Revista Brasileira de Administração Científica,
Aquidabã, v.3, n.2, Ago 2012.

ISSN ***2179-684X***

SEÇÃO: Artigos
TEMA: ***Sistemas de Informação***

Anais do Simpósio Brasileiro de Tecnologia da Informação (SBTI 2012)

DOI: 10.6008/ESS2179-684X.2012.002.0013

Michele de Vasconcelos Leitão
Universidade de Pernambuco, Brasil
http://lattes.cnpq.br/5393004794033161
micheledevasconcelos@gmail.com

Darliane Goes de Miranda
Universidade de Pernambuco, Brasil
http://lattes.cnpq.br/2243606127474297
darlygoes@gmail.com

Denis Silva da Silveira
Universidade Federal de Pernambuco, Brasil
http://lattes.cnpq.br/3799501798859187
dsilveira@ufpe.br

Maria Lencastre Pinheiro de Menezes Cruz
Universidade de Pernambuco, Brasil
http://lattes.cnpq.br/9016360553400035
mlpm@ecomp.poli.br

Recebido: 10/06/2012
Aprovado: 15/08/2012
Avaliado anonimamente em processo de pares cegas.

Referenciar assim:

LEITÃO, M. V.; MIRANDA, D. G.; SILVEIRA, D. S.; CRUZ, M. L. P. M.. Uma análise de usabilidade na técnica da análise de pontos de função. Revista Brasileira de Administração Científica, Aquidabã, v.3, n.2, p.197-213, 2012.

UMA ANÁLISE DE USABILIDADE NA TÉCNICA DA ANÁLISE DE PONTOS DE FUNÇÃO

RESUMO

O acesso à informação de qualidade é essencial para a eficácia de organizações. Essa demanda torna a interface com o usuário parte fundamental dos Sistemas de Informação, por constituir o meio pelo qual usuários se comunicam com aplicativos para ter acesso às informações. Mensurar o esforço de desenvolvimento de uma interface de qualidade – que satisfaça a muitos usuários – não é uma tarefa simples, tendo em vista a diversidades das pessoas e suas distintas necessidades. O esforço contempla não somente o tempo e o custo exigido para o desenvolvimento, mas também a quantidade de pessoas e horas necessárias para a construção. As métricas de software são consideradas as técnicas mais precisas para garantir qualidade em aplicações de software. Entre elas, a técnica de Análise de Pontos de Função é a mais difundida, devido à sua flexibilidade de plataforma e linguagem de desenvolvimento, e a mais utilizada para estimativa de custo nos contratos de sistemas governamentais. Entretanto, a definição de usabilidade nessa técnica não é aderente aos padrões utilizados para averiguar usabilidade, como descrito na literatura. Nesse contexto, este trabalho tem como propósito analisar a descrição da usabilidade presente nessa métrica, visando identificar sua adequação à taxonomia da usabilidade proposta por Ferreira e Nunes e às heurísticas de Jakob Nielsen.

PALAVRAS-CHAVES: Usabilidade; Análise de Ponto de Função; Taxonomia; Heurísticas de Nielsen.

AN ANALYSIS OF USABILITY IN POINT FUNCTION ANALYSIS

ABSTRACT

Access to quality information is essential for the effectiveness of organizations. This demand makes the user interface a fundamental part of information systems, as it is the means by which users communicate with applications to access information. Measuring the effort of developing a quality interface - that satisfies many users - is not a simple task, because of the diversity of people and their different needs. The effort includes not only the time and cost required for development, but also the amount of people and hours required for construction. The software metrics are considered the most accurate techniques for ensuring quality in software applications. Among them, the technique of Function Point Analysis is the most widespread, due to its flexibility and platform development language, and the most used to estimate the cost of government systems. However, the definition of usability presented at this technique is not adhering to the standards used to determine usability, as described in the literature. In this context, this paper aims to analyze the usability of this description in that metric, to identify their suitability for the taxonomy proposed by Ferreira and Nunes and the heuristics of Jakob Nielsen.

KEYWORDS: Usability; Function Point Analysis; Taxonomy; Nielsen's Heuristics.

INTRODUÇÃO

A crescente demanda por *software* e a proliferação de aplicações para gerenciamento de negócios, produtos e relacionamentos com clientes criou a necessidade de maior confiabilidade e qualidade no *software*. Como consequência, passou também a existir carência de engenheiros especializados e métodos apropriados para garantir esses requisitos (CHEESMAN, 2001).

Nesse sentido, a Engenharia de Requisitos - subdivisão da Engenharia de *Software* - atua como um processo para garantir que as especificações de um sistema atendam adequadamente às necessidades e expectativas dos clientes, oferecendo um mecanismo de análise de necessidades de funcionamento do sistema, avaliação de ambiguidade das informações e administração das requisições.

De acordo com Ferreira e Nunes (2008), deve-se elaborar o projeto de qualquer produto tendo em vista a expectativa dos usuários e a facilidade do seu uso. Porém, é na etapa de definição de requisitos que características do sistema - de aspectos funcionais e não funcionais - devem ser consideradas. As características funcionais descrevem as funções necessárias para cumprir os objetivos do sistema, mas é entre as características não funcionais que estão os aspectos relacionados à qualidade do *software* e à experiência do usuário.

Requisitos não funcionais definem as qualidades e atributos de um sistema, além das restrições sobre os serviços ou as funções oferecidas pelo sistema. Devido ao fato dos requisitos não funcionais serem restrições sobre os serviços do sistema, eles são frequentemente de importância crítica, e alguns requisitos funcionais podem ter que ser sacrificados para que outras restrições não funcionais possam ser alcançadas. Requisitos não funcionais incluem segurança de dados, segurança de acesso, usabilidade, confiabilidade, desempenho, entre outros (SOMMERVILLE, 2007).

Conforme Hix e Hartson (1993), as avaliações da qualidade de interfaces com usuários são de natureza subjetiva e giram em torno de sua usabilidade. Isto é, as avaliações estão associadas às condições oferecidas aos usuários na utilização da funcionalidade da aplicação (ROMANI e BARANAUSKAS, 1998).

Um problema comum com os requisitos não funcionais é que eles são, muitas vezes, de difícil verificação; eles podem refletir os objetivos gerais do cliente, como a facilidade de uso, a habilidade do sistema se recuperar de uma falha ou a rapidez de resposta ao usuário, aspectos de difícil medição.

Existem diversas medidas de garantia de qualidade fundamentais para o sucesso de qualquer tipo de aplicação de *software*. Dentre elas, uma das mais simples e menos custosa é a medição de *software*. Através de métricas de *software* é possível avaliar a eficácia dos métodos de programação e a confiabilidade de um sistema; pode-se também determinar o esforço ou tempo para realização de uma tarefa ou o tamanho do produto, por exemplo. Além disso, as métricas de *software* podem ser calculadas, entendidas e testadas e independem do observador

que as aplica, sendo também uma boa fonte para estudos estatísticos acerca do ciclo de vida do *software* (ABREU *et al*., 2010).

Entre as métricas de *software* desenvolvidas, a Análise de Pontos de Função (APF) – criada em 1979 por Allan Albrecht, da IBM – tem sido a mais utilizada e referenciada em termos contagem de requisitos funcionais. Trata-se de uma medida de tamanho funcional de projetos de *software* que considera as funcionalidades implementadas, independente da metodologia e tecnologia utilizadas, sob o ponto de vista do usuário. A APF, no entanto, não garante a contagem de usabilidade, visto que a descrição da usabilidade presente nessa métrica, representada pela Característica Geral do Sistema (CGS) "Eficiência do Usuário Final", não se enquadra no conceito de usabilidade presente na literatura.

Neste contexto, este trabalho tem como objetivo avaliar os 16 itens da CGS "Eficiência do Usuário Final", a fim de identificar se esses itens compõem usabilidade como descrita na taxonomia da usabilidade proposta por Ferreira e Nunes (2008) e nas heurísticas de Jakob Nielsen (NIELSEN e MOLICH, 1990).

Este trabalho é organizado da seguinte forma: na seção 1 é feita uma explanação sobre os conceitos e benefícios da usabilidade em sistemas de informação, como descrito na literatura; são também apresentadas as métricas mais usadas na medição de usabilidade e, por isso, escolhidas para este estudo. Além disso, é apresentada a métrica APF e a forma como a usabilidade é medida nessa abordagem, bem como o seu uso em projetos governamentais. Na seção 2 é discutida a metodologia usada na avaliação dos 16 itens da CGS de usabilidade e alguns resultados preliminares. Na seção 3 são apresentados os resultados a avaliação da CGS. Por fim, na seção 4 são apresentadas as conclusões e propostas para trabalhos futuros.

REVISÃO TEÓRICA

Conceitos de Usabilidade

A qualidade do uso de Sistemas de Informação, amplamente chamada de interação do homem com o computador, vem sendo motivo de muitos estudos. Neste contexto, a usabilidade estuda a interação via interface, ou seja, a maneira como um usuário realiza suas tarefas e interage com um determinado produto, considerando as diferentes necessidades e tipos de usuários. Dix *et al*. (1993) afirmam que o objetivo primeiro de um sistema interativo é permitir ao usuário alcançar metas particulares em algum domínio de aplicação; assim, o sistema interativo deve ser "*usável*".

Segundo Nielsen (1993), usabilidade é um conceito que busca definir as características de utilização, do desempenho e da satisfação dos usuários na interação com as interfaces computacionais, na perspectiva de um bom sistema interativo, e se refere a cinco componentes:

1. **Aprendizado:** O sistema deve ser fácil de aprender. Os utilizadores inexperientes devem ser capazes de completar tarefas básicas num curto período de tempo, com um mínimo de formação.
2. **Eficiência:** Os usuários experientes devem ser capazes de alcançar um estado constante de produtividade.
3. **Memorável:** O sistema deve ser fácil de lembrar. Os usuários podem retornar a ele depois de uma ausência e completar tarefas sem re-treino.
4. **Baixo número de erros:** Usuários devem enfrentar poucos erros durante o uso do sistema, e recuperarem-se rapidamente de erros.
5. **Satisfação dos usuários:** O sistema deve ser agradável de usar.

Métricas de Software

Medir a usabilidade de uma interface envolve não apenas medir questões relativas às funcionalidades de um *software*, mas também a facilidade de seu uso como ferramenta de trabalho, tendo como um dos principais desafios a redução do tempo necessário para aprender-se a utilizar o sistema (NIELSEN, 1993). Nessa sessão serão descritas as métricas mais utilizadas na medição de usabilidades e, portanto, escolhidas para este trabalho.

Taxonomia

Os requisitos não funcionais de usabilidade são descritos em (FERREIRA e NUNES, 2008) como aqueles “desejáveis em uma boa interface”, e podem ser agrupados em duas categorias, de acordo com Pressman (PRESSMAN, 2004): (i) Requisitos associados à exibição de informação e (ii) Requisitos relacionados à entrada de dados.

(i) Requisitos associados à exibição de informação

É possível exibir informações por meio de diversos formatos, como textos, sons e imagens, e através de combinações de cores e formas; entretanto, esta informação deve ser apresentada na sua completude, sem ambiguidade e inteligível (FERREIRA e NUNES, 2008). Portanto, quando se visa construir interfaces eficientes no que se refere à informação exibida, é necessário obedecer a alguns requisitos (PRESSMAN, 2004), descritos a seguir:

- Consistência: Trata-se de uma das principais características de usabilidade da interface (NIELSEN, 2000). Permite que o usuário generalize o conhecimento acerca de um aspecto do sistema para outros (FOLEY, 1997), além de reduzir a insatisfação provocada por comportamentos inesperados e logicamente incompreensíveis do sistema (FERREIRA e NUNES, 2008). Visando esse comportamento homogêneo e passível de generalização, a interface consistente é composta de *menus*, comandos de entrada e funções com apresentação visual e comportamentos idênticos, representados, por exemplo, pela padronização do emprego de maiúsculas em comandos globais como “Salvar”, “Sair” e “Ajuda” (FERREIRA e NUNES, 2008).

- *Feedback*: É necessário em todo tipo de comunicação (FERREIRA e NUNES, 2008). Logo, na interação entre o usuário e sistema, caracterizada como um meio de comunicação, é requerido *feedback*, e este precisa ser planejado e programado. De acordo com (FOLEY, 1997), o *feedback* pode ocorrer em três níveis: Nível de *Hardware*, que é gerado sempre que o usuário manipula um dispositivo de entrada (ex.: resposta de movimentação de mouse, ou visualização simultânea de caracteres digitados); Nível de Sequência, que indica ao usuário que um comando executado foi aceito (ex.: objetos selecionados devem ser realçados); e Nível Funcional, que indica o tempo de processamento de um comando executado pelo usuário (ex.: a ampulheta mostrando que a CPU está ocupada, ou uma barra de progresso percentual da execução do comando). Vale enfatizar que o tempo de espera é um fator crítico de usabilidade na *web*, que representa, atualmente, uma informação difícil de ser calculada e de baixa confiabilidade.

- Níveis de Habilidade e Comportamento Humano: De acordo com Ferreira e Nunes (2008), o processo de produção de um *software* depende profundamente do fator social. É fundamental levar em conta o tipo de formação e o meio social do conjunto de usuários do sistema (FERREIRA e NUNES, 2008) na criação de produtos efetivamente úteis para determinados grupos de pessoas. Esses grupos devem ser avaliados em relação às tarefas que realizam, aos seus conceitos sobre o ambiente de trabalho e aos tipos de imposições e limitação às quais estão sujeitas (APPLE, 1992).

- Percepção Humana: A memória humana compõe de duas partes: memória de curta duração (*Short-term memory* – STM), que armazena os inputs sensoriais (visuais, auditivos e táteis); e memória de longa duração (*Long-term memory* – LTM), que armazena conhecimento. Para auxiliar a memória, a maioria das pessoas usa uma série de heurísticas para resolver um problema associando com situações do mundo real. A interface de um bom sistema não exige do usuário uso excessivo de nenhuma dessas memórias, muito menos das duas simultaneamente; e permite que o usuário interaja com ela por meio de heurísticas coerentes (FERREIRA e NUNES, 2008).

- Metáforas: Consiste na técnica de substituir um signo por outro para tornar a comunicação mais efetiva (MARCUS, 1998). O uso de metáforas aproxima o usuário do mundo virtual fazendo-se analogias ao mundo real. Um exemplo amplamente utilizado em sistemas *web* comerciais é o carrinho de compras, que representa a cesta de compras do usuário à medida que ele vai escolhendo itens para comprar no sistema (FERREIRA e NUNES, 2008).

- Minimização da Carga de Memória: No *design* de interfaces, o uso de comandos mnemônicos e a permissão de ícones reduz o esforço exigido do usuário para memorizar suas características (FOLEY, 1997). É preciso cuidado na geração dos signos, de modo que não gerem dúvidas e expressem com precisão seus objetivos, evitando-se nomes longos de comandos, por exemplo (FERREIRA e NUNES, 2008).

- Eficiência no Diálogo, Movimento e Pensamentos: Deve ser considerado o posicionamento entre ícones num projeto de *layout* da tela, de forma a encurtar a distância que o mouse percorre entre dois cliques (FERREIRA e NUNES, 2008); principalmente entre ações que são precedentes, como “copiar” e “colar” trechos de textos e/ou imagens, por exemplo.

- Classificação Funcional dos Comandos: Um estudo do psicólogo George A. Miller revelou que as pessoas, ao lidar com determinada quantidade de informações (ou, no contexto de *menus*, com certa quantidade de itens),

sentem mais facilidade quando o número de elementos não excede a sete mais ou menos dois; ou seja, segundo essa teoria os menus devem conter no máximo nove itens (FERREIRA e NUNES, 2008). Esse estudo aponta uma das características funcionais de menus, mas a presença de indicações claras a respeito dos comandos relativos a cada *menu* também tem grande impacto na interação do usuário (ex.: o *menu* Editar deve possuir itens relativos à tarefa de edição).

- Manipulação Direta: Trata-se de permitir ao usuário manter controle dos objetos representados no computador, sendo estes visíveis ao mesmo durante qualquer operação feita por ele, bem como o efeito dessa operação se tornar imediatamente visível (APPLE, 1992).

- Exibição Exclusiva de Informação Relevante: Deve-se mostrar exclusivamente a informação que seja relevante ao contexto corrente, a fim de facilitar sua assimilação. Isto poupa ao usuário o trabalho de explorar vários menus, textos e imagens até encontrar o que precisa para executar sua tarefa (PRESSMAN, 2004).

- Uso de Rótulos, Abreviações e Mensagens Claras: De acordo com o Guia de Interface da Apple (APPLE, 1992), deve-se ter atenção à consistência dos rótulos, padronização de abreviaturas e previsibilidade das cores; os símbolos previamente padronizados não devem ser alterados, evitando a confusão de significados. Da mesma forma, as mensagens geradas devem ser claras e elucidativas; o projetista deve evitar que o usuário sinta-se culpado pelo erro, por exemplo (FERREIRA, 2008).

- Uso Adequado de Janelas: No trabalho (NIELSEN e LORANGER, 2006) os autores recomendam evitar a abertura de novas janelas ao clicar em um *link*, devido à poluição visual e à desativação o botão “Voltar”, visto que novas janelas não herdam o histórico da janela original. Entretanto, exceções são consideradas para os casos de *hyperlinks* (*link* para conteúdos externos) em *sites* - para conteúdos *web* e não *web*, como arquivos PDF e Word -, visto que o usuário pode perder o foco da leitura. Outro tipo de janela que deve ser evitada é o *pop-up*, pois este causa irritação no usuário e distorce a expectativa do usuário sobre o conteúdo *web*, que é a informação exibida na janela principal (NIELSEN e LORANGER, 2006).

- Projeto Independente da Resolução do Monitor: A questão de resolução de monitor interfere na usabilidade dos *sites*, visto que apresentam características dinâmicas, podendo ser acessados por meio de diversas plataformas e dispositivos. Um dos princípios básicos para o bom planejamento na construção de *sites*, independentes de resolução de monitores, consiste em definir a aparência dos componentes por meio de percentuais do espaço disponível, determinado pelo número de *pixels* (NIELSEN, 2000).

(ii) Requisitos relacionados à entrada de dados

A escolha de comandos, digitação de dados e outros *inputs* consomem muito tempo de trabalho; em (PRESSMAN, 2004) o autor recomenda alguns procedimentos para otimizar esse tempo e aprimorar as interfaces quanto à entrada de dados (FERREIRA, 2008). São estes:

- Mecanismos de Ajuda: De acordo com (FOLEY, 1997), devem ser fornecidas informações de ajuda para toda ação de entrada e sempre que o mouse “passar” sobre um item de interface; esse mecanismo auxilia o usuário a sempre ter a informação correta sobre as entradas que deve fornecer ao sistema.

- Prevenção de Erros: Um bom projeto de interface não permite que o usuário escolha uma opção inválida e, só após a ação, receba um *feedback* de erro. Assim, alguns requisitos devem ser seguidos: (i) desativação ou inibição de itens inválidos, (ii) exibição de orientações adequadas para a entrada correta de dados, (iii) minimização da quantidade de dados de entrada, (iv) permissão ao usuário de controlar o fluxo interativo (ex.: "pular" e alterar a ordem de ações) e (v) permissão ao usuário de personalizar comandos e mensagens.

- Tratamento de Erros: A ocorrência de erros é inevitável, mas a boa interface proporciona aos usuários meios de corrigir os erros rapidamente (FERREIRA, 2008). Existem dois tipos de erros: o sintático, que decorre do fornecimento de parâmetros/nomes errados para um comando/sequência de comandos; e o funcional, mais grave, que ocorre quando um comando é acionado por engano (FERREIRA, 2008). Para o erro sintático, a interface deve exibir uma mensagem clara, que mostre a causa do erro e dê ao usuário a opção de desfazê-lo. Quanto ao erro funcional, existem quatro comandos para tratá-lo: (i) *undo*, que reverte o efeito do comando que gerou o erro; (ii) *abort*, que cancela o comando durante a execução; (iii) *cancel*, que cancela o comando antes de ser acionado; e (iv) *correct*, que corrige o comando antes da execução (FERREIRA, 2008).

Heurísticas de Nielsen

O método de avaliação heurística foi proposto por Nielsen e Molich (1990) em resposta à necessidade de efetivamente se "depurar" o *design* de sistemas interativos (ROMANI e BARANAUSKAS, 1998). É uma das técnicas mais utilizadas porque consegue obter resultados razoáveis a baixo custo. Uma de suas heurísticas, chamada "*Match between system and real world*", funciona com uma avaliação de cada página do sistema e a mesma deverá usar palavras, frases e conceitos familiares ao usuário ao invés de termos técnicos (BLANDFORD *et al*., 2004).

As heurísticas mais comumente utilizadas são as 10 (dez) heurísticas de Nielsen, as 8 (oito) regras de ouro de *design* de interface de Shneiderman e os 7 (sete) princípios de Norman (NIELSEN, 1993; SHNEIDERMAN, 1998; NORMAN, 1998). As dez heurísticas desenvolvidas por Jakob Nielsen e Rolf Molich são as heurísticas de usabilidade mais populares e comumente utilizadas, e são derivadas de uma revisão de heurísticas baseadas em uma análise fatorial de 249 problemas de usabilidade (NIELSEN, 1994). São estas:

1. Visibilidade do status do sistema: o usuário deve ser informado pelo sistema em tempo razoável sobre o que está acontecendo.
2. Compatibilidade do sistema com o mundo real: o modelo lógico do sistema deve ser compatível com o modelo lógico do usuário.
3. Controle e liberdade do usuário: o sistema deve tornar disponíveis funções que possibilitem saída para funções indesejadas. O usuário controla o sistema, ele pode, a qualquer momento, abortar uma tarefa, ou desfazer uma operação e retornar ao estado anterior.
4. Consistência e padrões: o sistema deve ser consistente quanto à utilização de sua simbologia e à sua plataforma de *hardware* e *software*. A mesma operação deve ser apresentada na mesma localização e deve ser formatada/apresentada da mesma maneira para facilitar o reconhecimento.
5. Prevenção de erros: o sistema deve ter um *design* que se preocupe com as possibilidades de erro.

6. Reconhecimento ao invés de lembrança: as instruções para o bom funcionamento do sistema devem estar visíveis no contexto em que o usuário se encontra.
7. Flexibilidade e eficiência de utilização: o sistema deve prever o nível de proficiência do usuário em relação ao próprio sistema, dispondo de abreviações e recursos como atalhos, teclas de função, duplo clique no *mouse*, função de volta em sistemas hipertexto.
8. Estética e design minimalista: os diálogos do sistema devem conter somente informações relevantes ao funcionamento.
9. Ajuda aos usuários no reconhecimento, diagnóstico e correção de erros: as mensagens devem ser expressas em linguagem clara, indicando as possíveis soluções.
10. Ajuda e documentação: a informação desejada deve ser facilmente encontrada, de preferência deve ser contextualizada e não muito extensa.

Análise de Pontos de Função (APF)

A área de métricas de *software* tornou-se bastante atrativa para os gestores de Tecnologia da Informação e negócios e para instituições por oferecerem uma alternativa para o controle de recursos, de aceitação de entregáveis, de processo de desenvolvimento de *software* e qualidade de produto de *software*.

A APF tem sido o alvo de uso principalmente devido ao seu sucesso e reconhecimento em várias esferas organizacionais, justificado pela eficiência na medição de requisitos funcionais de sistemas.

Foi criada por Allan Albrecht visando minimizar as dificuldades associadas ao uso da métrica Linhas de Código (*Lines of Code* - LOC) como unidade de medida de tamanho de *software* e suportar a previsão do esforço de desenvolvimento do projeto de *software* (ALBRECHT, 1983). Trata-se de uma medida de tamanho funcional de projetos de *software* que considera as funcionalidades implementadas, independente da metodologia e tecnologia utilizadas, sob o ponto de vista do usuário. A APF é uma medida padronizada e mantida pelo Grupo Internacional de Usuários de Pontos de Função, o *International Function Point Users Group* (IFPUG).

Em 1986, em pesquisa do *Quality Assurance Institute* a APF foi considerada a melhor métrica para o estabelecimento de medições de qualidade e produtividade de projetos de *software* (PERRY, 1986). De acordo com Jones (1994), a APF se tornou a métrica mais utilizada e estudada na Engenharia de *Software* em 1993. Atualmente, essa métrica continua sendo a mais utilizada na indústria de *software*, como métrica padrão na definição de indicadores, como insumo para derivação de estimativas de prazo, custo e esforço e no estabelecimento de contratos de *software* (HAZAN, 2010).

Características Gerais do Sistema (CGS)

O processo de contagem de pontos de função envolve sete etapas. As duas primeiras fases abordam os tipos de contagem e o escopo da aplicação que será contada. As quatro etapas seguintes referem-se à contagem de requisitos funcionais e não funcionais. A última fase refere-se ao cálculo final em pontos de função da aplicação que está sendo mensurada (REINALDO, 2009). A Figura 2 apresenta a visão geral do processo de contagem de pontos de função.

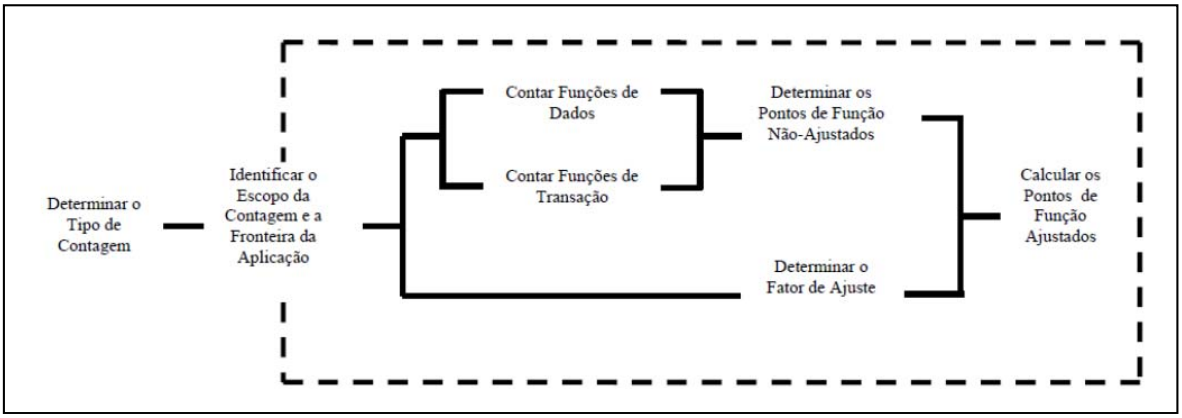

Figura 2: Visão geral do processo de contagem de pontos de função.
Fonte: Manual de Práticas de Contagem de Pontos de Função, 2005.

Dentre as quatro etapas destinadas à contagem de requisitos funcionais e não funcionais, somente a última é de fato relativa à contagem de alguns requisitos não funcionais. Essa contagem determina o Valor do Fator de Ajuste (VAF), variável de impacto no ajuste dos pontos de função (os pontos calculados através dos requisitos não funcionais são chamados de Pontos Não Ajustados) e, por consequência, na contagem total de pontos da aplicação.

O Valor do Fator de Ajuste é baseado em 14 Características Gerais de Sistema (CGS), como comunicação de dados, processamento, volume de transações, facilidade de mudanças, entre outras. Essas características classificam as funcionalidades gerais que influenciam na complexidade do *software* e no tamanho da aplicação que está sendo contada.

Cada característica tem descrições associadas, visando determinar seu nível de influência no sistema, que é medido através da atribuição de um peso para cada nível de influência (NI), distribuído numa escala de 0 a 5, sendo 0 de nenhuma influência e 5 de forte influência no sistema. O Quadro 1 mostra a lista de CGS como definidas no Manual de Práticas de Contagem de Pontos de Função, versão 4.2.1, 2005.

Quadro 1: Características Gerais do Sistema.

Características Gerais do Sistema (CGS)	
1. Comunicação de Dados	8. Atualização *On-Line*
2. Processamento Distribuído	9. Processamento Complexo
3. Performance	10. Reusabilidade

4. Configuração Intensamente Utilizada	11. Facilidade de Instalação
5. Volume de Transações	12. Facilidade de Operação
6. Entrada de Dados *On-Line*	13. Múltiplos Locais
7. Eficiência do Usuário Final	14. Facilidade de Mudança

Fonte: Manual de Práticas de Contagem de Pontos de Função, 2005.

Existem muitas críticas às CGS, em (LOKAN e ABRAN, 1999) os autores citam que estes fatores foram baseados na intuição dos elaboradores da métrica, assim não possuem o benefício de nenhuma teoria e nem se baseiam em algum fundamento estatístico. Em (FENTON e PFLEEGER, 1997) os autores identificam problemas com a subjetividade das CGS, que podem diminuir ou incrementar em até 35% (valor muito significante) o tamanho do *software*, além de apontar problemas de dupla contagem com relação às características gerais. Já em (KITCHENHAM, 1992) a autora encontrou variações comuns nas características: em seus estudos somente cinco características do total de 14 seriam suficientes (CALAZANS *et al*., 2005). Além disso, é possível observar que essa métrica não contempla requisitos não funcionais, em especial usabilidade.

CGS de Usabilidade: Eficiência do Usuário Final

Essa Característica Geral do Sistema descreve o nível segundo o qual foram considerados os fatores humanos e a facilidade de uso para o usuário na aplicação medida. É composta por 16 itens elaborados visando contemplar o máximo de aspectos relacionados com a eficiência do aplicativo na interação com o usuário. A pontuação do sistema em relação aos itens é decidida quanto aos recursos implementados para tornar o aplicativo "amigável". No Quadro 2 são listados os 16 itens da CGS Eficiência do Usuário Final.

Quadro 2: Relação dos 16 itens da Característica Geral do Sistema - Eficiência do Usuário Final.

ITENS DA CARACTERÍSTICA GERAL DO SISTEMA - EFICIÊNCIA DO USUÁRIO FINAL	
1. Auxílio à navegação (ex.: teclas de função, *menus* dinâmicos, *hyperlinks*)	9. *Combo-boxes* (caixas de combinação)
2. *Menus*	10. Uso intenso de vídeo reverso, brilho, cores, sublinhado e outros indicadores
3. Ajuda *on-line* e documentação	11. Documentação impressa das transações *on-line* (ex.: *print-screen*)
4. Movimentação automática do cursor	12. Interface de mouse
5. Paginação	13. Janelas *pop-up*
6. Impressão remota (através de transações *on-line*)	14. *Templates* e/ou *defaults*
7. Teclas de função pré-definidas (ex.: limpeza de tela, solicitação de ajuda)	15. Suporte a 2 idiomas (pontua como 4 itens)
8. Tarefas *batch* executadas a partir de transações *on-line*	16. Suporte a mais de 2 idiomas (pontua como 6 itens)

Fonte: Manual de Práticas de Contagem de Pontos de Função, 2005.

A pontuação é atribuída de acordo com a quantidade de itens presentes no sistema, de acordo com a escala representada no Quadro 3.

Quadro 3: Escala de pontuação da Característica Geral do Sistema – "Eficiência do Usuário Final".

Pontos	Número de Itens
0	0
2	4 a 5 itens
3	6 ou mais itens. Não existem requisitos específicos do usuário relacionados à eficiência.
4	6 ou mais itens. Os requisitos definidos quanto a eficiência são suficientemente fortes para requerer o projeto de tarefas que incluam fatores humanos.
5	6 ou mais itens. Os requisitos definidos quanto à eficiência são suficientemente fortes para requerer o uso de processos e ferramentas especiais para validar os objetivos alcançados.

Fonte: Manual de Práticas de Contagem de Pontos de Função, 2005.

APF em Projetos Governamentais

Visando a melhoria no gerenciamento e qualidade dos projetos de TI desenvolvidos por órgãos públicos brasileiros, foi criada a Portaria SLTI/MP Nº 31, de 29 de novembro de 2010, que recomenda que os órgãos integrantes do Sistema de Administração dos Recursos de Informação e Informática (SISP) adotem a Análise de Ponto de Função nas suas contratações de serviços de desenvolvimento e manutenção de soluções de *software*.

A adoção da métrica Análise de Pontos de Função, no entanto, não garante a contagem de usabilidade, visto que a descrição desta característica através da CGS de usabilidade "Eficiência do Usuário Final" não se enquadra no conceito de usabilidade.

METODOLOGIA

A fim de avaliar a qualidade da interação proporcionada pela usabilidade foram propostas várias métricas e técnicas, que podem ser aplicadas em diferentes etapas de desenvolvimento do sistema e envolver usuários ou avaliadores. Esses métodos são classificados em analíticos e empíricos (ISO 9126, 1991):

- Os métodos empíricos envolvem a participação de usuários para a coleta de dados, que são posteriormente analisados pelo especialista para identificar os problemas da interface. O uso desse tipo de método requer a implementação real do sistema, pelo menos em um formato que simule a capacidade interativa do sistema.
- Os métodos analíticos, também conhecidos como métodos de inspeção, ou de prognóstico, caracterizam-se pelo fato de o usuário não participar diretamente das avaliações. A avaliação analítica é usada geralmente para avaliar o design das interfaces, baseando-se no julgamento dos avaliadores. Além da identificação de potenciais erros, os avaliadores procuram fazer sugestões de acertos e assim contribuir com a melhoria da usabilidade do produto (ISO 9126, 1991). Esses métodos podem ser aplicados em qualquer fase de desenvolvimento do sistema, tendo como resultado um relatório formal dos problemas identificados e sugestões de melhorias.

Neste trabalho foi considerada a classificação de métodos analíticos na avaliação dos 16 itens da CGS de usabilidade da métrica de Análise de Pontos de Função. A análise da CGS de

usabilidade, visando identificar critérios de usabilidade como descritos na taxonomia de usabilidade e nas heurísticas de Nielsen, foi executada em 2 (duas) etapas.

Na primeira etapa foi elaborado um mapeamento entre essas métricas, que tinha por objetivo agrupar os critérios comuns e gerar uma nova relação de critérios, produto do relacionamento entre esses os critérios mapeados. As 10 heurísticas de Nielsen e os 16 critérios da taxonomia de usabilidade foram rotulados neste trabalho de N1 a N10 e de T1 a T16, respectivamente, de acordo com a ordenação da listagem apresentada na seção anterior. Todos os critérios, de ambas as métricas, foram comparados de acordo com a descrição de seus autores quanto à aplicação para a avaliação de usabilidade de sistemas, visando a junção mais completa desses critérios. Os Quadros 4 e 5 mostram as listagens dos critérios de cada métrica.

Quadro 4: Relação das heurísticas de Nielsen rotuladas.

Heurísticas de Nielsen rotuladas	
N1	Visibilidade do status do sistema
N2	Compatibilidade do sistema com o mundo real
N3	Controle e liberdade do usuário
N4	Consistência e padrões
N5	Prevenção de erros
N6	Reconhecimento ao invés de lembrança
N7	Flexibilidade e eficiência de utilização
N8	Estética e *design* minimalista
N9	Ajuda aos usuários no reconhecimento, diagnóstico e correção de erros
N10	Ajuda e documentação

Quadro 5: Relação dos critérios da taxonomia de usabilidade rotulados.

Taxonomia de usabilidade rotulada	
T1	Consistência
T2	*Feedback*
T3	Níveis de Habilidade e Comportamento Humano
T4	Percepção Humana
T5	Metáforas
T6	Minimização de Carga de Memória
T7	Eficiência no Diálogo, Movimento e Pensamentos
T8	Classificação Funcional dos Comandos
T9	Manipulação Direta
T10	Exibição Exclusiva de Informação Relevante
T11	Uso de Rótulos, Abreviações e Mensagens Claros
T12	Uso Adequado de Janelas
T13	Projeto Independente da Resolução do Monitor
T14	Mecanismos de Ajuda
T15	Prevenção de Erros
T16	Tratamento de Erros

A segunda etapa consistiu na análise dos 16 itens descritos na CGS de usabilidade da APF, intitulada “Eficiência do Usuário Final”, através da verificação desses itens com base na listagem dos novos critérios gerados pela primeira etapa. Essa verificação visou identificar se algum desses itens não estava contemplado nos novos critérios e/ou se algum dos novos critérios não possuía correspondência nos itens da CGS, e assim serem detectados os critérios a serem adaptados na CGS para permitir a adequação dessa métrica a usabilidade como descrita na taxonomia de usabilidade e nas heurísticas de Nielsen.

RESULTADOS E DISCUSSÕES

Através do mapeamento da primeira etapa foi possível observar que 2 (dois) critérios da taxonomia de usabilidade não possuíam correspondência com nenhum critério de Nielsen. O quadro da Figura 2 mostra o relacionamento entre os critérios, com destaque para esses dois critérios, denominados neste trabalho como T3 e T8 (e formalmente denominados "Níveis de Habilidade e Comportamento Humano" e "Classificação Funcional dos Comandos", respectivamente). Esse resultado representou uma agregação às heurísticas de Nielsen e consequente criação de 12 novos critérios, produto da associação dos critérios comuns das métricas mapeadas.

Quadro 6: Mapeamento entre os critérios das métricas de taxonomia da usabilidade e heurísticas de Nielsen.

Heurísticas de Nielsen x Taxonomia	N1	N2	N3	N4	N5	N6	N7	N8	N9	N10
T1				x						
T2	x									
T3										
T4						x				
T5		X								
T6						x				
T7							x			
T8										
T9	x									
T10								x		
T11				x						
T12				x						
T13				x						
T14										x
T15					x					
T16			x						x	

Os 12 (doze) novos critérios foram compilados numa listagem única – rotulados de NC1 à NC12 – através da junção das descrições dos critérios associados, visando facilitar a execução da segunda etapa da análise da CGS de usabilidade, como mostrado no Quadro 7.

Quadro 7: Relação dos 12 novos critérios gerados na primeira etapa, rotulados.

12 Novos Critérios	
NC1	Visibilidade do *status* do sistema com *feedback* ao usuário
NC2	Compatibilidade do sistema com o mundo real e uso de metáforas
NC3	Controle e liberdade do usuário no tratamento de erros
NC4	Consistência de padrões por meio do uso de rótulos, abreviações e mensagens Claros e uso adequado de janelas
NC5	Prevenção de erros
NC6	Valorização da percepção humana por meio do reconhecimento ao invés de lembrança e minimização de carga de memória do usuário

NC7	Flexibilidade e eficiência no diálogo, movimento e pensamentos
NC8	Estética e *design* minimalista, voltado para exibição exclusiva de informação relevante
NC9	Ajuda aos usuários no reconhecimento, diagnóstico e tratamento de erros
NC10	Mecanismos de ajuda e documentação
NC11	Níveis de habilidade e comportamento humano
NC12	Classificação funcional dos comandos

A partir da compilação dos novos critérios foi feita a verificação dos 16 itens da CGS de usabilidade com base nesses critérios. Essa verificação foi de grande complexidade, devido a não completude da documentação relativa à CGS "Eficiência do Usuário Final", sendo somente fornecidos títulos ou breves descrições de funções que deveriam permitir maior eficiência do usuário, tais como apresentadas no Quadro 2.

Essa escassez de informações impactou numa análise mais interpretativa que puramente comparativa: numa análise inicial dos 16 itens da CGS de usabilidade foi possível observar e inferir que alguns itens poderiam ser agrupados e teriam mais representatividade no mapeamento, como os itens "Janelas *pop-up*" e "*Menus*". Além desses, os itens "*Combos* (caixas de combinação)" e "Interface de *mouse*", por conta da brevidade da descrição, geraram dúvidas em relação à sua categoria e não puderam ser avaliados.

O item "*Menu*", em particular, foi atribuído a dois grupos: NC4 e NC12; este último, relativo à classificação funcional dos comandos, foi atribuído através da consideração da necessidade de um estudo acerca das características e apresentação dos menus, não somente da sua presença no sistema – como deixa a entender esse item na CGS.

Uma análise posterior indicou que os critérios de prevenção e tratamento de erros, de grande importância e impactos na avaliação de usabilidade – visto que foi citado nas duas métricas usadas neste trabalho – não estão sendo contempladas na CGS. No total, 6 (seis) dos 12 novos critérios não possuíam correspondência nos itens da CGS de usabilidade, são eles: NC2, NC3, NC5, NC8, NC9 e NC11, correspondentes à compatibilidade do sistema com o mundo real e uso de metáforas, exibição exclusiva de informação relevante e comportamento humano.

Em contrapartida, foi possível detectar 6 (seis) itens da CGS que também não possuíam equivalentes nos 12 novos critérios, os quais podem ser reagrupados para facilitar a adequação nos critérios. Entre eles, dois são relativos a suporte de idiomas, que pontuam de acordo com a quantidade de idiomas disponíveis no sistema; e um é relativo a tarefas *batch*, que são executadas sem promover a interação do usuário.

O quadro da Figura 3 ilustra o relacionamento entre os 12 novos critérios e os 16 itens da CGS de usabilidade, destacando os critérios e itens sem associação.

Quadro 8: Mapeamento entre os 12 novos critérios e os 16 itens da CGS de usabilidade.

12 Novos Critérios x 16 Itens da CGS de usabilidade	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8	NC9	NC10	NC11	NC12
1							x					

2				X								x
3										x		
4	x											
5												
6												
7				X								
8												
9												
10						x						
11												
12												
13				X								
14				X								
15												
16												

CONCLUSÕES

Como resultado da análise dos itens da Característica Geral do Sistema (CGS) de usabilidade foi possível concluir que a métrica de *software* Análise de Pontos de Função de fato não se enquadra plenamente nos conceitos de usabilidade presentes na literatura, representados neste trabalho pela taxonomia da usabilidade e pelas heurísticas de Jakob Nielsen.

Entretanto, foi observado que a adequação a usabilidade é um procedimento factível e de grande importância, visto que alguns itens da CGS de usabilidade podem ser associados com os critérios escolhidos neste trabalho, e os que ainda não possuem correspondência são de alta criticidade, relativos à prevenção e tratamento de erros do sistema, aspectos estes que podem gerar operações irreversíveis durante o uso do sistema, acarretando em frustração do usuário.

Como trabalho futuro, faz-se necessário um estudo para a adequação dos critérios mapeados e ainda não inseridos na CGS de usabilidade, bem como a adaptação dos 12 novos critérios aos itens sem correspondência da CGS de usabilidade. Esse estudo deve envolver, ainda, uma análise de revisão da escala de pontuação desses itens e à variação de sua quantidade.

REFERÊNCIAS

ABREU, T. C.; MOTA, L. S.; ARAÚJO, M. A. P.. Métricas de software - como utilizá-las no gerenciamento de projetos de software. **Revista Engenharia de Software Magazine**, Juiz de Fora, n.21, p.50-55, 2010.

ALBRECHT, A.; GAFFNEY, J.. Software function, source lines of code, and development effort prediction: a software science validation. **IEEE Transactions on Software Engineering**, Piscataway, v.9, n.6, p.639-648, 1983.

APPLE COMPUTER. **Macintosh human interface guidelines**. Massachusetts: Addison-Wesley, 1992.

BLANDFORD, A.; KEITH, S.; CONNEL, L.; EDWARDS, H.. Analytical usability evaluation for digital libraries: a case study. In: ACM/IEEE-CS JOINT CONFERENCE ON DIGITAL LIBRARIES (JCDL '04). **Anais**. Nova York: ACM, p.27-36, 2004.

CALAZANS, A. T. S; LISBOA, I. C. D.; OLIVEIRA, M. A. L.. Avaliação das características gerais de sistemas na análise por pontos de função - apf por meio da aplicação do gqm – goal, questions, metrics. In: SIMPÓSIO INTERNACIONAL DE MELHORIA DE PROCESSOS DE SOFTWARE (SIMPROS), 7. **Anais**. São Paulo, 2005.

CHEESMAN, J.; DANIELS, J.. **UML components**: A Simple Process for Specifying Component-Based Software. Massachusetts: Addison-Wesley, 2001.

DIX, A.; FINLAY, J.; ABOWD, G.; BEALE, R.. **Human-computer interaction**. Nova York: Prentice-Hall, 1993.

FENTON, N.; PFLEEGER, S.. **Software metrics**: a rigorous and practical approach. Boston: PWS Publishing Company, 1997.

FERREIRA, S. B. L.; NUNES, R. R.. **e-Usabilidade**. Rio de Janeiro: LTC, 2008.

FOLEY, J. D.; DAM, v. A.; FEINER, S.K; HUGHES, J. F.. **Computer graphics**: principles and practice. 2 ed. Massachusetts: Addison-Wesley, 1997.

HARTSON, H.; SHIVAKUMAR, P.; PÉREZ-QUIÑÕNES, A. M.. Usability Inspection of digital libraries: a case study. **International Journal on Digital Libraries (IJDL)**, Nova York, p.108-123, 2004.

HAZAN, C.. Como evitar armadilhas em contratos de fábricas de software. **Revista do Tribunal de Contas da União**, v.42, n.117, p.47-56, 2010.

HIX, D.; HARTSON, H. R.. **Developing user interfaces**: ensuring usability through product and process. Nova York: John Wiley, 1993.

ISO 9126.. **Software product evaluation**: quality characteristics and guidelines for their use. ISO/IEC Standard ISO-9126, 1991.

JONES, C.. Function points: A new way of looking at tools. **Computer**, v.27, n.8, p.66-67, 1994. DOI: http://doi.ieeecomputersociety.org/10.1109/MC.1994.10088

KITCHENHAM, B.A.. Empirical studies of assumptions the underlie software cost- estimation models. **Information and Software Technology**, ScienceDirect , p.211-218, 1992.

LOKAN, C.; ABRAN, A.. Multiple viewpoints in functional size measurement. In: INTERNATIONAL WORKSHOP ON SOFTWARE MEASUREMENT (IWSM'99). **Anais**. Canadá, p.121-132, 1999.

MARCUS, A.. Metaphor design in user interfaces. **ACM SIGDOC Asterisk Journal of Computer Documentation**. Nova York, v.22, n.2, p.43-57, 1998.

NIELSEN, J.. **Usability engineering**. San Diego: Academic Press, 1993.

NIELSEN, J.. Enhancing the explanatory power of usability heuristics. In: ACM CHI'94 CONFERENCE. **Anais**. Boston, p.152-158, 1994.

NIELSEN, J.. **Designing web usability**. Indianapolis: News Riders, 2000.

NIELSEN, J.; LORANGER, H.. **Prioritizing web usability**. Boston: New Riders, 2006.

NIELSEN, J.; MOLICH, R.. Heuristic evaluation of user interfaces. In: EMPOWERING PEOPLE - CHI'90 CONFERENCE. **Anais**. Nova York: ACMPress, 1990.

NORMAN, D.A.. **The Design of Everyday Things**. Massachusetts: MIT Press, 1998.

PERRY, W.E.. **The Best Measures for Measuring Data Processing Quality and Productivity**: A Research Study. Quality Assurance Institute. The Institute: 1986.

PRESSMAN, R. S.. **Software engineering**: a practioner's approach. 6 ed. Nova York: McGraw-Hill, 2004.

REINALDO, W. T.; FILIPAKIS, C. D.. Estimativa de Tamanho de Software Utilizando APF e a Abordagem NESMA. In: ENCONTRO DE ESTUDANTES DE INFORMÁTICA DO ESTADO DO TOCANTINS, 11. **Anais**. Tocantins: CEULP/ULBRA, p.151-160, 2009.

ROMANI, L. A. S.; BARANAUSKAS, M. C. C.. Avaliação Heurística de um sistema altamente dependente do domínio. **Relatório Técnico IC-98-26**, 1998. Disponível: <http://www.ic.unicamp.br/~reltech/1998/98-26.pdf>. Acesso: 30 Jun 2012.

SHNEIDERMAN, B.. **Designing the User Interface**. 3 ed. Massachusetts: Addison-Wesley, 1998.

SOMMERVILLE, I.. **Engenharia de Software**. 8 ed. São Paulo: Pearson / Addison-Wesley, 2007.