os quais foi solicitada permissão de acesso, conforme já apresentado anteriormente. A seguir, será apresentado o trecho do arquivo de *log* utilizado para a validação da ferramenta implementada.

4.2.1 Exemplo de Avaliação

Com a finalidade de se validar a ferramenta ErgoMonitor aqui apresentada, foi realizada uma avaliação empregando-se o arquivo de *log* do servidor *web* da EACH | USP que continha dados de uso do *website* do início do mês de maio de 2011. Percebe-se que cada recurso *web* acessado é mostrado em um bloco de informações pertinentes a ele, tal qual apresentados na Tabela 17 a seguir.

Em cada bloco de informação mostrado, além do próprio endereço do recurso *web* dentro do *website*, são apresentados o status da requisição retornado pelo servidor *web*, o IP com o qual o acesso foi feito, o navegador e o Sistema Operacional usado pelo usuário e o endereço base de acesso. Isso pode ser observado ao se verificar que as URLs apresentadas se iniciam com a sequência "/site/".

Além disso, os blocos mostram o momento em que o acesso foi realizado, a diferença de tempo entre pares de acesso consecutivos, cada uma das tarefas a ser avaliada, e quanto já foi percorrido em cada tarefa, ou seja, quais recursos *web* já foram identificados a partir do recurso inicial para esta tarefa.

Uma mensagem indicando que a tarefa foi completada é apresentada, caso o recurso *web* final da tarefa tenha sido atingido. Paralelamente à contabilização das tarefas completadas, a tarefa cuja execução é interrompida também é contabilizada, conforme pode ser visto pela observação do texto "Complete" e "Incomplete".

A Tabela 17 apresenta uma parte dos resultados obtidos com a utilização do ErgoMonitor para a avaliação que está sendo apresentada. Destaca-se que foi apresentada apenas esta parte, para que o texto não fique muito extenso. Os rótulos para cada bloco, a saber "BLOCO XX" foram inseridos posteriormente para facilitar a referência às informações

nas explicações da subseção 4.2.2 a seguir.

Tabela 17 *- Exemplo de Avaliação*

BLOCO 01
URL = /site/docentes-nome.php | Status = 200 [successful response]
IP = 143.107.58.217
Date/Time = 03/May/2011 12:50:32 -0300 -->
Agent = "Mozilla/5.0 (Windows; U; Windows NT 5.1; pt-BR; rv:1.9.2.17) Gecko/20110420 Firefox/3.6.17"
Referer = "http://each.uspnet.usp.br/site/"
[TASK 1 - STEP 1 | /site/graduacao-comissao.php] (Complete = 0 | Incomplete = 0)
[TASK 2 - STEP 1 | /site/docentes-nome.php] (Complete = 0 | Incomplete = 0)

BLOCO 02
URL = /site/conteudo-docentes-nome.php | Status = 200 [successful response]
IP = 143.107.58.217
Date/Time = 03/May/2011 12:50:33 -0300 -->
DIFFERENCE = 0 dias 0 hours 0 minutes 1 seconds [de 03/May/2011 às 12:50:32 até 03/May/2011 às 12:50:33]
Agent = "Mozilla/5.0 (Windows; U; Windows NT 5.1; pt-BR; rv:1.9.2.17) Gecko/20110420 Firefox/3.6.17"
Referer = "http://each.uspnet.usp.br/site/docentes-nome.php"
[TASK 1 - STEP 1 | /site/graduacao-comissao.php] (Complete = 0 | Incomplete = 0)
[TASK 2 - STEP 2 | /site/conteudo-docentes-nome.php] (Complete = 0 | Incomplete = 0)

BLOCO 03
URL = /site/conteudo-docentes-nome.php?letra=M | Status = 200 [successful response]
IP = 143.107.58.217
Date/Time = 03/May/2011 12:50:41 -0300 -->
DIFFERENCE = 0 dias 0 hours 0 minutes 8 seconds [de 03/May/2011 às 12:50:33 até 03/May/2011 às 12:50:41]
Agent = "Mozilla/5.0 (Windows; U; Windows NT 5.1; pt-BR; rv:1.9.2.17) Gecko/20110420 Firefox/3.6.17"
Referer = "http://each.uspnet.usp.br/site/conteudo-docentes-nome.php?letra=B"
[TASK 1 - STEP 1 | /site/graduacao-comissao.php] (Complete = 0 | Incomplete = 0)
[TASK 2 - STEP 3 | /site/conteudo-docentes-nome.php?letra=M] (Complete = 0 | Incomplete = 0)

BLOCO 04
URL = /site/conteudo-docentes-nome.php?letra=M&codigo=1364328 | Status = 200 [successful response]
IP = 143.107.58.217
Date/Time = 03/May/2011 12:50:47 -0300 -->
DIFFERENCE = 0 dias 0 hours 0 minutes 6 seconds [de 03/May/2011 às 12:50:41 até 03/May/2011 às 12:50:47]
Agent = "Mozilla/5.0 (Windows; U; Windows NT 5.1; pt-BR; rv:1.9.2.17) Gecko/20110420 Firefox/3.6.17"
Referer = "http://each.uspnet.usp.br/site/conteudo-docentes-nome.php?letra=M"
[TASK 1 - STEP 1 | /site/graduacao-comissao.php] (Complete = 0 | Incomplete = 0)
[TASK 2 - STEP 4 | /site/conteudo-docentes-nome.php?letra=M&codigo=1364328] -- TASK 2 COMPLETED --
(Complete = 1 | Incomplete = 0)

BLOCO 05
URL = /site/docentes-nome.php | Status = 200 [successful response]
IP = 143.107.58.217
Date/Time = 03/May/2011 12:55:26 -0300 -->
DIFFERENCE = 0 dias 0 hours 4 minutes 39 seconds [de 03/May/2011 às 12:50:47 até 03/May/2011 às 12:55:26]
Agent = "Mozilla/5.0 (Windows; U; Windows NT 5.1; pt-BR; rv:1.9.2.17) Gecko/20110420 Firefox/3.6.17"
Referer = "http://each.uspnet.usp.br/site/docentes-nome.php"
[TASK 1 - STEP 1 | /site/graduacao-comissao.php] (Complete = 0 | Incomplete = 0)
[TASK 2 - STEP 1 | /site/docentes-nome.php] (Complete = 1 | Incomplete = 0)

BLOCO 06

```
URL = /site/conteudo-docentes-nome.php | Status = 200 [ successful response ]
IP = 143.107.58.217
Date/Time = 03/May/2011 12:55:26 -0300 -->
DIFFERENCE = 0 dias 0 hours 0 minutes 0 seconds [ de 03/May/2011 às 12:55:26 até 03/May/2011 às 12:55:26 ]
Agent = "Mozilla/5.0 (Windows; U; Windows NT 5.1; pt-BR; rv:1.9.2.17) Gecko/20110420 Firefox/3.6.17"
Referer = "http://each.uspnet.usp.br/site/docentes-nome.php"
[TASK 1 - STEP 1 | /site/graduacao-comissao.php] (Complete = 0 | Incomplete = 0)
[TASK 2 - STEP 2 | /site/conteudo-docentes-nome.php] (Complete = 1 | Incomplete = 0)

BLOCO 07
URL = /site/graduacao-comissao.php | Status = 200 [ successful response ]
IP = 143.107.58.217
Date/Time = 03/May/2011 12:57:09 -0300 -->
DIFFERENCE = 0 dias 0 hours 1 minutes 43 seconds [ de 03/May/2011 às 12:55:26 até 03/May/2011 às 12:57:09 ]
Agent = "Mozilla/5.0 (Windows; U; Windows NT 5.1; pt-BR; rv:1.9.2.17) Gecko/20110420 Firefox/3.6.17"
Referer = "http://each.uspnet.usp.br/site/docentes-nome.php"
[TASK 1 - STEP 1 | /site/graduacao-comissao.php] (Complete = 0 | Incomplete = 0)
[TASK 2 - STEP 3 | /site/conteudo-docentes-nome.php?letra=M] (Complete = 1 | Incomplete = 1)

BLOCO 08
URL = /site/conteudo-graduacao-comissao-pautas.php | Status = 200 [ successful response ]
IP = 143.107.58.217
Date/Time = 03/May/2011 12:57:09 -0300 -->
DIFFERENCE = 0 dias 0 hours 0 minutes 0 seconds [ de 03/May/2011 às 12:57:09 até 03/May/2011 às 12:57:09 ]
Agent = "Mozilla/5.0 (Windows; U; Windows NT 5.1; pt-BR; rv:1.9.2.17) Gecko/20110420 Firefox/3.6.17"
Referer = "http://each.uspnet.usp.br/site/graduacao-comissao.php"
[TASK 1 - STEP 2 | /site/conteudo-graduacao-comissao-pautas.php] -- TASK 1 COMPLETED -- (Complete = 1 |
Incomplete = 0)
[TASK 2 - STEP 1 | /site/docentes-nome.php] (Complete = 1 | Incomplete = 1)
```

Na próxima seção, o resultado da avaliação acima é apresentado e analisado. Assim as taxas foram coletadas e as métricas de usabilidade, calculadas.

4.2.2 Definição das Taxas e Métricas Calculadas e Análise do Resultado da Avaliação

Inicialmente é apresentada a explicação de cada taxa coletada na avaliação, ao se empregar este processo. Estas taxas foram escolhidas por serem bastante significativas e estarem bastante relacionadas com a usabilidade e seus atributos.

- Taxa de URLs atingidas (*Hit Rate*): Indica cada URL que estava dentro das condições de busca, como por exemplo, dentro do intervalo de datas e horários de início e fim, pertencente ao *website* avaliado.

- Taxa de Tarefa Completada (*Task Complete Rate*): Mostra quantas vezes determinada tarefa foi completada, ou seja, todas as suas URLs foram percorridas, da primeira até a última, sem interrupções e na ordem inicialmente estabelecida. O valor da taxa para cada tarefa é exibido e, a seguir, a soma das taxas para todas as tarefas é apresentada.
- Taxa de Tarefas Não-Completadas (*Task Incomplete Rate*): Mostra quantas vezes determinada tarefa foi iniciada, mas não foi completada, ou seja, quando as suas URLs foram percorridas, a partir da primeira e na ordem inicialmente estabelecida, mas não tendo sido feito o percurso até a última. O valor da taxa para cada tarefa é exibido e, a seguir, a soma das taxas para todas as tarefas é apresentada.
- Taxa de Nenhuma Tarefa (*No Task Rate*): Relaciona as ocasiões em que nenhuma tarefa foi percorrida ou iniciada, ou seja, aquela URL não faz parte de nenhuma sequência de URLs das tarefas na ordem em que vinham sendo observadas. Se esta taxa for muito grande, sugere-se que outras tarefas sejam incluídas, mesmo que menores em quantidade de URLs, para que estes momentos sejam capturados por estas novas tarefas.
- Taxa de Mudança no Campo Agent (*Agent Change Rate*): O campo "Agent" do arquivo de *log* corresponde às informações do navegador e do Sistema Operacional que foram utilizados. Portanto, esta taxa pode indicar momentos em que o usuário teve alguma dificuldade em acessar alguma funcionalidade do *website* e optou por tentar novamente com algum outro recurso tecnológico.
- Taxa de Tempo Muito Grande Atingido (*Long Time Rate*): Esta taxa identifica os momentos em que se passou um tempo muito grande sem qualquer atividade de navegação. Este intervalo pode revelar que o usuário realmente requisitou muito tempo naquela interação, ou pode ter se ausentado por algum motivo. De qualquer forma, esta taxa deve ser calculada e inserida no contexto deste processo de avaliação.

Assim, tendo como base nas taxas coletadas, a ferramenta calcula as métricas correspondentes pela divisão da taxa em questão pelo total de acessos. Como exemplos, para

esta avaliação, a tarefa1 teve a taxa de tarefas completadas valendo 1 para um total de 20 acessos. Portanto a métrica de tarefas completadas calculada é 0,05 (ou seja, 1 dividido por 20). Temos ainda que o total de tempo gasto com a tarefa 1 para as tarefas completadas foi de 103 segundos, o que conduz a uma métrica de 5,15 (ou seja, 103 dividido por 20).

Algumas das métricas calculadas, mais especificamente as que foram exemplificadas acima, estão diretamente relacionadas com a eficácia na tarefa, ou seja, quantidade de usuários com sucesso na tarefa dividido pelo total de acessos na tarefa (*task complete rate / hit rate*); e com a eficiência na tarefa, que é a média dos tempos de comportamento sucesso na tarefa dividido pelo total de acessos na tarefa (*total elapsed time for complete tasks / hit rate*).

Além disso, estas métricas calculadas permitem que sejam obtidos valores que possam dar subsídios para uma avaliação remota e automatizada do *website*, pois capturam, para cada tarefa e para todos os usuários, como está o uso das funcionalidades do *website* expressas por meio de suas tarefas mapeadas.

Finalmente, a Tabela 18 relaciona as taxas coletadas para a avaliação completa, em que 20 recursos *web* foram encontrados. Além disso, em 5 entradas não ocorre o prosseguimento de nenhuma das tarefas em questão, ou seja, para nenhuma tarefa ocorreu o próximo passo que era esperado.

Comparando-se os dados apresentados nesta tabela com os da Tabela 17, podemos observar que, na Tabela 17, nos blocos 4 e 8 percebe-se que as tarefas 2 e 1 são cumpridas, respectivamente. Já no bloco 7, a tarefa 2 foi interrompida. Isso aconteceu mais duas vezes na avaliação completa, conforme pode ser observado pelo texto “Task Incomplete Rate Sum Task 2 = 3” da Tabela 18.

Tabela 18 *- Resultado de Avaliação*

COLLECTED RATES	CALCULATED METRICS
Hit Rate = 20 No Task Rate = 6 Help Rate = 1 Error Rate = 0	No Task Metric = 0,3000 Task Complete Metric Sum = 0,1000

Task Complete Rate Sum = 2 Task 1 = 1 Task 2 = 1 Task Incomplete Rate Sum = 3 Task 1 = 0 Task 2 = 3 Long Time Rate Sum = 0 Task 1 = 0 Task 2 = 0 Change Agent Rate Sum = 0 Task 1 = 0 Task 2 = 0 Total Elapsed Time = 295200 Total No Task Elapsed Time = 20246 Complete tasks Task 1 = 103 Task 2 = 15 Incomplete tasks Task 1 = 0 Task 2 = 274836	Task 1 = 0,0500 Task 2 = 0,0500 Task Incomplete Metric Sum = 0,1500 Task 1 = 0,0000 Task 2 = 0,1500 Long Time Metric = 0,0000 Task 1 = 0,0000 Task 2 = 0,0000 Agent Change Metric = 0,0000 Task 1 = 0,0000 Task 2 = 0,0000 Total Elapsed Time = 14.760,0000 Total No Task Elapsed Time = 1.012,3000 Complete tasks Task 1 = 5,1500 Task 2 = 0,7500 Incomplete tasks Task 1 = 0,0000 Task 2 = 13.741,8000

A ferramenta ainda faz a avaliação para uma tarefa específica ou ainda para um arquivo de *log* com vários IPs. Neste último caso, é obtida ainda a soma para as taxas individuais dos IPs.

No último capítulo serão apresentadas as considerações finais deste trabalho e sua contribuição até o momento.

5 *Conclusões*

Neste capítulo constam as considerações finais a respeito do trabalho e ainda algumas conclusões que se desprendem das atividades realizadas para tornar seu resultado possível. Além disso, algumas atividades estão relacionadas como sugestão para dar continuidade ao que foi aqui apresentado, com o objetivo de acrescentar ou melhorar o que está sendo entregue.

A expectativa é que, com o uso da ferramenta, os valores das taxas e métricas obtidos aproximem-se dos padrões esperados por meio da melhoria na realização do mapeamento de tarefas e ainda no aprimoramento dos próprios *websites* avaliados, após a identificação e correção de possíveis problemas de usabilidade encontrados.

Este capítulo esta dividido nas seguintes seções: considerações finais e trabalhos futuros.

5.1 Considerações Finais

A usabilidade está presente nos *websites* e tem se tornado parte importante, não apenas no seu desenvolvimento, como na sua utilização [Winckler & Pimenta, 2013]. Avaliar e desenvolver a usabilidade de *websites* pode garantir a eles um uso constante. Entre as diversas possibilidades de avaliar a usabilidade dos *websites*, este trabalho concentrou-se num método automatizado e com o qual o *website* possa ser avaliado com base em um contexto cotidiano de uso, para contemplar situações reais. Embora não exclua as outras avaliações, o método pretende destacar pontos nos quais as avaliações mais detalhadas e custosas deveriam se concentrar.

Este trabalho teve como meta obter taxas e calcular métricas a partir de informações contidas em arquivos de *log* de servidores *web*. As tarefas do *website* a ser avaliado foram mapeadas e, por meio das entradas do arquivo de *log*, pôde ser identificado se cada tarefa foi

ou não realizada e quanto tempo esta ação demorou. Além disso, podemos coletar taxas relacionadas a sucesso ou não na realização das tarefas, que poderão fornecer indicativos de possíveis problemas de usabilidade para o avaliador do *website* em questão.

Para tanto, determinar um modelo de tarefas foi fundamental, para que se pudessem definir quais tarefas seriam analisadas e, assim, procurar verificar a conformidade das mesmas. Ainda, a identificação das tarefas realizadas com ou sem sucesso, pôde servir como base para que outras ferramentas de apoio à avaliação da usabilidade pudessem ser definidas, uma vez que esses dados servem como entrada para as mesmas.

Esta abordagem proposta possui um interesse bastante significativo que é o de evitar possíveis constrangimentos aos usuários, uma vez que não é necessário que se mantenha contato direto com o usuário, tal como ocorre ao serem empregadas outras técnicas e metodologias de avaliação, como por exemplo: aplicação de questionários ou filmagens.

A utilização desta ferramenta não é muito complicada para os responsáveis pela avaliação. Espera-se que isso permita estimular o seu uso mais frequente por diversos desenvolvedores, de modo a contribuir para a formação de um histórico de valores para taxas e métricas de usabilidade entre os *websites*.

Encontrar o melhor mapeamento de tarefas é a atividade mais importante de todo o processo aqui relacionado e interfere diretamente no resultado final, a saber, as taxas encontradas e as métricas calculadas. No entanto, o que se deve ter em mente é que nem todas as tarefas necessariamente devem começar da página inicial. Um exemplo disto é quando o recurso *web* final de uma tarefa for acessível a partir do menu superior do *website*, não importando em qual página *web* se esteja.

Durante o mapeamento de tarefas, o *website* deve ser enxergado como vários caminhos com algumas alternativas de continuação cada um. A filosofia que se deve ter em mente é pegar os trechos de caminho mais significativos, não tão longos nem tão curtos, para conseguir capturar as situações cotidianas de navegação do *website*.

A seguir são apresentados possíveis trabalhos que podem dar continuidade às atividades desenvolvidas até o momento de forma a melhorar esse processo de avaliação da

usabilidade de *websites* empregando-se o ErgoMonitor.

5.2 Trabalhos Futuros

Como sugestão para possíveis trabalhos futuros, melhorias ou novas abordagens poderiam ser feitas neste trabalho, de modo a aumentar sua funcionalidade ou melhorar seu aspecto visual. Algumas delas são descritas abaixo:

- **Melhoria na *interface***: A *interface* da ferramenta foi desenvolvida com foco nas suas funcionalidades, sem preocupação com o aspecto visual. Em outras palavras, o objetivo era implementar a lógica de programação, da forma mais simples possível. Podem ser agregados alguns recursos visuais que tornem o uso da ferramenta mais agradável e prático.
- **Realização de novos testes:** Nos *websites* informacionais, como aquele para o qual os testes foram realizados, foi considerada a página *web* de ajuda como a página onde aparece o mapa do *website*. Não foram feitos testes com arquivos de *log* de um *website* transacional, onde ocorram páginas *web* de erro. No entanto, a lógica que foi utilizada para a página de ajuda do *website* informacional não é diferente da utilizada para a página de erro do *website* transacional e já está inclusive implementada na ferramenta.
- **Configuração do Parâmetro 'Tempo Muito Longo'**: O parâmetro que indica quando um tempo muito longo decorreu entre duas entradas do arquivo de *log* pode ser melhor calibrado. Além disso, seu valor pode variar entre *websites* de acordo com as características de cada um. Um *website* mais dinâmico pode exigir um tempo mais curto de resposta do que outro que demande mais reflexão entre uma página e outra. As primeiras avaliações vão indicar ao usuário os melhores valores para seu *website*. A realização de mais testes podem ajudar a validar esta

configuração.

- **Agregação de Outros Parâmetros**: Conforme a ferramenta vá sendo utilizada, os administradores, ou seja, aqueles que são responsáveis pela configuração da ferramenta para uso, podem sentir a necessidade de incluir outros parâmetros para melhor configurar a ferramenta nas avaliações. Isso pode ser feito pela inclusão de novos campos na tabela com os dados do *website* e a respectiva programação para os mesmos serem considerados na avaliação.
- **Taxas Coletivas**: As taxas individuais são coletadas na avaliação do *website*, mas poderiam ser calculadas taxas coletivas, para desconsiderar os valores que estivessem muito distantes da média dos valores.
- **Expressão Regular para valor 'Agent'**: O valor 'Agent' extraído do arquivo de *log* contém as informações de qual Sistema Operacional e de qual Navegador está sendo realizado o acesso, bem como das respectivas versões de cada um. Neste trabalho esta informação foi tratada como sendo única. Pode ser feita uma expressão regular que separe as partes deste valor para alguma utilidade específica.
- **Usar informação do campo 'Referer':** A informação 'Referer' está no arquivo de *log* para indicar de onde veio a requisição do recurso *web*. Isso pode ser usado para contornar o problema causado pelo uso do cache pelos navegadores ou pelo servidor *web*. Se o começo da tarefa estava armazenado em cache, mas o usuário realizou a tarefa toda, este detalhe pode ser percebido pelo 'Referer'.
- **Coleta automática dos recursos *web***: Através de uma navegação direcionada pelos recursos *web* do *website*, seriam armazenados os recursos *web* no Banco de Dados entre as informações do *website* na tabela **sites**.
- **Permitir o uso para outros formatos de *log***: Por meio de algumas adaptações no ErgoParser, podem ser tratados outros formatos de arquivos de *log* para servidores diferentes do Apache.

A seguir é relacionada a referência bibliográfica empregada nesta dissertação de

mestrado que serviram, tanto para motivar a proposta de aplicação do ErgoParser, como para uma estrutura de apoio à avaliação da usabilidade de *websites*, quanto para dar suporte à proposição e ao desenvolvimento deste ambiente.

Referências

[Apache, 2011] *Apache HTTP Server Documentation - Log Files.* Disponível em: <http://httpd.apache.org/docs/2.2/logs.html>. Acessado em Novembro de 2011.

[Apache, 2011b] *Apache Module mod_log_config.* Disponível em: <http://httpd.apache.org/docs/2.2/mod/mod_log_config.html#logformat>. Acessado em: Novembro de 2011.

[BasicoWeb, 2012] *Códigos HTTP de Resposta mais Comuns e seus Significados.* Disponível em: <http://www.obasicodaweb.com/http-codigos-de-resposta-mais-comuns-e-seus-significados/>. Acessado em: Janeiro de 2012.

[Bevan et al., 1991] BEVAN, N.; KIRAKOWSKI, J.; MAISSEL, J. What is Usability? In: *Proceedings of the 4th International Conference on HCI*, Stuttgart, Setembro, 1991

[Brajnik, 2000] BRAJNIK, G. Automatic Web Usability Evaluation: What Needs To Be Done. In: *Proc. of 6th Human Factors and the Web Conference*, Austin, Texas - EUA, Junho, 2000.

[Castillo et al., 1997] CASTILLO, José C.; HARTSON, H. R.; HIX, Debora. Remote Usability Evaluation At A Glance. 1997. Disponível em: *<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.40.5299&rep=rep1&type=pdf>*. Acessado em Dezembro, 2010.

[CCP, 2007] CENTER FOR CIVIC PARTNERSHIPS. *Quantitative and Qualitative Evaluation Methods*. Disponível em: <http://www.civicpartnerships.org/docs/tools_resources/Quan_Qual%20Methods%209.07.htm>. Acessado em: Janeiro de 2013.

[Cleardesign, 2011] *Curso sobre Usabilidade na Web*. Disponível em: <http://cleardesign.net/webusability/attributes.html>. Acessado em:Novembro de 2011.

[Cybis et al.. 2007] CYBIS, W. A.; BETIOL, A. H.; FAUST, R. *Ergonomia e Usabilidade: conhecimentos, métodos e aplicações.* Novatec Editora, São Paulo, 2007.

[Cybis et al., 2002] CYBIS, W. A.; SCAPIN, D. L.; ANDRES, D. P.; MORANDINI, M. ErgoCoin: a usability inspection tool for web sites evaluation. INRIA – Institut National de Recherche en Informatique et en Automatique, 2002.

[Fernandez & Hussmann, 2009] FERNANDEZ, F. B.; HUSSMANN, H. *A Methodology and Framework to Simplify Usability Analysis of Mobile Applications.* Em: 2009 IEEEACM International Conference on Automated Software Engineering.

[Garzotto et al., 1998] GARZOTO, F.; MATERA, M.; PAOLINI, P. *To Use or Not to Use? Evaluating Usability of Museum Web Sites*. Em II Conferência Internacional de Museus. Disponível em <http://www.museumsandtheweb.com/mw98/papers/garzotto/garzotto_paper.html>

[Hartson et al., 2000] HARTSON, H. R.; ANDRE, Terence S.; WILLIGES, Robert C. Criteria for evaluating usability evaluation methods. *International Journal of Human-Computer Interaction, vol. 13*, 2001, pp. 373-410.

[ISO 9126] ISO/DIS 9126. *International Standard ISO/IEC 9126. Information technology - Software product evaluation - Quality characteristics and guidelines for their use.* International Organization for Standardization, International Electrotechnical Commission, Geneva.

[ISO 9241] ISO/DIS 9241. *Dialogue Principles in Guiding the Evaluation of User Interfaces - Guidance on Usability.* 1997.

[ISO25010, 2012] *Software has a new quality standard: ISO 25010.* Developing Webapplications. Disponível em: <http://ryreitsma.blogspot.com.br/2011/07/software-has-new-quality-model-iso.html>. Accessado em: Outubro, 2012.

[Ivory & Hearst, 2001] IVORY, Melody Y.; HEARST, Marti A. The State of the Art in Automating Usability Evaluation of User Interfaces. In: *ACM Computing Surveys, Vol. 33, No. 4*, December 2001, pp. 470-516. University of California, Berkeley.

[Jordan, 2002] JORDAN, P. W. *An Introduction to Usability.* Londres, Taylor & Francis Ltd., 2002.

[Labiutil, 2012] *Perspectiva da Avaliação: Objetivos de uma Avaliação de Usabilidade.* Disponível em: <http://www.labiutil.inf.ufsc.br/hiperdocumento/unidade3_3.html>. Acessado em: Novembro de 2012.

[Mairs, 2012] MAIRS, Jane (editor). *Merriam Webster Learner's Dictionary*. Disponível em: <http://www.learnersdictionary.com/>. Acessado em: Agosto de 2012.

[Moran, 1981] MORAN, T. (1981) "The Command Language Grammars: a representation for the user interface of interactive computer systems". Em *International Journal of Man-Machine Studies* 15:3-50, Academic Press London

[Morandini, 2002] MORANDINI, Marcelo. *ErgoMonitor: Monitoramento da Usabilidade em Ambiente Web por Meio da Análise de Arquivos de Log.* Florianópolis: UFSC, 2003. 172 p.

Tese (Doutorado) - Programa de Pós-Graduação em Engenharia de Produção e Sistemas, Universidade Federal de Santa Catarina, Florianópolis, 2003.

[Netcraft 2011] *Website de Notícias da Netcraft*. Disponível em: <http://news.netcraft.com>. Acessado em: Dezembro de 2011.

[Nielsen, 2000] NIELSEN, J. *Projetando Web Sites - Designing Web Usability.* Campinas: Editora Campus, 2000.

[Nielsen, 2000] NIELSEN, J. *Designing Web Usability*. California, New Riders, 2000.

[Nielsen, 2007] NIELSEN, J; LORANGER, H.. Usabilidade na Web - Projetando Websites com Qualidade. Rio de Janeiro, Elsevier, 2007.

[Nielsen, 2012] NIELSEN, J. Heuristic Evaluation. Disponível em: <*http://www.useit.com/ papers/heuristic/*>. Acessado em Agosto, 2012.

[Orfali et al., 1996] ORFALI, R.; HARKEY, D.; EDWARDS, J. The Essential ClientServer Survival Guide (2nd edition). New York, Wiley Computer Publishing, 1996.

[Preece et al.. 1994] PREECE, J., ROGERS, Y., SHARP, H., BENYON, D., HOLLAND, S., CAREY, T. Human Computer Interaction, Addison-Wesley, 1994.

[Regexp, 2011] Regular Expressions. Disponível em: <http://www.regular-expressions.info>. Acessado em: Novembro de 2011.

[RNP, 2012] *Introdução ao Cache de Web.* Disponível em: <http://www.rnp.br/newsgen/

0003/cache.html>. Acessado em: Janeiro de 2012.

[Rocha e Baranauskas, 2000] ROCHA, H. V.; BARANAUSKAS, M. C. *Design e Avaliação de Interfaces Humano-Computador*. Escola de Computação, USP, São Paulo, 2000.

[Sauro, 2011] SAURO, Jeff. 10 Essential Usability Metrics. In: Measuring Usability, 30/novembro/2011. Disponível em: <http://www.measuringusability.com/blog/essential-metrics.php>. Acessado em: Outubro, 2012.

[Scapin & Bastien, 1997] SCAPIN, D. L.; BASTIEN, J. M. C. Ergonomic Criteria for Evaluating the Ergonomic Quality of Interactive Systems. In: *Behaviour and Information Technology, vol 16*, no. 4/5, 1997.

[SearSOA, 2012] Parser Definition. Disponível em: < http://searchsoa.techtarget.com/ definition/parser>. Acessado em Outubro, 2012.

[Shneiderman, 1992] SHNEIDERMAN, B.; Designing the User Interface: Strategies for Effective Human-Computer Interaction, Addison-Wesley, 1992, 2ª. Ed.

[Scholtz, 2004] SCHOLTZ, Jean. Usability Evaluation. National Institute of Standards and Technology, 2004. Disponível em: <http://www.itl.nist.gov/iad/IADpapers/2004/Usability %20Evaluation_rev1.pdf>. Acessado em: Dezembro, 2010.

[Tec-Ed, 1999] Tec-Ed Inc. *Assessing Web Site Usability from Server Log Files*. White Paper, Dezembro, 1999.

[TechTerms, 2012] TechTerms.com. Disponível em: <http://www.techterms.com/category/

internet> com os termos “web server”, “client”, “website”, “web page”, “WWW”, e “Internet”. Acessado em: Outubro, 2012.

[Williams & Lane, 2004] WILLIAMS, H. E.; LANE, D. Web Database Applications with PHP and MySQL (2nd edition). California, O’Reilly, 2004.

[Winckler & Pimenta, 2013] WINCKLER, M.; PIMENTA, M. S. Avaliação de Usabilidade de Sites Web. Disponível em: <http://www.funtec.org.ar/usabilidadsitiosweb.pdf>. Acessado em: fevereiro/2013.

APÊNDICE A

Estruturas das Tabelas do Banco de Dados

Neste apêndice são relacionadas as estruturas das tabelas que compõem a ferramenta e permitem sua execução. Elas estão apresentadas na linguagem SQL, utilizada nos bancos de dados para gerar internamente tabelas.

Percebe-se que as tabelas de avaliação, isto é as tabelas “evaluations” e “evaluationsTask”, só diferem no campo “task” existente na segunda.

```
--
-- Estrutura da tabela `evaluations`
--

CREATE TABLE IF NOT EXISTS `evaluations` (
  `id` timestamp NOT NULL default CURRENT_TIMESTAMP on update CURRENT_TIMESTAMP,
  `site` varchar(50) NOT NULL,
  `ips` text NOT NULL,
  `hitRate` int(10) NOT NULL,
  `noTaskRate` int(10) NOT NULL,
  `taskCompleteRate` int(10) NOT NULL,
  `taskIncompleteRate` int(10) NOT NULL,
  `agentChangeRate` int(10) NOT NULL,
  `longTimeRate` int(10) NOT NULL,
  `initialDate` date NOT NULL,
  `finalDate` date NOT NULL,
  `elapsedTime` int(10) NOT NULL,
  PRIMARY KEY  (`site`,`initialDate`,`finalDate`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='Mestrado - Histórico de Avaliações realizadas';

-- --------------------------------------------------------

--
-- Estrutura da tabela `evaluationsTask`
--

CREATE TABLE IF NOT EXISTS `evaluationsTask` (
  `id` timestamp NOT NULL default CURRENT_TIMESTAMP on update CURRENT_TIMESTAMP,
  `site` varchar(50) NOT NULL,
  `task` varchar(100) NOT NULL,
  `ips` text NOT NULL,
  `hitRate` int(10) NOT NULL,
  `noTaskRate` int(10) NOT NULL,
  `taskCompleteRate` int(10) NOT NULL,
```

```
  `taskIncompleteRate` int(10) NOT NULL,
  `agentChangeRate` int(10) NOT NULL,
  `longTimeRate` int(10) NOT NULL,
  `initialDate` date NOT NULL,
  `finalDate` date NOT NULL,
  `elapsedTime` int(10) NOT NULL,
  PRIMARY KEY  (`site`,`initialDate`,`finalDate`,`task`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='Mestrado - Histórico de Avaliações realizadas';

-- -------------------------------------------------------

--
-- Estrutura da tabela `sites`
--

CREATE TABLE IF NOT EXISTS `sites` (
  `user` varchar(50) NOT NULL,
  `site` varchar(50) NOT NULL,
  `logFile` varchar(100) NOT NULL,
  `longTime` int(10) NOT NULL COMMENT 'em segundos',
  `URLs` text NOT NULL,
  `helpURLs` text NOT NULL,
  `errorURLs` text NOT NULL,
  PRIMARY KEY  (`user`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='Mestrado - Sites';

-- -------------------------------------------------------

--
-- Estrutura da tabela `tasks`
--

CREATE TABLE IF NOT EXISTS `tasks` (
  `site` varchar(50) NOT NULL,
  `task` varchar(100) NOT NULL,
  `steps` text NOT NULL,
  PRIMARY KEY  (`site`,`task`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='Mestrado - Tarefas dos Sites';

-- -------------------------------------------------------

--
-- Estrutura da tabela `usuarios`
--

CREATE TABLE IF NOT EXISTS `usuarios` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `nome` varchar(100) NOT NULL,
  `usuario` varchar(50) NOT NULL,
  `senha` varchar(50) NOT NULL,
  PRIMARY KEY  (`id`),
  UNIQUE KEY `usuario` (`usuario`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=3 ;
```