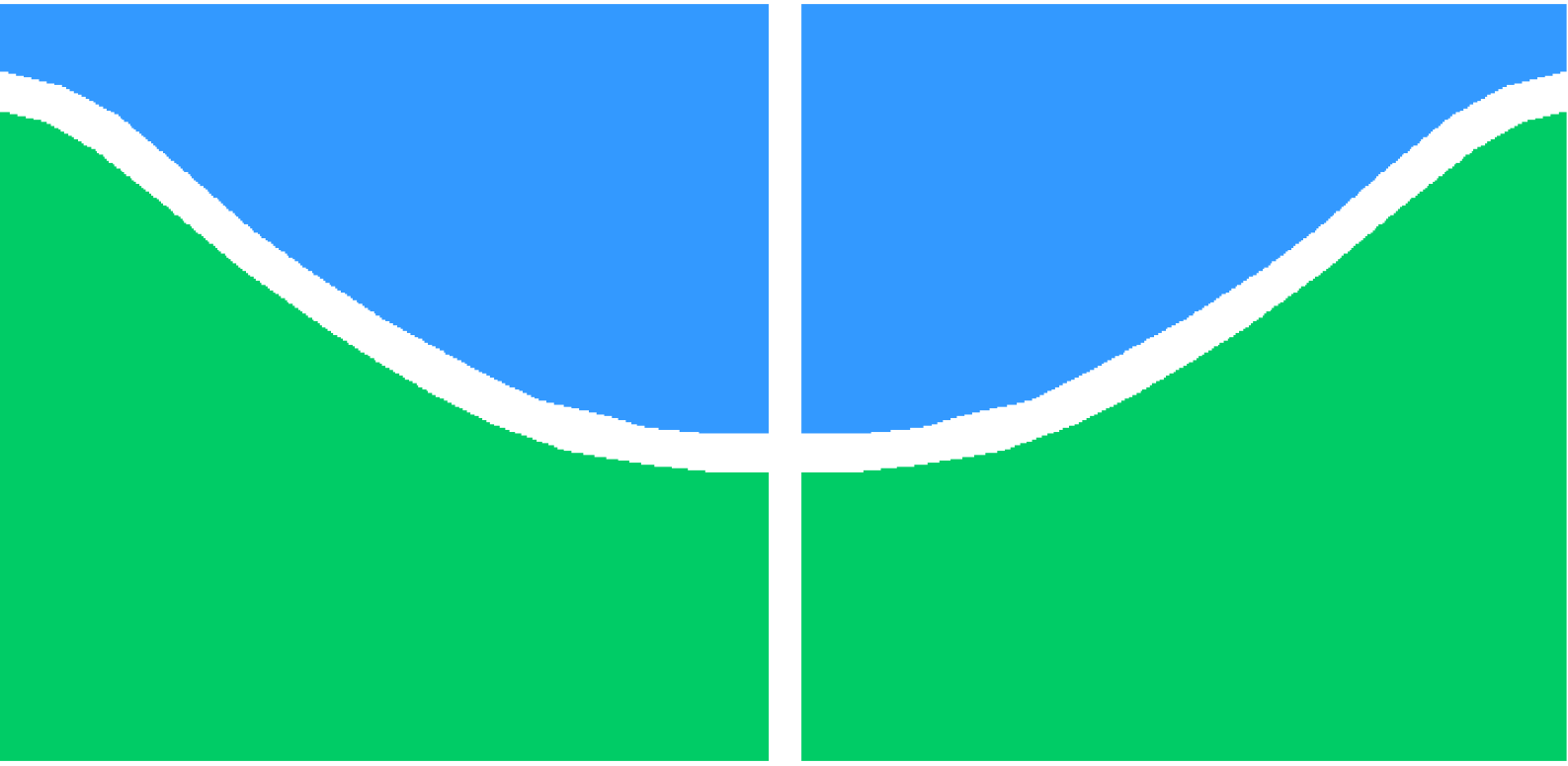


Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Engenharia de Software

Um estudo do relacionamento entre técnicas de usabilidade e testes automatizados em métodos empíricos de desenvolvimento de software

Autores: Rodrigo Medeiros Soares da Silva,
Jônatas Medeiros de Mendonça

Orientador: Prof. Dr. Paulo Roberto Miranda Meirelles

Brasília, DF

2014

Rodrigo Medeiros Soares da Silva,
Jônatas Medeiros de Mendonça

Um estudo do relacionamento entre técnicas de usabilidade e testes automatizados em métodos empíricos de desenvolvimento de software

Monografia submetida ao curso de graduação em Engenharia de Softwareda Universidade de Brasília, como requisito para obtenção do Título de Bacharel em Engenharia de Software.

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Prof. Dr. Paulo Roberto Miranda Meirelles

Brasília, DF

2014

Rodrigo Medeiros Soares da Silva,
Jônatas Medeiros de Mendonça
Um estudo do relacionamento entre técnicas de usabilidade e testes automatizados em métodos empíricos de desenvolvimento de software / Rodrigo Medeiros Soares da Silva,
Jônatas Medeiros de Mendonça . – Brasília, DF, 2014-
99 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Dr. Paulo Roberto Miranda Meirelles

Trabalho de Conclusão de Curso – Universidade de Brasília - UnB
Faculdade UnB Gama - FGA , 2014.

1. Testes automatizados. 2. Desenvolvimento empírico. 3. Usabilidade. I. Prof. Dr. Paulo Roberto Miranda Meirelles. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Um estudo do relacionamento entre técnicas de usabilidade e testes automatizados em métodos empíricos de desenvolvimento de software

CDU 02:141:005.6

Rodrigo Medeiros Soares da Silva,
Jônatas Medeiros de Mendonça

Um estudo do relacionamento entre técnicas de usabilidade e testes automatizados em métodos empíricos de desenvolvimento de software

Monografia submetida ao curso de graduação em Engenharia de Softwareda Universidade de Brasília, como requisito para obtenção do Título de Bacharel em Engenharia de Software.

Trabalho aprovado. Brasília, DF, Dezembro de 2014:

Prof. Dr. Paulo Roberto Miranda Meirelles
Orientador

Prof. Hilmer Rodrigues Neri
Convidado 1

Prof. Tiago Barros Pontes e Silva
Convidado 2

Brasília, DF
2014

Agradecimentos

Agradecemos primeiramente a Deus pela vida, pela sabedoria , e pela força dada durante todo este tempo.

Agradecemos aos nossos familiares por todo o apoio e suporte dado, sempre que foi preciso, por nunca medir esforços para tornar o sonho da graduação possível, pelos ensinamentos passados e pelo exemplo dado todos os dias.

Agradecemos também ao nosso orientador, por nos guiar durante todo este trabalho, pela paciência e pela instruções nos momentos em que precisamos e por acreditar no potencial do nosso trabalho.

Agradecemos à equipe do Portal do Software Público pela parceria dentro e fora do laboratório nesses últimos meses, pelo auxílio nas tarefas realizadas.

Resumo

Este trabalho de conclusão de curso de engenharia de software apresenta um estudo sobre usabilidade e testes automatizados no desenvolvimento empírico de software a fim de verificar como essas técnicas de testes podem se relacionar com as avaliações de usabilidade de um software. Para isso, realizamos um estudo sobre aplicabilidade de técnicas de usabilidade, além de técnicas de BDD (desenvolvimento de software dirigido por comportamento) em projetos de software, além de compreender como inserir as práticas de usabilidade estudadas no desenvolvimento empírico de software. Este estudo serviu de base para levantamento e apuração de algumas hipóteses a respeito da relação dos testes automatizados na usabilidade de um sistema de software.

Palavras-chaves: software livre. testes automatizados. usabilidade.

Abstract

This degree monograph presents a study about software usability and how to automate testing on the empirical software development to verify how automated testing may be related to software usability. For this, we conducted a study about applicability of BDD (Behaviour Driver Development) and usability techniques in software projects, as well as understand how to insert some studied techniques. In this study, we have tested some hypotheses to investigate the relationship between about automated testing approaches and software usability techniques.

Key-words: free software. automated tests. usability.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ASQ	The After-Scenario Questionnaire
BDD	Behavior Driven Development
DCU	Design Centrado no Usuário
DSA	Directory System Agent
FGA	Faculdade UnB Gama
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
ICC	Instituto Central de Ciências
IHC	Interação Humano Computador
LAPPIS	Laboratório de Produção, Pesquisa e Inovação em Software
LDAP	Lightweight Directory Access Protocol
MES	Manutenção e Evolução de Software
MVC	Model-View-Controller
ProIC	Projeto de Iniciação Científica
PSQ	The Printer-Scenario Questionnaire
PSSUQ	The Post-Study System Questionnaire
PuSH	PubSubHubbub
QUIS	Questionnaire for User Interaction Satisfaction
Rails	Ruby on Rails
SLTI	Secretaria de Logística e Tecnologia da Informação
SMT	Tecnologias de Mídia Social
SUMI	Usability Measurement Inventory

SUS	System Usability Scale
SSL	Secure Socket Layer
SPB	Software Público Brasileiro
TCC	Trabalho de Conclusão de Curso
TCP	Transmission Control Protocol
TDD	Test Driven Develepment
TLS	Transport Layer Security
UDP	User Datagram Protocol
UPA	Usability Professionals' Association - Associação de Profissionais de usabilidade
UnB	Universidade de Brasília
USP	Universidade de São Paulo
UI	User Interface
UX	User Experience
W3C	World Wide Web Consortium
XP	Extreme Programming

Sumário

1 Introdução

O desenvolvimento de software usando métodos empíricos, como software livre e métodos ágeis, é uma realidade. Uma prática básica de métodos ágeis são os testes automatizados, preocupando-se com as aplicações de testes em sistemas cada vez mais dinâmicos, com propostas para aumento de qualidade e produtividade de software (VICENTE, 2010). Adicionalmente, as áreas relacionadas a usabilidade vem sendo estudadas para serem aplicadas no contexto de desenvolvimento empírico de software. Criar software que seja útil e fácil de usar é um fator importante para a evolução dos sistemas (SANTOS, 2012).

Por exemplo, um problema encontrado em projetos de software livre é a pouca atenção dada aos aspectos referentes a usabilidade das aplicações. Para Moreira et al. (2012) enfatizar na criação, melhoria e teste do código fonte pode ser um dos principais problemas que contribui para a falta de usabilidade dos softwares livres. Ainda, segundo Preece, Rogers e Sharp (2007), uma das causas da baixa adoção de softwares livres em mercados de larga escala é a baixa qualidade dos estilos de interação implementados nas interfaces dos produtos.

Atualmente, algumas comunidades de software livre vêm se atentando à usabilidade como forma de manter o software no mercado. O Noosfero[1], um sistema para desenvolvimento de redes sociais na qual faz parte este estudo, tem tido algumas iniciativas por parte dos desenvolvedores da plataforma para a melhoria da usabilidade da ferramenta.

A adoção das práticas de usabilidade no contexto de métodos ágeis vêm sendo estudada por vários autores no qual afirmam que é possível a integração visto que tanto os métodos ágeis como os processo de usabilidade têm em comum características que colocam o foco do desenvolvimento nas necessidades e anseios dos usuários finais, na interação entre os *stakeholders* envolvidos e na qualidade final do produto a ser desenvolvido.

A automação de testes é uma prática ágil, eficaz e de baixo custo que também busca melhorar a qualidade dos sistemas de software (POTEL; COTTER, 1995). O que este trabalho busca é encontrar um ponto de intersecção entre os testes automatizados e a usabilidade, utilizados no desenvolvimento ágil.

1.1 Problemas

Um dos problemas encontrados no desenvolvimento empírico é a baixa usabilidade de suas interfaces, o que resulta na perda de usuários. Os desenvolvedores possuem

[1] <noosfero.org>

uma mentalidade mais voltada para a funcionalidade do que para os usuários do sistema (SANTOS, 2012). Outro problema do desenvolvimento empírico está em garantir qualidade de software de forma prática e econômica, testes podem interferir nestas características se não forem bem planejados.

A partir destes problemas levantamos as seguintes questões:

1. Como inserir os princípios de usabilidade no desenvolvimento empírico de software?
2. Como as práticas do BDD podem se relacionar com as práticas de usabilidade?

1.2 Objetivos

Durante este trabalho de conclusão de curso buscamos analisar técnicas e métodos de usabilidade, assim como técnicas de desenvolvimento de testes no desenvolvimento empírico de software para verificarmos as possíveis relações de técnicas de BDD (Behavior Driven Development) na usabilidade de um software.

Para satisfazer os objetivos gerais do trabalho definimos objetivos específicos abaixo:

1. Integrar práticas de usabilidade no ciclo de vida de desenvolvimento empírico de software;
2. Identificar quais técnicas de usabilidade podem ser utilizadas em cada fase do desenvolvimento empírico.
3. Verificar a relação de práticas de BDD e práticas de usabilidade no desenvolvimento empírico de software.

1.3 Organização do Trabalho

Nesta seção apresentamos a organização deste trabalho, e o que se refere cada capítulo. O estudo inicia-se no segundo capítulo, em que se aborda o desenvolvimento empírico, onde dissertamos sobre software livre assim como uma breve base conceitual sobre metodologias ágeis de desenvolvimento. O terceiro capítulo aborda conceitualmente as técnicas de testes automatizados que geralmente são utilizadas no processo empírico, além dos tipos de testes que podem ser aplicados. Nesta seção destaca-se as práticas do BDD. O quarto capítulo aborda os principais conceitos e as áreas referentes à usabilidade, além de a abordar os processos da engenharia de usabilidade e interação humano computador e as formas de como poder inserir as técnicas no ciclo de vida de desenvolvimento empírico. O quinto capítulo aborda o estudo de caso realizado no Portal do Software Público Brasileiro (SPB), baseado na plataforma Noosfero, assim como os resultados obtidos. O

trabalho se encerra com as conclusões, assim como as análises obtidas de todo o processo do trabalho.

2 Métodos Empíricos de Desenvolvimento de Software

O Empirismo baseia-se na aquisição de sabedoria através da percepção do mundo externo, ou então do exame da atividade da nossa mente, que abstrai a realidade que nos é exterior e as modifica internamente (CHAUí, 2003). Mecanismos do controle de processo empírico, onde ciclos de *feedback* são a base da técnica de gerenciamento de projetos. É uma forma de planejar e gerenciar projetos trazendo a autoridade da tomada de decisão em níveis de propriedade de operação e certeza (SCHWABER, 2004). Neste capítulo serão abordadas algumas ideias e práticas de processo de desenvolvimento de software que utilizam métodos empíricos, como os métodos ágeis e o processo de desenvolvimento de software livre.

2.1 Software Livre

De acordo com Stallman (2001), ativista fundador do movimento software livre, um software livre deve seguir quatro princípios:

1. Liberdade de execução para qualquer uso;
2. Liberdade de estudar o funcionamento de um software e de adaptá-lo às suas necessidades
3. Liberdade de redistribuir cópias;
4. Liberdade de melhorar o software e de tornar as modificações públicas de modo que a comunidade se beneficie da melhoria.

Um software é considerado software livre se segue esses quatro princípios, portanto o usuário deve poder utilizar, estudar e modificar o software como ele bem entender. Não significa que o software é necessariamente de graça, mas a partir do momento em que se obtém posse de um programa um usuário pode modificar e redistribuir o mesmo programa. Para Stallman (2001), a partir do momento em que os custos de desenvolvimento de um software são pagos, não há motivos para restrição de acesso, pois a disseminação de conhecimento é muito mais benéfica do que potenciais lucros para o produtor.

2.2 Métodos Ágeis

A utilização de métodos ágeis no desenvolvimento de software tem como características intrínsecas a flexibilidade e rapidez nas respostas a mudanças. A agilidade, para uma organização de desenvolvimento de software, é a habilidade de adotar e reagir rapidamente e apropriadamente a mudanças no seu ambiente e por exigências impostas pelos "clientes" (NERUR; MAHAPATRA; MANGALARAJ, 2005). Os métodos ágeis compartilham valores como comunicação, *feedback* constante, colaboração com o cliente e constante adaptação são baseados no manifesto ágil[1]. Os quatro princípios básicos do manifesto ágil mostram o que se espera de qualquer método de desenvolvimento dessa categoria:

1. Indivíduos e interações sobre processos e ferramentas;
2. Software funcionando sobre documentação extensiva;
3. Colaboração com o cliente sobre negociação de contrato;
4. Responder às mudanças sobre seguir um planejamento;

Existe uma relação entre as práticas ágeis e o desenvolvimento baseado em software livre. Corbucci (2011) fala que o desenvolvimento baseado em software livre vem crescendo, porém a essência da comunidade ao redor do programa é de manter indivíduos que interajam de forma a produzir o que é mais importante. As ferramentas apenas possibilitam isso. Quanto à documentação Corbucci (2011) sugere que documentação completa e detalhada cresce com a comunidade ao redor do software funcionando.

Para Corbucci (2011) os clientes podem colaborar com projetos baseados em software livre, porém não são induzidos a fazê-lo, por conta da pouca experiência das comunidades em relacionamento com o cliente. Em projetos ágeis, o cliente/usuário é mais ativo durante o processo de desenvolvimento, determinando em conjunto com a equipe de desenvolvimento o que será desenvolvido, além de participar da validação do o que foi desenvolvido. Sobre mudanças Corbucci (2011) fala que a habilidade de responder às mudanças é crucial para determinar os projetos que sobrevivem.

Os projetos ágeis também buscam estabelecer um tempo determinado e curto para entrega de novas versões do sistema, com o objetivo de trazer mais satisfação ao cliente/usuário. A partir desses curtos ciclos a equipe de desenvolvimento deve se preocupar mais com a evolução dos requisitos, que pode gerar mudanças, porém mantém o projeto atualizado e diminui o riscos de grandes mudanças a medida que o projeto chega ao final. Um método ágil conhecido como programação extrema (*Extreme Programing* -

[1] <http://manifestoagil.com.br>

XP), tornou-se bastante popular por utilizar práticas focadas em codificação. O objetivo principal do XP é a excelência no desenvolvimento de software, visando baixo custo e poucos defeitos, de acordo com Sato (2007). O XP conta com algumas práticas que são: refatoração, integração contínua, testes automatizados, código coletivo e programação em pares.

Em suma, o ambiente de desenvolvimento que este trabalho está inserido é baseado em métodos de desenvolvimento empíricos de software, como XP e Software Livre, buscando seguir os princípios ágeis para o desenvolvimento de software livre.

O desenvolvimento baseado em software livre e as práticas ágeis compartilham os mesmos valores, pois ambos são métodos empíricos, que buscam tomar decisões rápidas de acordo com a percepção do mundo externo. Nos próximos capítulos abordaremos como o desenvolvimento de testes é abordado no desenvolvimento empírico e como a usabilidade pode ser inserida neste contexto.

3 Testes

Testar é uma pratica intrínseca ao desenvolvimento e é antiga a necessidade de criar programas para testar cenários específicos, de acordo com Everett et al. (2007). Testes de software serão abordados neste capítulo, iniciando com uma visão geral de testes automatizados e conhecendo algumas práticas e padrões utilizados para desenvolver os mesmos.

Para Potel e Cotter (1995) a automação de testes é uma prática ágil, eficaz e de baixo custo para melhorar a qualidade dos sistemas de software. No entanto utilizar testes automatizados como uma premissa básica do desenvolvimento é um fenômeno relativamente recente, com início em meados da década de 1990. Além do fato de ser uma técnica bastante utilizada pelas metodologias ágeis de desenvolvimento.

3.1 Testes Automatizados

Teste automatizado é a prática de tornar os testes de software independentes da intervenção humana, criando *scripts* ou programas simples de computador que exercitam o sistema em teste, capturam os efeitos colaterais e fazem verificações, tudo automática e dinamicamente (MESZAROS; WESLEY, 2007). Os testes automatizados afetam diretamente a qualidade dos sistemas de software, portanto, agregam valor ao produto final, mesmo que os artefatos adicionais produzidos não sejam visíveis para os usuários finais dos sistemas (BERNARDO, 2011).

Será abordado neste trabalho o desenvolvimento de testes de aceitação para verificar sua relação com a usabilidade.

3.1.1 Testes de aceitação

Os testes de aceitação, que possuem uma visão mais voltada para o usuário, fazem parte de uma fase do processo em que um teste de caixa-preta é realizado num sistema antes de sua disponibilização. Para isso utilizamos o cucumber[1], uma ferramenta que pode executar uma funcionalidades escrita em texto puro. Com base nas especificações da funcionalidade, o cucumber executa testes, proporcionando uma melhor comunicação entre equipe de desenvolvimento e cliente. No cucumber, uma *feature* é um requisito de alto nível expressado da perspectiva de uma pessoa e possui uma estrutura similar as histórias de usuário do XP (CHELIMSKKY et al., 2010).

[1] <cukes.info>

Os testes de aceitação foram utilizados neste trabalho a fim de analisar possíveis influencias durante a inserção de práticas de usabilidade no desenvolvimento.

3.1.2 Testes Funcionais e Unitários

Os testes funcionais tem como objetivo no desenvolvimento na plataforma noosfero, verificar a integração da aplicação desenvolvida. Os testes unitários são executados em conjunto com os testes funcionais, porém com o objetivo de verificar trechos menores de códigos.

A descrição detalhada de testes de aceitação e de testes funcionais e unitários encontra-se no apêndice B.

3.2 Técnicas de desenvolvimento de testes automatizados

Automação de testes é uma técnica voltada principalmente para a melhoria de qualidade dos sistemas de software. No processo de desenvolvimento de software é fundamental controlar o custo do processo de testes, para isso baterias de testes automatizados devem ser bem definidas e implementadas. Assim, é importante conhecer boas práticas e técnicas de desenvolvimento de testes automatizados. Existem várias técnicas de desenvolvimento de software com testes que influenciam diretamente na qualidade do sistema. Essas técnicas geralmente possuem um processo de atividades pequeno e simples, como TDD e BDD.

3.2.1 TDD - Test Driven Development

Desenvolvimento dirigido por testes, TDD *(Test-Driven Develepment)*, é uma técnica de desenvolvimento de software que se dá pela repetição disciplinada de um ciclo curto de passos de implementação de testes e do sistema (KOSKELA, 2007). O ciclo de TDD é definido pelos seguintes passos:

1. Implementar um caso de teste;
2. Implementar um trecho do código suficiente para o novo caso de teste ter sucesso de tal modo que não quebre os testes previamente escritos;
3. Se necessário, refatorar o código produzido para que ele fique mais organizado;

A técnica de desenvolvimento dirigido por testes foi definida por Beck (2002), como representados na Figura 1. Uma boa prática do TDD é a bateria de testes, que ajuda o desenvolvedor a evitar erros de regressão, quando o desenvolvimento de uma nova funcionalidade quebra uma já existente. TDD também tende a contribuir com uma

Figura 1: Ciclo de atividades TDD (BECK, 2002)

alta cobertura de código, uma fez que o desenvolvedor precisa escrever o testes antes da funcionalidade, possibilitando a criação de um código mais preciso, coeso e menos acoplado.

Para Massol e Husted (2003), o objetivo de TDD é código claro que funciona. TDD propõe o desenvolvimento sempre em pequenos passo, deve-se escrever testes sempre para uma menor funcionalidade possível, escrever o código mais simples que faça o teste passar e fazer sempre apenas uma refatoração por vez (BECK, 2002). Assim o desenvolvedor se detém a criar soluções simples, sempre acompanhado de um constante *feedback* dos testes. O ciclo curto de passos definidos por TDD cria uma dependência forte entre codificação e testes, o que favorece e facilita a criação de sistemas com alta testabilidade (BERNARDO, 2011). Índices altos de cobertura de código e testabilidade não garantem necessariamente qualidade do sistema, mas são métricas bem vistas para sistemas bem desenvolvidos.

3.2.2 BDD - Behavior Driven Development

Desenvolvimento dirigido por comportamento é uma prática que recomenda o mesmo ciclo de desenvolvimento de TDD, contudo, induzindo a utilização de uma linguagem ubíqua entre cliente e equipe de desenvolvimento, substituindo termos como assert, *assert, test case, test suite* por termos mais comuns ao cliente, como *should, context, specification* (BERNARDO, 2011).

Embora seja principalmente uma ideia de como um processo de desenvolvimento de software deve ser gerenciado, a prática do BDD assume a utilização de ferramentas como suporte para o desenvolvimento de software (HARING, 2011). O BDD coloca em foco o comportamento em vez da estrutura de uma funcionalidade e faz isso em todos os

níveis de desenvolvimento. Uma vez reconhecido isso, muda-se a forma de pensar sobre desenvolvimento para fora do código. Assim o foco parte para as interações, as pessoas e o sistema, sobre a estrutura do próprio sistema (CHELIMSKKY et al., 2010).

Para que o comportamento seja analisado, é necessário entender o ponto de vista do cliente/usuário, entendendo o comportamento que o sistema deve ter a partir da visão do cliente/usuário. De acordo com Chelimskky et al. (2010), estes são os três princípios do BDD:

1. **O suficiente é suficiente:** parte da ideia de gerenciar o esforço no planejamento inicial do sistema, para não fazer menos nem mais do que o necessário para começar, o que se aplica também ao processo de automação.
2. **Agregar valor às partes interessadas:** Se você está fazendo algo que não agrega valor ou que não aumenta a capacidade de agregar valor, pare e faça outra coisa em seu lugar.
3. **Tudo é comportamento:** Do código à aplicação, pode-se usar o mesmo pensamento e as mesmas construções linguísticas para descrever o comportamento, em qualquer nível de granularidade.

O comportamento do sistema é descrito em histórias de usuário, que são escritas com a participação tanto de clientes como desenvolvedores do sistema.

Testes automatizados devem ser desenvolvidos com prioridade, buscando um rápido *feedback*, contribuindo assim com a melhoria do sistema. Para isso é necessário que os cenários de testes estejam bem definidos junto à equipe. Considerando que o BDD se econtra bem adotado no ciclo de desenvolvimento deste estudo de caso, buscaremos nos próximos capítulos formas de abordar a usabilidade também no processo de desenvolvimento empírico, para na sequência mostrarmos o estudo de como relacionar as práticas de testes e as técnicas de usabilidade.

4 Usabilidade

Neste capítulo abordaremos os conceitos de usabilidade e suas relações, além de entender a sua importância em um software, apresentar os modelos de ciclo de vida utilizados na usabilidade e mostrar como pode ser inserida a usabilidade no ciclo de desenvolvimento empírico de software.

4.1 Conceito de Usabilidade

A usabilidade é um termo que começou a ser utilizado pela Ciência Cognitiva e depois pela Psicologia e Ergonomia em substituição ao termo "amigável" (DIAS, 2006). A usabilidade é um conjunto de propriedades de uma interface que reúne os seguintes componentes: fácil aprendizado, eficiência, capacidade de memorização, baixo índice de erros, satisfação e prazer ao uso (NIELSEN, 1994). Ainda de acordo com Nielsen e Loranger (2007) usabilidade é um atributo de qualidade relacionado à facilidade de uso de algo. Refere-se a rapidez com que os usuários podem aprender a usar alguma coisa, a eficiência deles ao usá-la, o quanto lembram daquilo, seu grau de propensão a erros e o quanto gostam de utilizá-la.

A norma ISO/IEC 9241-11 (2003) define alguns termos que são amplamente utilizados quando trata-se de usabilidade:

- **Eficácia:** é a capacidade que os sistemas conferem a diferentes tipos de usuários para alcançar seus objetivos em número e com a qualidade necessária.
- **Eficiência:** é a quantidade de recursos que os sistemas solicitam aos usuários para a observação de seu objetivos com o sistema.
- **Satisfação:** é a emoção que os sistemas proporciona, aos usuários em face dos resultados obtidos e dos recursos necessários para alcançar tais objetivos.
- **Tarefa:** Objetivo a ser atingido ou um resultado a obter.
- **Usuários:** Pessoas que utilizam alguma instância do sistema.
- **Contexto:** Conjunto de elementos, onde se destacam: o ambiente físico, a tecnologia utilizada e a a motivação.

Adicionalmente, existem metas de usabilidade que servem para guiar o desenvolvimento de um software (PREECE; ROGERS; SHARP, 2007). São elas:

1. **Utilidade:** é a medida na qual o sistema propicia o tipo certo de funcionalidade, de maneira que os usuários possam realizar aquilo de que precisam ou desejam.
2. **Eficácia:** refere-se ao quanto um sistema é bom em fazer o que se espera dele.
3. **Eficiência:** refere-se à maneira como o sistema auxilia os usuários na realização de suas tarefas.
4. **Segurança:** implica em proteger o usuário de condições perigosas e situações indesejáveis.
5. **Facilidade de aprendizado:** refere-se a qual fácil é aprender a usar o sistema.
6. **Facilidade de lembrar como se usa:** refere-se à facilidade de lembrar como utiliza um sistema, depois de já ter aprendido como usar.

A usabilidade é um termo transversal, utilizada em diversas áreas e por diversos profissionais como designs, arquitetos de informação, engenheiros de software, ergonomistas e etc. Para entender como usabilidade está inserida no ciclo de vida do desenvolvimento de software, precisamos compreender as relações que o termo tem com as diversas áreas que a envolve:

Interação Humano Computador (IHC): A interação humano computador tem o principal objetivo de melhorar a eficácia e proporcionar satisfação do usuário, desenvolvendo sistemas computacionais nos quais os usuários possam executar tarefas com segurança, eficiência e satisfação (PREECE; ROGERS; SHARP, 2007). Esse é o campo interdisciplinar mais conhecido que se preocupa com o design, a avaliação e a implementação de sistemas computacionais para uso humano.

Arquitetura da Informação: A arquitetura da informação é a ciência de estruturar, bem como, de organizar os ambientes informacionais para ajudar as pessoas a encontrarem e administrarem informações (GARRET, 2003).

***Design* Centrado no usuário (DCU):** Para Norman (2006) DCU é uma filosofia baseada nas necessidades e interesses dos usuário, com ênfase em fazer produtos usáveis e inteligíveis. As pessoas que utilizarão um produto ou serviço sabem de suas necessidades, metas e preferências, e é papel do *design* descobrir isso para projetar para eles – essa é a filosofia por trás do DCU (SAFFER, 2010).

Design de Interação: O *design* de interação redireciona a preocupação que se tinha apenas em produzir algo que funcionasse para produzir algo que será fácil de utilizar, agradável e eficaz na perspectiva do usuário, trazendo a usabilidade para dentro do processo de *design* (PREECE; ROGERS; SHARP, 2007). O design de interação é

fundamental para todas as disciplinas, campos e abordagens que se preocupam em criar sistemas com usabilidade.

Engenharia de Usabilidade: A engenharia de usabilidade ocupa-se da interface com o usuário que interliga as funções do sistema com os usuários de forma que a interface do sistema seja agradável, intuitivo, eficiente e fácil de operar (CYBIS; BETIOL; FAUST, 2010). Já a engenharia de software ocupa do desenvolvimento do núcleo funcional de um sistema interativo formado por estruturas de dados, algoritmos e recursos de processamento de dados. Esse núcleo é construído de forma que o sistema funcione bem, de forma correta, rápida e sem erros.

Experiência do Usuário - UX: Toda a interação com um produto, serviço ou marca. O termo é usado frequentemente para sintetizar toda a experiência com um produto de software. Não engloba somente as funcionalidades e sim o quanto o aplicativo é agradável ao usuário (LOWDERMILK, 2013).

A usabilidade está relacionada com os fatores humanos e corresponde ao estudo de como usuários se relacionam com qualquer produto. A interação humano computador está baseada na usabilidade e foca no modo como as pessoas se relacionam com softwares. Várias outras áreas de estudo estão preocupadas em projetar sistemas que atendam aos objetivos dos usuários tendo como principais diferenças o foco e as metodologias utilizadas em seu processo.

4.2 Métodos da Engenharia de Usabilidade

Existem diversas técnicas e métodos que são utilizados pela interação humano computador para desenvolver um sistema com usabilidade. Cybis, Betiol e Faust (2010) divide as técnicas da engenharia de usabilidade em três tipos: Concepção, Análise e Avaliação.

4.2.1 Métodos de Concepção

Os métodos de concepção são utilizados para implementar as especificações e os requisitos para a interface e usabilidade de um sistema, neste trabalho utilizamos prototipação:

Prototipação: Os protótipos de papel são úteis para detectar problemas de usabilidade logo no início do proceso de design. São usados para esclarecer os requisitos específicos para o projeto da interface do sistema. São rápidos de construir, permitindo rápidas interações de projeto e necessitam de poucos recursos para serem criados, economizando tempo de design e desenvolvimento. Existem também os protótipos

de baixa, média e alta que simulam o sistema com mais fidelidade do que os protótipos em papel.

No apêndice A estão detalhadas outros métodos de concepção.

4.2.2 Métodos de Análise

Os métodos de análise são utilizados para buscar informações sobre a usabilidade de um sistema. Podem ser feitas análises do perfil do usuário, do ambiente de uso, das tarefas, possibilidades e restrições do sistema, etc. Neste trabalho foi utilizado o método de cenário de uso;

Cenário de Uso: É uma técnica simples e eficaz para analisar e comunicar uma parte das especificações de requisitos produzidos para a usabilidade e a interface. São utilizados para comunicar os cenários atuais de uma tarefa (problema) e o cenário futuro (solução). Os cenários de solução deve descrever em linguagem natural, como determinados usuários realizarão tarefas específicas com o sistema em um determinado contexto. É preciso decompor os objetivos dos usuários segundo as operações necessárias para alcança-los, identificando as atividades que serão realizadas pelos usuários (CYBIS; BETIOL; FAUST, 2010).

No apêndice A estão detalhadas outros métodos de análise.

4.2.3 Métodos de Avaliação da Usabilidade

Utilizados para avaliar a qualidade das interações entre o usuário e o sistema. As avaliações de interface podem ser divididas quanto a: formas, tipo de dados e o local da avaliação. Quanto a forma, temos objetiva, quando são baseadas em técnicas que utilizam medições quantitativas em vez de opiniões dos usuários/especialistas e subjetivas, quando são baseadas em opiniões e relatos de usuários e especialistas. Já quanto ao tipo de dados podem ser definidas como: quantitativa, quando envolvem medidas e tendem a ser vistas como objetivas e imparciais, e qualitativas quando envolvem descrições e relatos e são vistas como subjetivas. Quanto ao local da avaliação podem ser feita em laboratório, onde o ambiente é controlado ou estudo de campo quando estão situadas no contexto do mundo real no qual o sistema é utilizado. No estudo de caso aplicamos as seguintes técnicas:

Checklists: É uma técnica de inspeção que oferece uma maneira de abordagem rápida, fácil e de baixo custo para a avaliação de uma interface. Permite com que usuários que não são especialistas identifiquem problemas menores e repetitivos (CYBIS; BETIOL; FAUST, 2010). Várias vantagens são encontradas na utilização

dessa técnica como: redução de custos de avaliação por não precisar de especialistas; sistematização das avaliações; reduz a subjetividade dos processos de avaliação e fornece deversas questões sobre o que avaliar. Os checklists utilizados encontra-se na seção apêndice A.5.

Avaliação heurística: Representa um julgamento de valor sobre as qualidades ergonômicas das interfaces Humano-Computador. Essa avaliação é realizada por especialistas em ergonomia e usabilidade. Eles examinam o sistema interativo e diagnosticam os problemas ou as barreiras que os usuários provavelmente encontrarão durante a interação (CYBIS; BETIOL; FAUST, 2010).

O principal objetivo desse tipo de avaliação é avaliar a interface em fases iniciais do sistema, sem o envolvimento do usuário. Os graves problemas de interface devem ser localizados antes que realizem os testes de usabilidade com usuários reais (SANTOS, 2012).

As avaliações mais utilizadas são as 10 heurísticas de Nielsen (1994). São elas:

- **Diálogos simples e naturais:** Deve-se apresentar exatamente a informação que o usuário precisa no momento, nem mais nem menos. A seqüência da interação e o acesso aos objetos e operações devem ser compatíveis com o modo pelo qual o usuário realiza suas tarefas.
- **Falar a linguagem do usuário:** A terminologia deve ser baseada na linguagem do usuário e não orientada ao sistema. As informações devem ser organizadas conforme o modelo mental do usuário.
- **Minimizar a sobrecarga de memória do usuário:** O sistema deve mostrar os elementos de diálogo e permitir que o usuário faça suas escolhas, sem a necessidade de lembrar um comando específico.
- **Consistência e Padrões:** Um mesmo comando ou ação deve ter sempre o mesmo efeito. A mesma operação deve ser apresentada na mesma localização e deve ser formatada/apresentada da mesma maneira para facilitar o reconhecimento.
- **Feedback:** O sistema deve informar continuamente ao usuário sobre o que ele está fazendo. 10 segundos é o limite para manter a atenção do usuário focalizada no diálogo.
- **Saídas claramente marcadas:** O usuário controla o sistema. Ele pode, a qualquer momento, abortar uma tarefa, ou desfazer uma operação e retornar ao estado anterior.
- **Atalhos:** Para usuários experientes executarem as operações mais rapidamente. Abreviações, teclas de função, duplo clique no mouse, função de volta

em sistemas hipertexto. Atalhos também servem para recuperar informações que estão numa profundidade na árvore navegacional a partir da interface principal.

- **Boas mensagens de erro:** Linguagem clara e sem códigos. Devem ajudar o usuário a entender e resolver o problema. Não devem culpar ou intimidar o usuário.
- **Prevenir erros:** Evitar situações de erro. Conhecer as situações que mais provocam erros e modificar a interface para que estes erros não ocorram.
- **Ajuda e documentação:** O ideal é que um software seja tão fácil de usar (intuitivo) que não necessite de ajuda ou documentação. Se for necessária a ajuda deve estar facilmente acessível e on-line.

As avaliações podem ser feitas em planilhas (como a figura 4.2.3), onde são descritos:

- **ID:** Número sequencial do erro.
- **Data:** Data em que o erro foi identificado.
- **Local do problema de usabilidade:** Local onde o problema foi encontrado, podendo conter:

  Nome da tela;

  Caminho de Navegação;

  Campo ou tela onde foi encontrado o erro;
- **Descrição:** Descrição do problema de usabilidade.
- **Heurística desobecida:** Regra geral, diretriz ou orientação que foi desobedecida.
- **Criticidade:** Grau de erro, podendo ser:

  Alta: Obstáculo grave ou que impede o uso do sistema;

  Média: Obstáculo que dificulta pouco o uso do sistema;

  Baixa: Problema de perfumária do sistema;

A avaliação heurística requer pouco planejamento e pode fazer parte de um processo interativo de um projeto e aplicado em todas as fases de desenvolvimento da interface, mas é preciso conhecer as heurísticas para serem aplicadas.

No apêndice A estão detalhadas os outros métodos de avaliação .

Planilha de Análise Heurística			Qtd Heurísticas desobedecidas	###
Site:			Grave	###
Inspetor:			Alta	###
Ambiente / Browser:			Média	###
ID	Data	Local do problema de usabilidade	Descrição	Heurística desobedecida	Criticidade
###					
###					
###					
###					
###					
###					

Figura 2: Planilha de Avaliação

4.3 Processos de Usabilidade

Alguns autores como Mayhew (1999) e Hix e Hartson (1993) propuseram modelos de processo baseados em ciclo de vidas bem definidos para as atividades de usabilidade. O modelo estrela e o de Engenharia de Usabilidade propostos pelos autores estão definidos no A.

4.3.1 Design Centrado no Usuário - DCU

Definido pela norma ISO/IEC 13407 (1999), o DCU, tem como objetivo definir um processo necessário ao desenvolvimento de produtos fáceis de utilizar na qual deve envolver os usuários no processo de desenvolvimento.

Sabemos que a ciência experimental que utiliza-se dos métodos empíricos tradicionais para coletar dados e testar hipóteses. Essa abordagem preocupa-se com os usuários quando as representa mas ela não leva em conta diversos aspectos do usuário real, por não envolvê-los no processo. Não basta fazer para o usuário, é preciso fazer com o usuário (EASON, 2005).

No DCU o usuário tem que ser o elemento central e é necessário envolvê-lo do início ao fim do projeto. Baseia-se nas necessidades, desejos e limitações das pessoas. Lowdermilk (2013) afirma que "para criar produtos que os usuário amem, é necessário incluir os usuários no processo de criação dos produtos".

Existem quatro atividades de DCU que devem estar presentes no inicío do projeto:

- Entender e especificar o contexto de uso;
- Especificar os requisitos de usuário;
- Produzir soluções de *design*;
- Avaliar o *design* frente aos requisitos;

A ISO/IEC 13407 (1999) propõe que o envolvimento do usuário seja uma prática frequente em empresas que desenvolvem sistemas interativos.

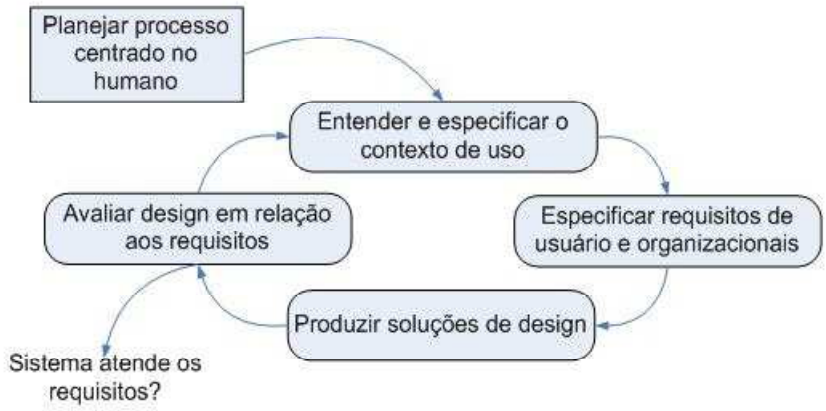


Figura 3: Ciclo do *design* centrado no humano (ISO/IEC 13407, 1999).

4.4 Aplicação de Usabilidade nos Métodos Empíricos

Partindo do que foi conceituado sobre desenvolvimento empírico e IHC, podemos ver semelhança entre seus valores, o que proporciona um quadro favorável à integração de práticas de usabilidade em desenvolvimento empírico. Podemos citar alguns desse valores: (i) ciclos curtos com entregas contínuas e incrementais, que favorecem a aplicação de técnicas de prototipagem; (ii) forte envolvimento do usuário que favorece a aplicação de princípios de projetos participativos e (iii) programação em pares onde em IHC geralmente a avaliação de usabilidade é feita em pares (BARBOSA; FURTADO; GOMES, 2008). Segundo Cybis, Betiol e Faust (2010) as características da abordagem ágil facilitam na utilização da ergonomia e da usabilidade durante o desenvolvimento de software.

Porém adaptações são necessárias, principalmente na questão da granularidade da pesquisa, no tempo gasto com ela e na maneira de relatar as descobertas de usabilidade (SANTOS, 2012). Hodgetts (2005) afirma que os métodos de desenvolvimento empírico são quase sempre apresentados sob a visão dos programadores, deixando de lado outras disciplinas como Design de UX.

No modo geral, os projetistas sabem da importância de desenvolver com enfoque no usuário e na usabilidade, mas normalmente os projetos são desenvolvidos sem que tenham sido realizadas pesquisas e aplicados métodos e técnicas de usabilidade. O tempo e os recursos limitados são as principais razões que impede a implantação dos testes de usabilidade nas equipes de desenvolvimento de software. Também há o desconhecimento por parte da equipe de desenvolvimento das técnicas a serem empregadas.

4.4.1 Integração de práticas

Alguns estudos foram encontrados tentando integrar práticas de usabilidade com os métodos empíricos. Constantine e Lockwood (2002) foi um dos primeiros a iniciar a discussão sobre o tema. Sua proposta é mostrar que o DCU está prontamente integrado aos métodos ditos "leves". Ele afirma que XP e os demais métodos empíricos compartilham das mesmas fraquezas apresentadas por processos tradicionais como o processo unificado quando se refere à usabilidade e ao desenho de interface do usuário.

Em 2008 foi proposto uma metodologia revisional intitulada XPlus que insere as técnicas de *design* e de usabilidade em determinadas fases do ciclo de desenvolvimento de software. O XPlus agrega algumas práticas de UX às práticas tradicionais de XP. A figura 4 mostra o processo de desenvolvimento de software de XPlus (GUIMARAES; CHAVES,).

O papel de *designer* de interação não é reconhecido no núcleo XP da equipe e o XP não tem nenhum processo explícito para lidar com o *design* de interação.

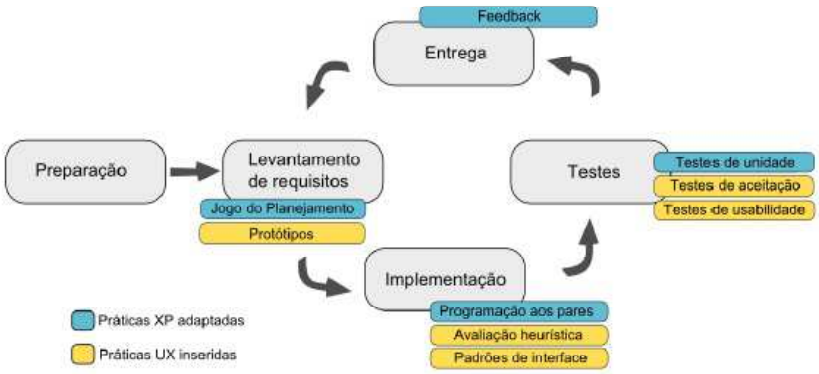


Figura 4: Visão geral do processo de desenvolvimento de software de XPlus (GUIMARAES; CHAVES,)

O XPU, processo ágil centrado em usabilidade descrito por Vasconcelos, Garcia e Turnell (2003) tinha como objetivo prover à equipe de desenvolvimento um modelo para a construção de sistemas de software centrado no usuário que valorizasse a usabilidade como característica fundamental da qualidade.

O XPU serve para guiar o time de desenvolvimento seguindo técnicas e artefatos simplificados afim de não se ter apenas sistemas portáteis reutilizáveis ou acoplados, mas sim com uma interface que reflete as características e as necessidades dos usuários. As atividades no XPu estão divididas em atividades técnicas e gerenciais. As atividades técnicas estão relacionadas à execução correta do processo de usabilidade, já as gerenciais garantem o controle e execução dessas práticas.

Nas próximas tabelas são apresentadas as atividades técnicas e gerenciais de cada visão (projeto, iteração e desenvolvimento) estabelecidas pelo XP e adaptadas pelo método XPu:

- Visão de Projeto
  - Atividades Técnicas

Atividade	**Técnica**	**Descrição**
Análise de Usuários	Personas	Caracterizar grupos de usuários finais. Descrever expectativas de um usuário em relação ao desenvolvido.
Análise de Tarefas	Roteiros	Propõe uma narrativa textual para descrever a execução de uma tarefa pelo usuário.
Análise de Contexto de uso	Histórias de usabilidade	São formadas pelo Roteiro sendo realizada pela Persona.
Definição de requisitos e metas de usabilidade	Benchmarks	Servem de balizadores para avaliar a usabilidade. São gerados a partir dos atributos de usabilidade
Avaliação da Usabilidade	Cenários de Testes	São utilizados como insumos de avaliações de usabilidade.
Avaliação da Usabilidade	Testes de Usabilidade	Estes testes podem ser feitos através de avaliações preditivas, experimentais.

Tabela 1: Atividades Técnicas de Visão de Projeto

– Atividades Gerenciais

Atividade	**Técnica**	**Descrição**	Insumos	Saídas
Planejamento de usabilidade	Planning Poker	Ocorre simultaneamente ao planejamento da release. O Projetista de interação contribui com o seu parecer sobre as estimativas de tamanho das histórias de usabilidade.	Histórias de usabilidade e Planejamento das avaliações	Estimativas de Recursos (Tempo e Custo)
Controle de Usabilidade	Velocity	Acompanhamento das tarefas relativas à usabilidade, comparando o esforço previsto com o realizado.	Estimativas de recursos	Refinamento de estimativas e Velocidade do projeto

Tabela 2: Atividades Gerenciais de Visão de Projeto

- Visão de Iteração

Atividade	**Técnica**	**Descrição**	Insumos	Saídas
Definição de estilos de interação.	Template	São modelos estruturados de interface que permite definir um guia de estilo padrão. O guia de estilo descreve diretrizes e padrões para o desenvolvimento de software.	Requisitos de Usabilidade	Guia de estilos (Templates)
Desenho de Interação	Interface Final	As interfaces geradas devem estar em conformidade com os templates definidos pelo projeto.	Templates	Interfaces finais

Tabela 3: Atividades Técnicas de Visão de Interação

- Visão de Desenvolvimento

Atividade	**Técnica**	**Descrição**	Insumos	Saídas
Prototipação de Interface	Prototipação Rápida	É feita através de protótipos de baixa fidelidade em papel. São feitos pelos projetistas de interação e apresentados aos usuários com objetivo de avaliá-los e discutir alternativas de desenhos.	Templates	Protótipo Simplificado
Avaliação de Usabilidade	Podem ser utilizadas várias técnicas	Diversas avaliações podem ser utilizadas durante o ciclo de desenvolvimento.	Cenários de Testes	

Tabela 4: Atividades Técnicas de Visão de Desenvolvimento

4.4.2 DCU Ágil

Pode ser definido o DCU ágil como a prática de DCU quando conduzida dentro de uma metodologia ágil (SANTOS, 2012). O ciclo de DCU ágil foi proposto por Sy (2007) e integra tantos as atividades de DCU como as de desenvolvimento em um único ciclo, trabalhando nas mesmas características e garantindo que as investigações de usabilidade serão tratadas durante a interação. Segundo Najafi e Toyoshiba (2008) o DCU pode ser incorporado no desenvolvimento empírico sem grandes impactos no processo. No método de desenvolvimento ágil scrum, uma funcionalidade é desenvolvida, testada e documentada em uma *sprint* assim como no DCU. Segundo Santos (2012) o grande desafio do DCU ágil é encontrar a melhor maneira de realizar as atividades de pesquisa do usuário, *design* de versões que atendem às necessidades dos usuário, construção de versões interativas e realização de testes de usabilidade, dentro de um ambiente de desenvolvimento empírico, tendo a participação dos usuários típicos em todas as atividades.

4.4.3 Princípios de Desenvolvimento Empírico Aplicados ao DCU

Alguns princípios do desenvolvimento empírico foram levantados pela comunidade envolvida em usabilidade para a aplicação no DCU.

1. Compreender e identificar as necessidades reais dos usuários

   O Objetivo dessa etapa é conhecer o usuário alvo. Deve-se projetar ferramentas que irão dar suporte aos objetivos e atividades das pessoas.

2. Focar na essência

   O foco na essência ajuda a desenvolver soluções que atendem às reais necessidades dos usuários diminuindo os riscos de desenvolvimento de funcionalidades com pouco ou nenhum uso.

3. Iterar mais rápido

   É importante colocar o produto nas mãos dos usuários para se ter um *feedback* o mais cedo possível. Quando a iteração é feita de forma rápida e começando cedo podemos identificar os problemas com antecedência o que diminui o tempo com retrabalho e evita que o produto não atenda aos usuários.

4. Criar *designs* alternativos

   Deve-se pensar em criar várias soluções para um mesmo problema afim de que possa ter mais possibilidades de escolha.

5. Prototipação em baixa resolução

Os protótipos de baixa resolução permite visualizar uma solução de forma mais rápida e concreta economizando tempo de *design* e desenvolvimento. Esses protótipos ajudam a comunicar e entender ideias e serve para validar uma solução.

6. Menos documentação, mais comunicação

   É essencial que tenha uma boa comunicação em todo o processo entre os desenvolvedores, *designs* e *stakeholders* envolvidos no projeto.

7. Testes de usabilidades ágeis

   Os testes de usabilidade devem ser feitos em todos os ciclos do projeto. Os resultados encontrados nos testes devem ser comunicados à equipe para que possa corrigir possíveis erros.

8. O fim não é o lançamento

   No fim de cada ciclo lança-se o mínimo adequado às necessidades reais dos usuários. Ao observar o uso real identificamos que existem novas formas para usar a ferramenta e podemos propor novas melhorias para o sistema.

A integração entre os processos de usabilidade e os métodos empíricos é esperada e possível. Tanto o desenvolvimento empírico como os processo de usabilidade tem em comum características que colocam o foco do desenvolvimento nas necessidade e anseios dos usuários finais, na interação entre os *stakeholders* envolvidos e na qualidade final do produto a ser desenvolvido.

No proximo capítulo, utilizamos no estudo de caso algumas técnicas de usabilidade que foram aplicadas juntamente com os testes de desenvolvimento empírico de software.

5 Estudo de Caso

Neste capítulo apresentamos o estudo de caso sobre a utilização de práticas de usabilidade e testes automatizados no desenvolvimento do Portal do Software Público, onde será definido um guia de como essas práticas podem ser inseridas nesse contexto.

5.1 Definição

Neste trabalho foi apresentado um estudo teórico relacionado à integração das técnicas de usabilidade ao longo do ciclo de vida ágil de desenvolvimento de testes de softwares existentes.

O objetivo global desse estudo é introduzir técnicas de usabilidade no desenvolvimento empírico a partir das práticas de BDD. Tendo em vista a utilização de várias técnicas de usabilidade e de testes de software em um projeto de desenvolvimento empírico, definimos os objetivos específicos à fim de:

1. Avaliar relação entre práticas de BDD e práticas de usabilidade de um software.
2. Integrar técnicas de usabilidade em um contexto específico de desenvolvimento empírico de software.

Estes objetivos foram definidos levando em consideração que o desenvolvimento empírico tem como prática básica os testes automatizados, e em contrapartida se dá pouca atenção à usabilidade. Assim pensamos em utilizar as práticas de testes que já são constantes para introduzir no ciclo as práticas de usabilidade. Para nos guiarmos em ralação aos objetivos estabelecidos, definimos as seguintes questões:

1. Como inserir os princípios e técnicas de usabilidade dentro do desenvolvimento empírico de software?
2. Como o processo de desenvolvimento utilizando práticas do BDD podem se relacionar com testes de usabilidade?

Para alcançar os objetivos definidos e responder as questões levantadas, definimos algumas questões específicas:

1. Qual a quantidade de testes de aceitação executados por ciclo?

   Fonte: suíte de testes do Cucumber.

Métrica: Quantidade de testes de aceitação executados por ciclo.

2. Quantos problemas de usabilidade de criticidade baixa são encontrados por ciclo?

   Fonte: Planilha de avaliação.

   Métrica: Quantidade de problemas de usabilidade de criticidade baixa por ciclo.

3. Quantos problemas de usabilidade de criticidade média são encontrados por ciclo?

   Fonte: Planilha de avaliação.

   Métrica: Quantidade de problemas de usabilidade de criticidade média por ciclo.

4. Quantos problemas de usabilidade de criticidade alta são encontrados por ciclo?

   Fonte: Planilha de avaliação.

   Métrica: Quantidade de problemas de usabilidade de criticidade alta por ciclo.

5.2 Coleta de Dados

Coletamos os dados entre as sprints 17 e 19 do Projeto do Portal do SPB. Coletamos os dados relacionados ao testes de aceitação a partir dos cenários de testes (presente no apêndice B.3) desenvolvidos e executados utilizando a ferramenta Cucumber. Já os dados relacionados aos problemas de usabilidade, coletamos a partir das avaliações da seçao 4.2.3, realizadas durante as sprints já mencionadas.

5.3 Objeto de Estudo

O Porta do SPB consiste em um sistema que permite o compartilhamento de softwares e faz parte da política de software livre no setor público. O SPB será o objeto de estudo de caso, assim como o processo de colaboração com o SPB, que é realizado no LAPPIS da UnB.

O processo de colaboração com o SPB baseia-se no desenvolvimento empírico. O desenvolvimento de testes automatizados é intrínseco ao processo de desenvolvimento, assim buscamos evoluir a forma com que o processo de colaboração com o SPB lida com problemas de usabilidade. O desenvolvimento é feito a partir de *sprints* de duas semanas, em que são realizadas reuniões com as equipes e reuniões de planejamento (Planning Poker) em cada equipe para definir as atividades de cada *sprint*.

O estudo de caso abrange algumas funcionalidades, que foram desenvolvidas durante o projeto, sendo as seguintes histórias: (1) Cadastro de Usuário, (2) Cadastro de

Instituição e (3) Cadastro de Software, que passaram por ciclos de avaliações de usabilidade.

5.3.1 Ciclo 1 de Avaliações

As avaliações começaram durante a release 2, a partir da sprint 17, seguindo a descrição de avaliação de heurística na seção 4.2.3. Para cada funcionalidade desenvolvida foi determinado um cenário de uso, base para a implementação dos testes de aceitação e consequentemente o seu desenvolvimento. Durante a *release* 1 (sprint 17) foi desenvolvida a história de "Cadastro de Usuário", que possui o seguinte cenário de sucesso:

- **Cenário 01:** Cadastro com sucesso de apenas campos obrigatórios

  [**Dado**] que não existe nenhum usuário com o nome de usuário "josesilva"

  [**Quando**] eu clicar em cadastrar novo usuário

  [**E**] eu preencho os seguintes campos:

  nome de usuário: "josesilva"

  e-mail: "jose@gmail.com"

  senha: "123456"

  confirmação da senha: "123456"

  nome completo: "José da Silva"

  país: "Brasil"

  estado: "Distrito Federal"

  cidade: "Brasília"

  [**E**] eu clico em cadastrar

  [**Então**] eu recebo uma confirmação de cadastro realizado com sucesso

Outra funcionalidade desenvolvida foi a história chamada "Manter Instituição", que possui o seguinte cenário de sucesso:

- **Cenário 01:** Cadastro de nova instituição com sucesso

  [**Dado**] que eu estou na página de cadastro de usuário

  [**Quando**] eu clicar em "Cadastrar nova instituição"

  [**E**] eu preencher os seguintes campos:

  sigla: "MP"

  poder: "executivo"

esfera: “federal”

tipo: “pública”

cnpj: “00.489.828/0002-36”

[**Então**] eu devo visualizar a mensagem “Instituição cadastrada com sucesso!”

Os demais cenários encontram-se na seção apêndice B.3.

Após desenvolvidos, estes cenários apresentaram alguns problemas de usabilidade na avaliação das heurísticas de Nielsen (seção 4.2.3). Estes problemas estão descritos na tabela 5

ID	Local	Descrição do Problema	Heurística Desobedecida	Criticidade
1	Cadastro de Usuário	Linguagens estrangeiras e avisos em outros idiomas	Diálogos simples	Média
2	Cadastro de Usuário	Botão de adicionar nova instituição não é claro para o usuário.	Minimizar a sobrecarga de memória do usuário;	Alta
3	Cadastro de Usuário	Diferença entre botão adicionar nova instituição e criar nova instituição para o usuário	Minimizar a sobrecarga de memória do usuário	Media
4	Cadastro de Instituição	Seleção de País deve ter Brasil como default	Diálogos simples e naturais	Baixa
5	Cadastro de Instituição	Opção de escolha do estado: Random button não posicionado e não funciona no primeiro clique.	Diálogos simples e naturais	Baixa
6	Cadastro de Usuário	Caso você não preencha o email secundário ele informa a seguinte mensagem: E-mail or secondary e-mail already taken	Boas mensagens de erros	Média
7	Cadastro de Usuário	Opção recaptcha só aparece na segunda vez	Consistência	Alta
8	Perfil de Usuário	Ao clicar em hide no bloco de progresso de perfil ele some e não foi encontrada uma opção fácil para reverter a situação.	Prevenção de erros	Baixa
9	Cadastro de Usuário	Caso se tenha uma infinidade de grupos, fica inviável a opção de checkbox. Ou será utilizada apenas para grupos mais “conhecidos”?	Atalhos	Média
10	Cadastro de Usuário	Mensagem de erro um pouco confusa: Usuário com este Usuário já existe.	Consistência	Baixa

Tabela 5: Avaliações de usabilidade - sprint 17

Cada problema levantado nas avaliações gerou uma proposta de melhorias para os cenários envolvidos. Essas melhorias, quando aceitas, geraram mudanças nos cenários de testes de aceitação, consequentemente no desenvolvimento da funcionalidade em si. Os testes de aceitação alterados foram utilizados para verificar os seguintes fatores:

- Capacidade de cadastrar informações de usuário;
- Capacidade de editar informações de usuário;
- Capacidade de cadastrar informações de instituição;
- Capacidade de editar informações de instituição;

5.3.2 Ciclo 2 de Avaliações

O ciclo 2 de avaliações foi realizado na sprint 18. Durante a *release* 2 a história de "Cadastro de Usuário" foi desenvolvida novamente com novo cenário B.4. Para este cenário foi desenvolvido o seguinte protótipo, baseando-se nos requisitos definidos pelos clientes.

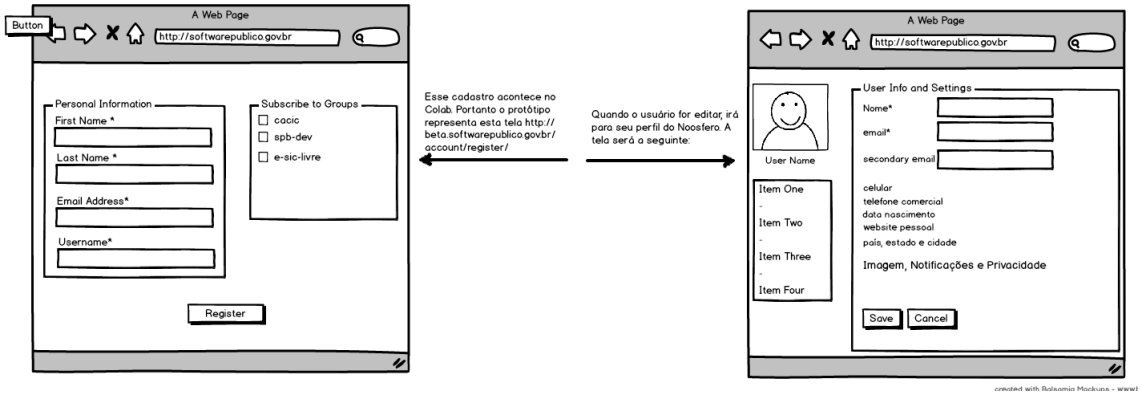


Figura 5: Protótipos de cadastro de usuário

Assim a história de "Novo Software" também foi desenvolvida, com o seguinte cenário de sucesso:

- **Cenário 01:** Novo Software

  [**Dado**] que não existe nenhum software com o localhost "software"

  [**Quando**] eu clicar em "Novo Software"

  [**E**] eu preencho os seguintes campos:

  localhost: "software"

  finalidade: "Finalidade do software"

  licença: "licença"

  [**E**] eu clico em "Salvar"

  [**Então**] eu recebo uma confirmação de cadastro realizado com sucesso, e encontro a pagina de edição de software

Assim como os cenários anteriores, os demais cenários da história de "Novo Software" encontra-se em B.5. Para esta história também foi desenvolvido um protótipo:

Para dar continuidade ao processo avaliamos os cenários da história de "Novo Software" (tabela 6) durante a sprint 18:

Cada problema levantado nas avaliações da sprint 18 gerou uma proposta de melhorias para os cenários envolvidos. Como fizemos as avaliações do segundo ciclo em cima

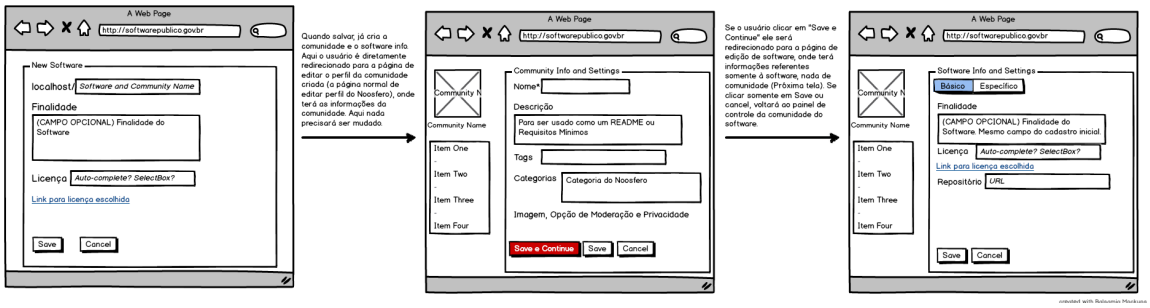


Figura 6: Protótipos de cadastro de software

ID	Local	Descrição do Problema	Heurística Desobedecida	Criticidade
1	Protótipo - Novo Software	Ao escolher um nome para o software, não há evidência do que aconteceria caso o nome fosse igual a outro existente	Prevenção de erros	Média
2	Protótipo - Novo Software	Não há informação para o usuário do que seria o "Link"	Ajuda e documentação	Média
3	Protótipo - Nova Comunidade	Palavras em inglês e português	Linguagem Clara	Baixa
4	Protótipo - Nova Comunidade	Falta de informação sobre as tags e as categorias do noosfero	Ajuda e documentação	Baixa
5	Protótipo - Novo Software	Campos obrigatórios não são definidos	Prevenção de erros	Média
6	Protótipo - Novo Software	Mensagens de ajuda ao preenchimentos dos campos	Prevenção de erros	Baixa
7	Protótipo - Nova Comunidade	Botão de Save e Continue altera estrutura da pagina de software	Consistência	Baixa

Tabela 6: Avaliações de usabilidade - sprint 18

dos protótipos, geramos insumos para o desenvolvimento dos testes de aceitação, que verificam os seguintes fatores:

- Capacidade de cadastrar informações software;
- Capacidade de editar informações de software;
- Capacidade de desativar software;

5.3.3 Ciclo 3 de Avaliações

No ciclo 3, realizado durante a sprint 19, aplicamos checklists (A.5) para auxiliar nas avaliações e levantamos os seguintes problemas em relação as histórias de "Cadastro de Usuário" e "Cadastro de Software", na tabela 7:

Após as melhorias propostas serem desenvolvidas e longo do processo de evolução das histórias mencionadas, os testes de aceitação foram utilizados para verificar os seguintes fatores:

- Capacidade editar informações de usuário;

ID	Local	Descrição do Problema	Heurística Desobedecida	Criticidade
1	Cadastro de Usuário	Rótulos dos campos de registro não contém um elemento de convite à entrada de dados (“ : ”)	Diálogos simples e naturais	Baixa
2	Cadastro de Usuário	Usuário não encontra informações suficientes sobre grupos que pode participar	Diálogos simples e naturais	Média
3	Cadastro de Usuário	Ao se excluir um e-mail secundário, não existe solicitação de confirmação	Feedback	Alta
4	Cadastro de Software	Dificuldade para acessar o botão de criar software	Minimizar sobrecarga	Baixa
5	Cadastro de Software	Funcionalidade de criar software também cria comunidade, porém o usuário não é informado disto	Feedback	Média
6	Cadastro de Software	Dificuldade para acessar o botão de editar software	Minimizar sobrecarga	Baixa
7	Cadastro de Software	Fluxo de edição não está claro (etapas)	Feedback	Média
8	Cadastro de Software	frase “The highlighted fields are mandatory” não especifica o char (*)	Diálogos simples e naturais	Baixa

Tabela 7: Avaliações de usabilidade - sprint 19

- Capacidade de registrar informações de software;
- Capacidade de desativar usuário;
- Capacidade de validar software público;

5.4 Análise e Interpretação dos Dados

Analisamos a evolução de cada história ao longo dos ciclos de avaliações. Cada ciclo de avaliação representa um sprint, em que foram realizadas as avaliações. O ciclo 1 foi realizado na sprint 17, o ciclo 2 na sprint 18 e o ciclo 3 na sprint 19. Uma das histórias avaliadas foi a de “Cadastro de Usuário”:

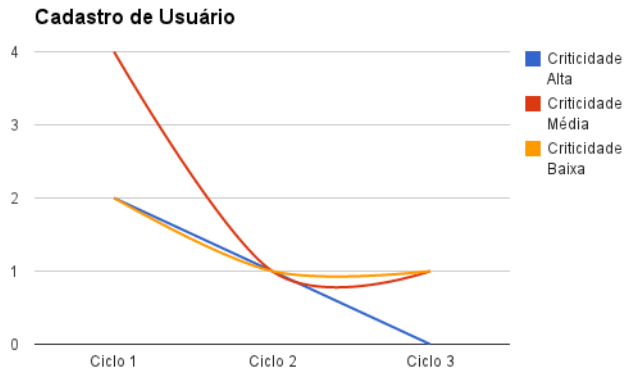


Figura 7: Resultado das avaliações de “Cadastro de Usuário”

Já em “Cadastro de Instituição”, temos:

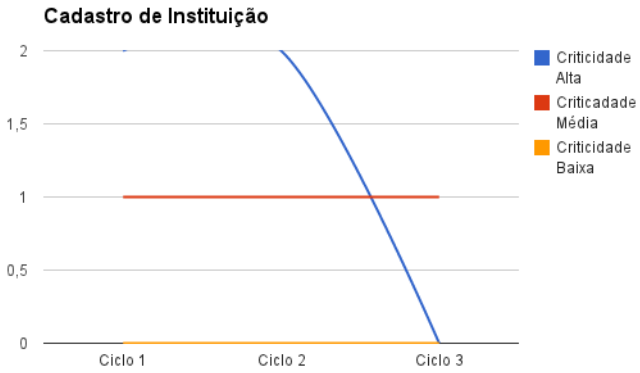


Figura 8: Resultado das avaliações de “Cadastro de Instituição”

Quanto à história de “Cadastro de Software”, temos:

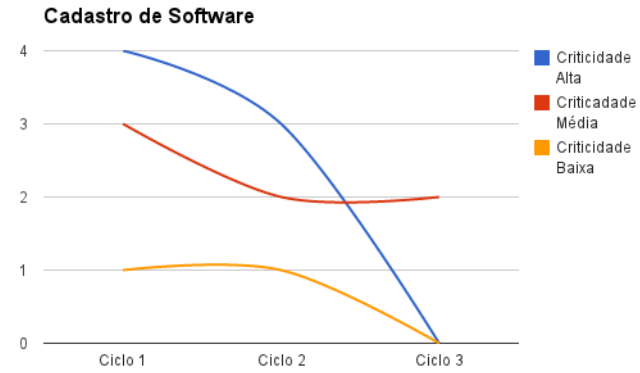


Figura 9: Resultado das avaliações de “Cadastro de Software”

5.4.1 Testes de Aceitação

Quanto aos testes de aceitação dos cenários avaliados tivemos os resultados de medição do primeiro ao último ciclo de avaliações na figura 10 para a história "Cadastro de Usuário":

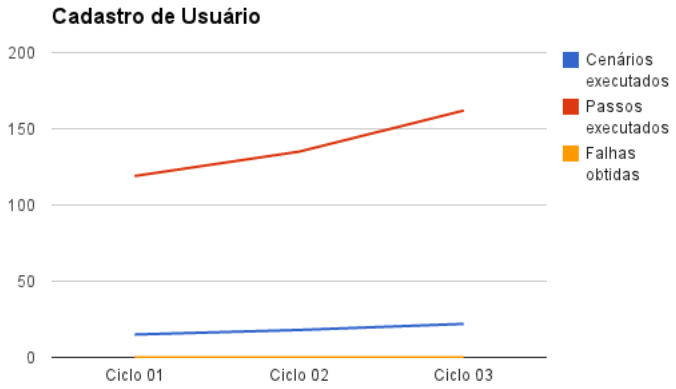


Figura 10: Testes de "Cadastro de Usuário"

Para os testes da história de "Cadastro de Instituição" tesmos a figura 11:

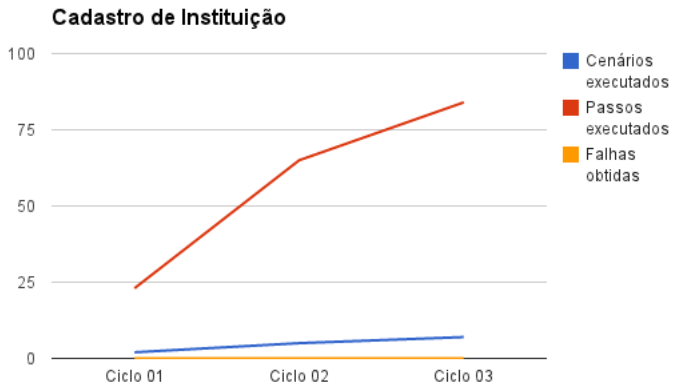


Figura 11: Testes de "Cadastro de Instituição"

Para os testes da história de “Cadastro de Software” temos a figura 12:

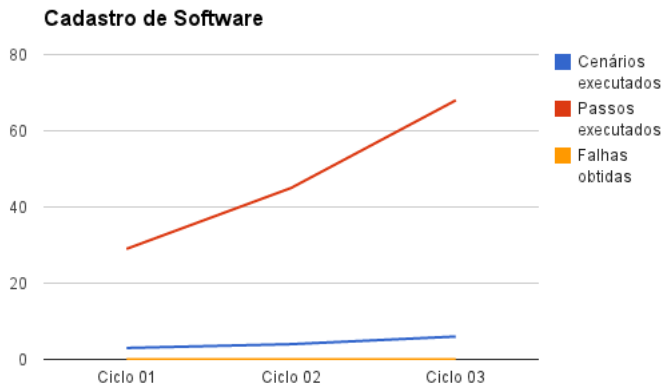


Figura 12: Testes de “Cadastro de Software”

Percebemos assim que houve um aumento na quantidade de testes desenvolvidos.

5.5 Inserção das Práticas de Usabilidade

Durante o estudo de caso inserimos algumas práticas (prototipação, checklist e avaliação de heurísticas) de usabilidade no objeto de estudo, integrando com práticas de métodos empíricos já utilizadas como os testes de aceitação. A prototipação é feita durante o lenvantamento de requisitos com base nos cenários de testes, esses protótipos devem ser submetidos aos checklists A.5, que serão usados como base para as avaliações de heurísticas. As avaliações devem apresentar os problemas de usabilidade, que devem ser utilizadas como insumo para possíveis atividades de melhorias e verificar a implementação das melhorias através dos testes de aceitação, tornando possível o início de um novo ciclo:

1. **Elaboração dos cenários de uso/testes** levantados a partir dos requisitos do sistema;
2. **Criação dos protótipos** de acordo com cenários;
3. **Aplicação dos checklists** A.5 nos protótipos criados;
4. **Avaliação de heurística** nos protótipos criados, com o auxílio dos resultados dos checklists, levantando problemas de usabilidade;
5. **Propor atividades de melhorias** para cada problema levantado durante as avaliações;
6. **Verificação pelos testes de aceitação** de cada atividade de melhoria implementada;

7. **Reinício do ciclo no item 1;**

Estes passos estão ilustrados na figura 13.

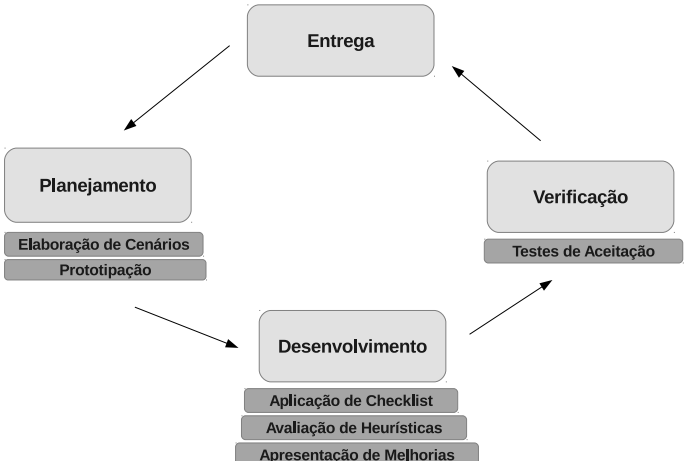


Figura 13: Atividades do estudo de caso

Com este ciclo possibilitamos a inserção de práticas de usabilidade de forma empírica, diminuindo assim a curva de apredizagem da equipe e agregando a importância da usabilidade no processo de desenvolvimento, ficando assim uma grande contribuição para a equipe de desenvolvimento, mesmo com o término do estudo de caso.]

Este capítulo apresentou o estudo de caso sobre o desenvolvimento do Portal do Software Público a partir de métodos empíricos, utilizando práticas de BDD e usabilidade.

6 Conclusão

Durante este trabalho de conclusão de curso, analisamos como inserir práticas de melhorias de usabilidade nos métodos empíricos de desenvolvimento, a partir das práticas de BDD.

Os objetivos deste estudo de caso foram:

1. Integrar usabilidade no ciclo de vida de desenvolvimento empírico de software;
2. Identificar quais técnicas de usabilidade podem ser utilizadas em cada fase do desenvolvimento empírico.
3. Verificar a relação de práticas de BDD e práticas de usabilidade no desenvolvimento empírico de software.

O estudo de caso deste trabalho foi o projeto de desenvolvimento do Portal do Software Público Brasileiro, em que focamos para este trabalho no estudo das práticas de desenvolvimento dos cenários de testes de aceitação, prototipação, avaliação de heurísticas de usabilidade, aplicação de *checklists* de usabilidade; assim como a relação entre essas práticas.

Para atingir dos objetivos do estudo de caso, buscamos responder as seguintes questões:

1. Como inserir os princípios de usabilidade no desenvolvimento empírico de software?
2. Como as práticas do BDD podem se relacionar com as práticas de usabilidade?

A partir do estudo de caso inserimos algumas práticas de usabilidade dentro do processo de desenvolvimento do Portal do SPB, sendo estas práticas: prototipação, avaliação de heurísticas de usabilidade, aplicação de *checklists* de usabilidade. Porém existe uma certa dificuldade em aderir tanto práticas de testes, quanto práticas de usabilidade pela equipe, principalmente durante sprints de final de *release*, quando o prazo de entrega encontra-se mais perto e a equipe tende a se preocupar mais com o desenvolvimento da funcionalidade em si. Quando as práticas de usabilidade e de testes são utilizadas de forma frequente como planejadas verificamos que os testes de aceitação podem ser usados para verificar a aplicação das melhorias estabelecidas através das avaliações de heurísticas de usabilidade.

Entretanto é necessário dar continuidade as práticas de usabilidade durante todo o processo de desenvolvimento, para que a usabilidade possa evoluir juntamente com o

processo de desenvolvimento, assim é as atividades de usabilidade estar sempre à frente das atividades de desenvolvimento, pois necessitam de planejamento de avaliações de usabilidade e da produção de de cenários e protótipos.

De acordo com a análise dos resultados coletados na subseção 5.4 deste estudo, verificamos que a aplicação de práticas de usabilidade como prototipação, avaliação de heurísticas e *checklists* podem ser inseridas no ciclo de desenvolvimento a partir dos cenários dos testes de aceitação. Podemos ver também nas figuras 7, 8 e 9 que o numero de casos com problemas de usabilidade diminuiu a partir das avaliações de heurísticas dos protótipos e cenários de uso diminuiu. Assim o estudo apresentou indícios que o desenvolvimento empírico e suas práticas combinadas com as práticas de usabilidade, pode trazer grandes ganhos em usabilidade.

6.1 Trabalhos Futuros

Encerramos este estudo antes da entrega oficial do Portal do SPB, assim a atividade de testes de usabilidade com usuários ainda será executada, o que é um passo importante para a evolução da usabilidade do Portal e continuação do trabalho realizado neste estudo.

Nas demais sprints do projeto do Portal do SPB, outras práticas de usabilidade podem ser utilizadas pela equipe do projeto. É importante compreender as necessidades reais dos usuários. Um protocolo para a realização de testes de usabilidade estão relatados no apêndice A.3. Esses testes devem ser executados nas próximas com o objetivo avaliar o desempenho do usuário na execução de uma tarefa do sistema. Questionários de satisfação, descritos no apêndice A.2.2 também podem ser utilizados para medir de forma subjetiva o grau de satisfação do usuário.

Como foi aplicado um questionário no início do projeto para analisar o perfil de uso da versão antiga do Portal do SPB, pode-se também ser aplicado um novo questionário para o novo portal. Além disso, iniciamos uma pesquisa para compreender quais práticas de usabilidade os profissionais que trabalham com métodos empíricos utilizam. Um estudo mais detalhado poderá ser feito para uma melhor compreensão do assunto aplicado nesse TCC.

Este trabalho também pode servir de base para um estudo mais aprofundado sobre práticas de BDD e usabilidade, além da integração dessas práticas, também pode-se analisar como uma influencia a outra.

A Apêndice 1

A.1 Processos de Usabilidade

Alguns autores como Mayhew (1999) e Hix e Hartson (1993) propuseram modelos de processo baseados em ciclo de vidas bem definidos para as atividades de usabilidade. O termo modelo de ciclo de vida é utilizado para representar um modelo que capta um conjunto de atividades e a maneira como elas se relacionam (PREECE; ROGERS; SHARP, 2007). Outro modelo muito utilizado para descrever processos de usabilidade é o proposto pela norma ISO/IEC-13407 (1999).

A.1.1 Modelo Estrela

Proposto por Hix e Hartson (1993), o modelo de ciclo de vida estrela derivou do trabalho empírico de entender como os *designers* lidavam com os problemas de *design* em IHC. Eles observaram dois diferentes modelos de trabalho: analítico e o sintético. O primeiro parte-se da visão do sistema para a visão do usuário, já o segundo da visão do usuário para a do sistema (CYBIS; BETIOL; FAUST, 2010).

No modelo estrela não há especificação de ordenamento de atividades. Pode-se ir de uma atividade à outra há qualquer momento, desde que passe primeiramente pela avaliação. Sempre que uma atividade for completada deve-se avaliar o seus resultados.

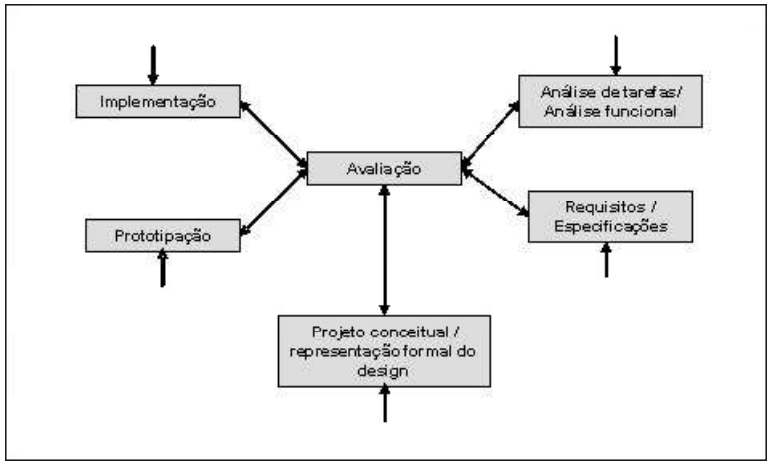


Figura 14: Ciclo de Vida Estrela

A.1.2 O ciclo de vida da Engenharia de Usabilidade - Mayhew

O ciclo proposto por Mayhew (1999) oferece uma visão holística acerca dessa engenharia e uma descrição detalhada de como podemos realizar os testes de usabilidade

A primeira etapa do ciclo é a análise dos requisitos. Mayhew propõe quatro tipos de atividades de análise de requisitos que são detalhadas abaixo:

- **Análise do perfil do usuário:** Os projetistas devem conhecer os atributos pessoais e suas habilidades para cada tipo de usuário identificado.
- **Análise do contexto da tarefa:** Os projetistas devem conhecer os objetivos e resultados, a estrutura, duração, custos e etc. de cada tarefa a ser realizada.
- **Análise das possibilidades e restrições da plataforma:** Verificar quais são as possibilidades e restrições em termos de equipamentos, sistemas operacionais e etc.
- **Análise dos princípios gerais para o projeto:** Atividade relacionada à pesquisa e catalogação dos conhecimentos de ergonomia disponível para a concepção da interface no tipo de contexto de uso.

Depois da análise dos requisitos é preciso especificar as metas de usabilidade do futuro sistema. A norma ISO/IEC 9241-11 (2003) orienta como podemos especificar essas metas.

Na etapa de projetos, testes e implementação, Mayhew propôs que os ciclos devem se repetir de forma a tratar três níveis de aspectos de uma interface: No primeiro nível sendo a interface definida conceitualmente; no segundo nível refere-se as definições em termos de estilo; e no terceiro nível as interações e componentes relacionados com os contextos das tarefas especiais. A última fase é a de instalação do sistema onde depois que o usuário já estiver acostumado com o sistema ele pode dar um *feedback* sobre a usabilidade do produto de forma mais fidedigna por já ser um "especialista" da ferramenta como cita a autora. Uma das técnicas utilizadas para coletar *feedback* são os testes de usabilidade no local do trabalho dos usuários ou utilizando métodos de análise como entrevistas, observações, questionários, grupos de discussões.

A.2 Técnicas e Métodos da Engenharia de Usabilidade

Esta seção mostra as técnicas e os métodos que são utilizados pela interação humano computador para desenvolver um sistema com usabilidade. De acordo com BERVIAN, Cervo e SILVA (2002) as técnicas são procedimentos específicos utilizados por uma ciência determinada. O conjunto com várias técnicas constituem os métodos. Algumas técnicas são utilizadas em várias ciências e os métodos se adapta às diversas ciências a medida que há imposição de uso de técnicas especializadas. Cybis, Betiol e Faust (2010) divide as técnicas e os métodos da engenharia de usabilidade em três tipos: Concepção, Análise e Avaliação.

A.2.1 Métodos e técnicas de concepção

Os métodos de concepção são utilizados para implementar as especificações e os requisitos para a interface a usabilidade de um sistema.

Brainstorming: Bastante utilizado em ambientes ágeis para obter ideias, entrar em consenso sobre problemas ou novas propostas.

Storyboard: É uma representação das interações entre os usuários e o sistema em seu ambiente de trabalho. Corresponde ao detalhamento de um cenário de uso especificado para o sistema consistindo em uma sequência de desenhos representando não só os esboços de telas, mas também os elementos do contexto (usuários, equipamentos, móveis, telefones, colegas).

Card Sorting: É uma técnica usada para descobrir como o usuário classifica determinada informação em sua mente. O usuário recebe uma série de cartões embaralhados descrevendo conteúdos e agrupam os cartões que tenham alguma relação. Podem ser distribuídos cartões com nomes de categorias.

Diagramas de Afinidade: São utilizados para organizar uma grande quantidade de itens em grupos lógicos. Diferente do *card sorting*, nesta técnica os projetistas e usuários trabalham juntos para obter consenso sobre a organização dos itens. Essa técnica pode ser usada para analisar os resultados de estudos de campo e analisar as conclusões de uma avaliação de usabilidade.

Protótipos: Os protótipos de papel são úteis para detectar problemas de usabilidade logo no início do proceso de *design*. São usados para esclarecer os requisitos específicos para o projeto da interface do sistema. São rápidos de construir, permitindo rápidas interações de projeto e necessitam de poucos recursos para serem criados. Existem também os protótipos de baixa, média e alta que simulam o sistema com mais fidelidade do que os protótipos em papel.

A.2.2 Métodos e técnicas de análise

Estes métodos são utilizados para buscar informações sobre o contexto de uso e sobre a usabilidade de um sistema. Podem ser feitas análises do perfil do usuário, o ambiente de uso, as tarefas, possibilidades e restrições do sistema.

Observação: Essa técnica caracteriza-se por um pesquisador observando o usuário e tomando notas, enquanto este trabalha em seu contexto usual. É uma técnica útil para obter dados quantitativos (tempo para as tarefas) e qualitativos (práticas e estratégias do usuário). No planejamento é importante definir os objetivos e as maneiras de

como será registrada os acontecimentos. Deve-se levar em consideração o fato que muitos usuários por estarem sendo observados podem alterar seu comportamento ao utilizar a ferramenta. É importante que todos estejam cientes dos objetivos do estudo, deixando claro que não é ele que será avaliado e sim conhecer uma situação de uso (CYBIS; BETIOL; FAUST, 2010).

Entrevistas Tradicionais: Através de entrevistas podemos obter as opiniões tanto dos usuários atuais como dos futuros usuários dos sistemas. Primeiramente é importante identificar as necessidades das pessoas em acessar uma determinada informação.

Eyetracking: É uma técnica que rastreia o movimento dos olhos e da cabeça para registrar a tomada de informações numa interface.

Questionários de Perfil de Uso: É utilizado para obter informações sobre as características reais dos usuários de um sistema e saber como eles realmente utilizam tais ferramentas. É importante que ao utilizar os questionários de perfil de uso é preciso definir um foco para a sua pesquisa. Deve-se identificar as principais dúvidas da equipe de projeto em relação ao uso do sistema (CYBIS; BETIOL; FAUST, 2010).

Questionários de Satisfação: Questionários de satisfação são utilizados principalmente quando existem usuários experientes que utilizam o sistema com frequência, podendo ter informações fidedigna sobre aspectos satisfatórios e insatisfatórios no sistema. Também podem ser aplicados por usuários de uma nova versão de um sistema imediatamente após um teste de usabilidade. Essa relação com os testes de usabilidade é interessante por permitir a correlação das medidas de desempenho (tempo, frequência) com as medidas de satisfação do usuário (CYBIS; BETIOL; FAUST, 2010).

É recomendado que se utilize um questionário padronizado, pois permite a comparação de resultados obtidos por diferentes sistemas. Estes questionários apresentam opções de respostas fechadas, o que permite a produção de dados quantitativos e objetivos (CYBIS; BETIOL; FAUST, 2010).

Um grande número de questionários foram desenvolvidos pela comunidade científica para a avaliação da usabilidade. Alguns exemplos de questionários são: QUIS, SUMI, WAMMI, SUS, ASQ, PSQ, PSSUQ e CSUQ. Os detalhes referentes a cada questionário estão detalhados no apêndice A.

Grupos de Foco: É uma reunião informal de usuários que manifestam suas opiniões sobre o determinado assunto, que pode ser tanto uma oportunidade de novas funcionalidades ou algum problema específico. Um moderador deve preparar um roteiro com uma lista de assuntos a serem tratados. É importante que os participantes sejam de perfis diferentes para que possa ter uma maior diversidade de informações. Costuma-se ter em média de 6 a 12 usuários em uma mesa de reuniões. Os registros

das informações podem ser através de vídeos ou blocos de anotações. O objetivo não é ter a obtenção do consenso em torno das ideias, mas sim ter uma boa quantidade de opiniões sobre o assunto a ser tratados (CYBIS; BETIOL; FAUST, 2010).

Diários: É uma técnica útil quando a experiência do usuário é ampla e depende da utilização em muitos lugares. Nessa técnica os usuários carregam consigo um pequeno diário para nele anotar as informações do seu dia-a-dia na utilização do sistema (CYBIS; BETIOL; FAUST, 2010).

Benchmarking de Usabilidade: Definido como método sistemático e contínuo de avaliação de sistemas reconhecidos como representantes das melhores e mais eficazes práticas com a finalidade de comparar desempenhos e identificar oportunidades de melhoria (SPENDOLINI, 1994).

Cenários de uso: É uma técnica simples e eficaz para analisar e comunicar uma parte das especificações de requisitos produzidos para a usabilidade e a interface. São utilizados para comunicar os cenários atuais de uma tarefa (problema) e o cenário futuro (solução). Os cenários de solução deve descrever em linguagem natural, como determinados usuários realizarão tarefas específicas com o sistema em um determinado contexto. É preciso decompor os objetivos dos usuários segundo as operações necessárias para alcança-los, identificando as atividades que serão realizadas pelos usuários (CYBIS; BETIOL; FAUST, 2010).

Personas: A técnica de persona descreve o perfil de uma pessoa fictícia envolvida com o produto. Trata-se de inventar um conjunto de pessoas (três ou quatro) que estejam dentro da população de usuários pretendidos e descrevê-las em detalhes. As informações devem ser qualitativas e coletadas por meio de entrevistas e questionários junto à população alvo do sistema. As personas permitem maior entendimento dos usuários, colocando-os no centro das decisões do projeto. Essa técnica tem objetivos similares aos de cenários, porém ao invés do foco ser na tarefa, deve-se ter foco em uma pessoa que faça parte do público alvo do sistema (CYBIS; BETIOL; FAUST, 2010).

A.2.3 Métodos e técnicas de avaliação

Percurso Cognitivo: Percurso cognitivo é um método de inspeção de usabilidade que tem como objetivo avaliar a interface considerando a facilidade da interface. A finalidade do percurso cognitivo é fazer com que o *design* de interação seja fácil de aprender por meio da exploração. Os inspetores aplicam uma lista de verificação orientada à tarefa interativa, abordando os processos cognitivos que se estabelecem quando o usuário a realiza pela primeira vez (CYBIS; BETIOL; FAUST, 2010).

Teste de Usabilidade: É um dos métodos de teste de experiência do usuário (UX) mais frequentemente utilizado e conhecido entre aqueles que não são projetistas da UX. Realizar testes com usuários é o núcleo do *Design* Centrado no Usuário, pois é através destes que podemos saber se as reais expectativas dos usuários são atendidas (SANTOS, 2012).

As técnicas descritas nessa seção podem ser associadas com várias outras técnicas e para a escolha de cada técnica é importante examinar as possibilidades dos recursos necessários e os disponíveis e as expectativas de resultados da avaliação da usabilidade. Cada técnica apresentam qualidades diferentes em relação à quantidade de problemas que as identificam, sistematização dos resultados, à facilidade da aplicação.

A.3 Testes de Usabilidade

Existem vários tipos de testes de usabilidade, mas sabe-se que todos eles têm algo em comum, que é observar as pessoas utilizando algo. Nesta seção definimos de modo genérico como é realizado um teste de usabilidade e também uma definição de um protocolo rápido para realização de testes em um contexto de métodos empíricos.

- **Escolher abordagem**

  As abordagens de pesquisa podem ser de dois tipos: quantitativa ou qualitativas. As pesquisas quantitativas são focadas nos dados numéricos e é voltada para fornecer alta confiança e resultados repetidos dentro de seus grupos de usuários. É preciso ter o envolvimento de um número maior de participantes para contar as variações que você encontrará de indivíduo para indivíduo (UNGER; CHANDLER, 2009). As pesquisas qualitativas não são focadas em níveis de segurança e da possibilidade de repetição, mas sim ganhar contexto e percepção considerando o comportamento do usuário. Ela depende da interpretação do projetista sobre as descobertas, a intuição e o senso comum (UNGER; CHANDLER, 2009).

  Os testes quantitativos são como experimentos científicos que precisam ser rigorosos ou os resultados não serão confiáveis. Deve-se definir um protocolo de teste e segui-lo consistentemente para todos os participantes (KRUG, 2010). Nos testes qualitativos você tenta minimizar a quantidade de interação com o participante para evitar a influência nos resultados (KRUG, 2010).

- **Planejar a pesquisa**

  Algumas questões devem ser respondidas ao criar o teste de usabilidade: Estas questões te ajudam a oferecer foco e escopo. Abaixo algumas perguntas que devem ser respondidas no planejamento de sua pesquisa:

- Defina seu objetivo: Por que você está testando?
- Defina Usuários: Quem você está testando?
- Defina o método para representar sua aplicação: O que você está testando?
- Quais informações você está reunindo?

Nas pesquisas qualitativas geralmente queremos compreender as questões que os usuários podem encontrar, os níveis de frustrações que eles podem experimentar e a gravidade de um problema em particular. Para os testes qualitativos devem se pensar em medidas que serão possíveis de ser respondidas com cinco usuários (UNGER; CHANDLER, 2009).

- Taxa de Sucesso: O grau em que o usuário foi capaz de completar a tarefa.
- Satisfação do usuário

- **Usuários**

  Existem algumas diretrizes que podem ser adotadas para a definição da quantidade de usuários. Nielsen (1994) definiu algumas dessas diretrizes:

  - No teste quantitativo planeje uma quantidade maior de participantes. Em média 20 por rodada de pesquisa.
  - No teste qualitativo é suficiente ter entre 5 e 8 participantes.

  Krug (2010) defende a ideia de que três usuários são ideais para realização de testes de usabilidade para quem está realizando o teste por conta própria. Ele afirma que é mais fácil encontrar três participantes do que uma quantidade maior e que é bem provável que eles já encontrem muitos dos problemas significativos relacionados as tarefas que você está testando.

  Em equipes ágeis, o teste com o usuário é realizado com o cliente do projeto e envolve o especialista em usabilidade como mediador do teste (SANTOS, 2012).

- **Recrutamento e logística**

  Depois de ter feito o plano de pesquisa e definido os tipos de usuários que podem ser inseridos na pesquisa é hora de recrutar os participantes. Para o recrutamento de participantes é interessante gerar uma lista com os potenciais participantes do teste. Essa lista pode vir de usuários registrados do site da empresa relacionada; informações de contatos do cliente; *e-mails* para conhecidos que tenha relação com o assunto do teste; requisições em pequenas pesquisas que pré-qualificam os participantes, e etc. (UNGER; CHANDLER, 2009).

  Você pode realizar uma filtragem com os participantes potenciais antes de selecionalos. As perguntas do questionário de filtragem devem ser voltadas para:

  - Garantir que o participante seja um usuário das funções em que você está testando.
  - Determinar se ele se encaixa em um dos seus grupos de usuários.
  - Ajudar a ter uma boa mistura de participantes.

  O questionário de perfil de usuário pode ser utilizado para realizar essa filtragem de participantes.

- **Criação de guias de discussão**

  Para a execução do teste de usabilidade é preciso que as instruções estejam claras aos participantes, contendo todas as informações específicas que ele precisará para completar as tarefas com sucesso.

  Os seus materiais de teste deve incluir:

  - Formulário de consentimento para gravação de vídeo
  - Guia de discussão para o participante, com tarefas detalhadas e perguntas sobre a satisfação do usuário.
  - Questionários

A.3.1 Protocolo rápido de teste com usuários

Para realização de testes de usabilidade em um contexto de métodos empíricos, definimos um protocolo na qual ao final do desenvolvimento de uma funcionalidade o usuário deverá ser consultado através de um Teste de Usabilidade para coletar o feedback a cerca das funcionalidades desenvolvidas até o momento.

O protocolo para o teste com os usuários seguirá os seguintes passos:

- Elabore um cenário de uso com tarefas à serem executadas.
- O usuário irá testar o sistema de acordo com o cenário especificado.
- Uma pessoa deve observar o que o usuário está fazendo e anotando a sua reação ao utilizar a ferramenta.
- O participante deve verbalizar tudo o que está passando em sua mente ao realizar a atividade: o que estão tentando fazer, o que estão olhando, o que estão lendo ou procurando e que perguntas eles tem em mente.
- Ao final o usuário deve responder algumas perguntas referente as tarefas executadas.

A.4 Questionário de Satisfação

Levantamos informações sobre alguns questionários de satisfação de uso existentes na literatura e foi feito uma comparação entre cada um deles.

A.4.1 QUIS - Questionnaire for User Interaction Satisfaction

O QUIS mede a satisfação do usuário quanto â usabilidade do produto de maneira padronizada, segura e válida, a fim de obter informações precisas em relação à reação dos usuários a novos produtos (QUIS, 2009);

É um questionário proprietário, é sugerido o uso de planilhas eletrônicas e softwares estatísticos até que se implementem recursos de análise no servidor web dos proprietários.

A.4.2 SUS – System Usability Scale

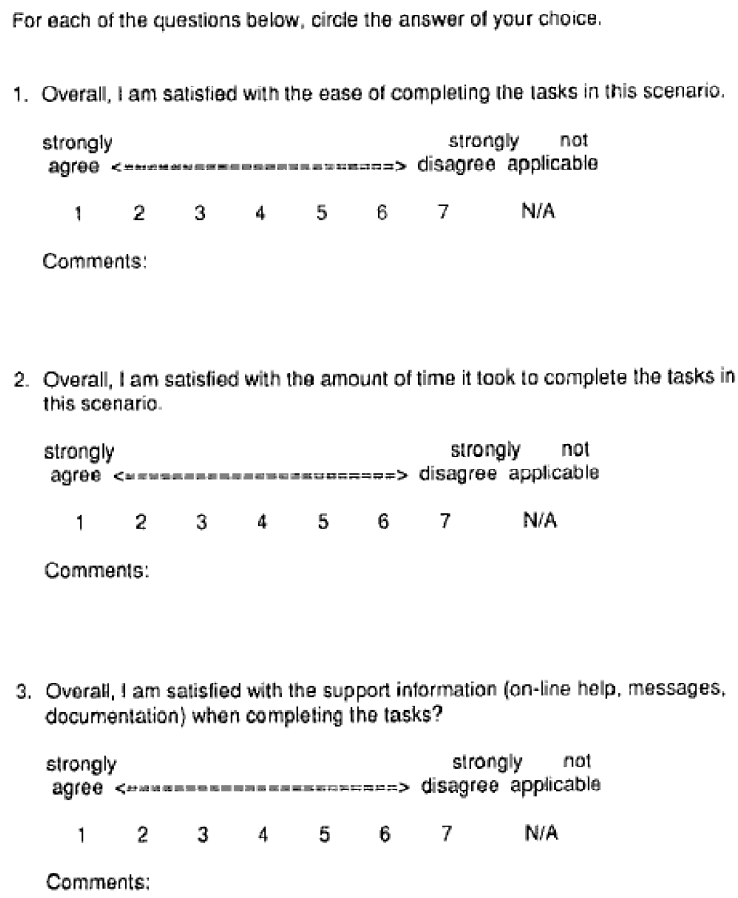
O SUS é uma escala de usabilidade do tipo Likert que possui uma visão global e subjetiva em suas avaliações de usabilidade. Ele apresenta ao entrevistado uma lista de perguntas que devem ser respondidas em uma escala de satisfação (indica o grau de concordância ou discordância do usuário) (BROOKE, 1996).

A.4.3 SUMI – Usability Measurement Inventory

O SUMI proposto por Kirakowski e Corbett (1988) é um questionário para medição da qualidade de um software do ponto de vista do usuário, é um método consistente usado para avaliar a qualidade de uso de um produto de software ou protótipo, e pode ajudar na descoberta de falhas de usabilidade. É mencionado na norma ISO 9241 como um método reconhecido para testar a satisfação do usuário. O SUMI é um questionário comercial.

A.4.4 ASQ – The After-Scenario Questionnaire

O ASQ é um questionário de três itens que são utilizados para avaliar a satisfação do usuário após a conclusão de cada cenário/tarefa. São realizadas umas séries de tarefas que estão de acordo com a realidade do usuário.Este questionário aborda questões como: facilidade de conclusão da tarefa, tempo para completar uma tarefa e adequação das informações de suporte.São questões do tipo Likert, aplicando uma escala de 1 a 7, onde 1 representa “ Concordo totalmente ” e 7 para “ Discordo totalmente ”. (LEWIS, 1995)



For each of the questions below, circle the answer of your choice.

1. Overall, I am satisfied with the ease of completing the tasks in this scenario.

strongly agree <=========> strongly disagree not applicable

1 2 3 4 5 6 7 N/A

Comments:

2. Overall, I am satisfied with the amount of time it took to complete the tasks in this scenario.

strongly agree <=========> strongly disagree not applicable

1 2 3 4 5 6 7 N/A

Comments:

3. Overall, I am satisfied with the support information (on-line help, messages, documentation) when completing the tasks?

strongly agree <=========> strongly disagree not applicable

1 2 3 4 5 6 7 N/A

Comments:

Figura 15: ASQ - The After-Scenario Questionnaire

A.4.5 PSQ – The Printer-Scenario Questionnaire

O PSQ é uma versão inicial do ASQ, mas difere no formato e numero de itens. São escalas de 5 pontos com os termos " Aceitável " com nota 1 e " Precisa de muita Melhoria " com nota 5, e não marcado " Para Avaliar " (LEWIS, 1995).

A.4.6 PSSUQ – The Post-Study System Questionnaire

O PSSUQ fornece uma avaliação global do sistema utilizado. Esse questionário possui 19 itens para avaliação da satisfação do usuário com a usabilidade do sistema. É gasto em média 10 minutos para completar o questionário, mas só é preciso completar uma vez o questionário no fim do estudo de usabilidade. (LEWIS, 1995)

Existem 4 tipos de pontuações para as respostas aos itens do PSSUQ: Escore da satisfação geral (OVERALL), a utilidade do sistema(SYSUSE), a qualidade da informação (INFOQUAL) e a qualidade da interface (INTERQUAL).

A escala Global está relacionada com a soma das classificações ASQ que os participantes deram após completar cada cenário.

	1 2 3 4 5 6 7		
1. No geral, estou satisfeito com o quão fácil é usar o sistema.	Concordo Fortemente	○ ○ ○ ○ ○ ○ ○	Discordo Fortemente
Comentários:			
2. Foi fácil utilizar este sistema.	Concordo Fortemente	○ ○ ○ ○ ○ ○ ○	Discordo Fortemente
Comentários:			
3. Eu pude completar as tarefas e cenários de forma **efetiva**, usando este sistema.	Concordo Fortemente	○ ○ ○ ○ ○ ○ ○	Discordo Fortemente
Comentários:			
4. Eu fui capaz de completar as tarefas e cenários de forma **rápida**, usando este sistema.	Concordo Fortemente	○ ○ ○ ○ ○ ○ ○	Discordo Fortemente
Comentários:			
5. Eu fui capaz de completar as tarefas e cenários de forma **eficiente**, usando este sistema.	Concordo Fortemente	○ ○ ○ ○ ○ ○ ○	Discordo Fortemente
Comentários:			
6. Eu me senti confortável usando este sistema.	Concordo Fortemente	○ ○ ○ ○ ○ ○ ○	Discordo Fortemente
Comentários:			
7. Foi fácil aprender a usar este sistema.	Concordo Fortemente	○ ○ ○ ○ ○ ○ ○	Discordo Fortemente
Comentários:			

Figura 16: PSSUQ – The Post-Study System Questionnaire

8. Eu acredito que eu poderia me tornar produtivo rapidamente usando este sistema.	Concordo Fortemente	○ ○ ○ ○ ○ ○ ○	Discordo Fortemente
Comentários:			
9. As mensagens de erros do sistema foram claras o suficiente para me ajudar na correção erros.	Concordo Fortemente	○ ○ ○ ○ ○ ○ ○	Discordo Fortemente
Comentários:			
10. Sempre que eu cometi algum erro, eu pude recuperar de forma fácil e rápida.	Concordo Fortemente	○ ○ ○ ○ ○ ○ ○	Discordo Fortemente
Comentários:			
11. As informações (como ajuda on-line, na tela de mensagens e outros documentos) fornecidas com este sistema foram claras.	Concordo Fortemente	○ ○ ○ ○ ○ ○ ○	Discordo Fortemente
Comentários:			
12. Foi fácil encontrar a informação que eu precisava.	Concordo Fortemente	○ ○ ○ ○ ○ ○ ○	Discordo Fortemente
Comentários:			
13. A informação fornecida pelo sistema é fácil de entender.	Concordo Fortemente	○ ○ ○ ○ ○ ○ ○	Discordo Fortemente
Comentários:			
14. A informação foi eficaz em me ajudar a completar as tarefas e cenários.	Concordo Fortemente	○ ○ ○ ○ ○ ○ ○	Discordo Fortemente
Comentários:			
15. A organização das informações nas telas do sistema é clara.	Concordo Fortemente	○ ○ ○ ○ ○ ○ ○	Discordo Fortemente
Comentários:			
16. A interface deste sistema é agradável.	Concordo Fortemente	○ ○ ○ ○ ○ ○ ○	Discordo Fortemente

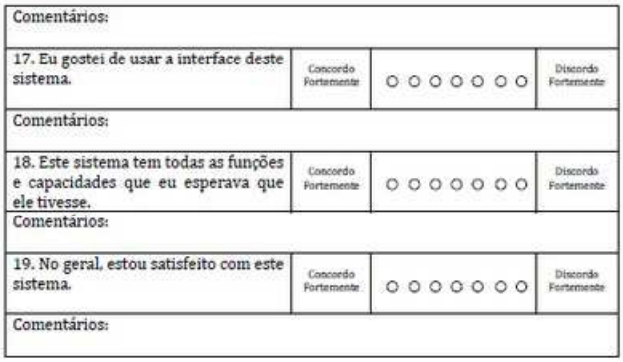


Figura 17: PSSUQ: The Post-Study System Questionnaire - Questões de 8 à 19

A.4.7 CSUQ - Computer System Usability Questionnaire

Este questionário é parecido com o PSSUQ, mas a sua redação e diferente. Enquanto no PSSUQ afirma que " Eu poderia efetivamente realizar as tarefas e cenários usando este sistema " o CSUQ escreve: " Eu posso terminar meu trabalho de forma eficaz usando esse sistema?". Na IBM, este questionário foi aplicado através de e-mail, enviado para funcionários de diferentes locais, o que houve uma maior quantidade de participantes, do que feito com grupos reduzidos presencialmente. (LEWIS, 1995)

A.4.8 Comparativo dos questionários

Os questionários ASQ e PSQ são utilizados após a realização de um cenário. Contém os mesmos itens, mas possuem escalas diferentes. O ASQ possui uma maior confiabilidade em relação ao PSQ.

PSSUQ e CSUQ são ambos os questionários de satisfação global. O PSSUQ utiliza itens adequados para uma situação de teste de usabilidade, já o CSUQ são apropriados para uma situação de teste de campo. Os questionários possuem propriedades psicométricas aceitáveis de usabilidade e podem ser usados com confiança como medidas padronizadas de satisfação. É interessante utilizar o PSSUQ junto com o ASQ.

Nome	**Criador**	**Questões**	**Licença**	**Int. Avaliada**	**Confiab.**	**Escala**
ASQ	IBM	3	Aberto	Qualquer	0,93	Discordo Fortemente / Concordo Fortemente
CSUQ	IBM	19	Aberto	Baseado em computador	0,95	Discordo Fortemente / Concordo Fortemente
PSSUQ	IBM	19	Aberto	Baseado em computador	0,96	Discordo Fortemente / Concordo Fortemente
SUMI	HERG	27	Proprietário	Software	0,89	Discordo Fortemente / Concordo Fortemente
SUS	DEC	10	Aberto	Qualquer	0,85	Discordo Fortemente / Concordo Fortemente
QUIS	UMD	50	Proprietário	-	-	0 a 9

Tabela 8: Comparativo dos questionários

A.5 *Checklist* de Usabilidade

Este checklist contém questões especializada em aspectos ou critérios que determinam a ergonomia de uma interface homem-computador.

1. Prestreza

Guia o usuário poupando-o do aprendizado de uma série de comandos

1.1. Os títulos de telas, janelas e caixas de diálogo estão no alto, centrados ou justificados à esquerda?

- (•) Sim
- ( ) Não
- ( ) Não se Aplica

1.2. Caso o dado a entrar possua um formato particular, esse formato encontra-se descrito na tela?

- ( ) Sim
- ( ) Não
- ( ) Não se aplica

1.3. Os rótulos dos campos contêm um elemento específico, por exemplo " : ", como convite às entradas de dados?

- ( ) Sim
- ( ) Não
- ( ) Não se aplica

1.4. O usuário encontra disponíveis as informações necessárias para suas ações?

- ( ) Sim
- ( ) Não
- ( ) Não se aplica

Figura 18: Checklist de usabilidade- Questões da seção 01

2. Agrupamento por localização

A compreensão de uma tela pelo usuário depende da ordenação dos objetos que são apresentados.

2.1. A disposição dos objetos de interação de uma caixa de dialogo segue uma ordem lógica?

- ( ) Sim
- ( ) Não
- ( ) Não se aplica

Figura 19: Checklist de usabilidade- Questões da seção 02

3. Agrupamento por formato

Será mais fácil para o usuário perceber relacionamento(s) entre itens ou classes de itens, se diferentes formatos ou diferentes códigos ilustrarem suas similaridades ou diferenças.

3.1. Os diferentes tipos de elementos de uma tela de consulta (dados, comandos e instruções) sã visualmente distintos uns dos outros?

- ○ Sim
- ○ Não
- ○ Não se aplica

3.2. Os cabeçalhos de uma tabela estão diferenciados através do emprego de cores diferentes, letras maiores ou sublinhadas?

- ○ Sim
- ○ Não
- ○ Não se aplica

3.3. Na apresentação de textos, os recursos de estilo, como itálico, negrito, sublinhado ou diferentes fontes são empregados para salientar palavras ou noções importantes?

- ○ Sim
- ○ Não
- ○ Não se aplica

Figura 20: Checklist de usabilidade- Questões da seção 03

3.4. Os campos obrigatórios são diferenciados dos campos opcionais de forma visualmente clara?

- ○ Sim
- ○ Não
- ○ Não se aplica

3.5. Nas caixas de mensagens, o botão selecionado por default tem uma apresentação visual suficientemente distinta dos outros?

- ○ Sim
- ○ Não
- ○ Não se aplica

Figura 21: Checklist de usabilidade- Questões da seção 03

4. Feedback

A qualidade e a rapidez do feedback são dois fatores importantes para o estabelecimento de satisfação e confiança do usuário, assim como para o entendimento do diálogo.

4.1. O sistema fornece feedback para todas as ações do usuário?

- ○ Sim
- ○ Não
- ○ Não se aplica

4.2. Quando, durante a entrada de dados, o sistema torna-se indisponível ao usuário, devido a algum processamento longo, o usuário é avisado desse estado do sistema e do tempo dessa indisponibilidade?

- ○ Sim
- ○ Não
- ○ Não se aplica

4.3. Os itens selecionados de uma lista são realçados visualmente de imediato?

- ○ Sim
- ○ Não
- ○ Não se aplica

Figura 22: Checklist de usabilidade- Questões da seção 04

5. Legibilidade

A performance melhora quando a apresentação da informação leva em conta as características cognitivas e perceptivas dos usuários. Uma boa legibilidade facilita a leitura da informação apresentada.

5.1. As áreas livres são usadas para separar grupos lógicos em vez de tê-los todos de um só lado da tela, caixa ou janela?

- ○ Sim
- ○ Não
- ○ Não se aplica

5.2. Os rótulos de campos organizados verticalmente e muito diferentes em tamanho estão justificados à direita?

- ○ Sim
- ○ Não
- ○ Não se aplica

5.3. Os ícones são legíveis?

- ○ Sim
- ○ Não
- ○ Não se aplica

Figura 23: Checklist de usabilidade- Questões da seção 05

6. Concisão

Quanto menos entradas, menor a probabilidade de cometer erros.

6.1. O sistema oferece valores defaults para acelerar a entrada de dados?

- ○ Sim
- ○ Não
- ○ Não se aplica

7. Ações Mínimas

Quanto mais numerosas e complexas forem as ações necessárias para se chegar a uma meta, a carga de trabalho aumentará e a probabilidade de ocorrência de erros.

7.1. Os grupos de botões de comando possuem sempre um botão definido como default?

- ○ Sim
- ○ Não
- ○ Não se aplica

Figura 24: Checklist de usabilidade- Questões das seções 06 e 07

8. Controle do usuário

O controle sobre as interações favorece a aprendizagem e, assim, diminui a probabilidade de erros.

8.1. O usuário pode terminar um diálogo seqüencial repetitivo a qualquer instante?

- ○ Sim
- ○ Não
- ○ Não se aplica

8.2. O usuário pode interromper e retomar um diálogo seqüencial a qualquer instante?

- ○ Sim
- ○ Não
- ○ Não se aplica

Figura 25: Checklist de usabilidade- Questões da seção 08

9. Proteção contra erros

É preferível detectar os erros no momento da digitação, do que no momento da validação. Isto pode evitar perturbações na planificação da tarefa.

9.1. O sistema apresenta uma separação adequada entre áreas selecionáveis de um painel de menu de modo a minimizar as ativações acidentais?

- ○ Sim
- ○ Não
- ○ Não se aplica

9.2. Em toda ação destrutiva, os botões selecionados por default realizam a anulação dessa ação?

- ○ Sim
- ○ Não
- ○ Não se aplica

9.3. O sistema solicita confirmação (dupla) de ações que podem gerar perdas de dados e/ou resultados catastróficos?

- ○ Sim
- ○ Não
- ○ Não se aplica

Figura 26: Checklist de usabilidade- Questões da seção 09

10. Mensagens de erro

A qualidade das mensagens favorece o aprendizado do sistema, indicando ao usuário a razão ou a natureza do erro cometido, o que ele fez de errado, o que ele deveria ter feito e o que ele deve fazer.

10.1. As mensagens de erro ajudam a resolver o problema do usuário, fornecendo com precisão o local e a causa específica ou provável do erro, bem como as ações que o usuário poderia realizar para corrigí-lo?

- ○ Sim
- ○ Não
- ○ Não se aplica

10.2. As frases das mensagens de erro são curtas e construídas a partir de palavras curtas, significativas e de uso comum?

- ○ Sim
- ○ Não
- ○ Não se aplica

Figura 27: Checklist de usabilidade- Questões da seção 10

11. Correção de erros

Os erros são bem menos perturbadores quando eles são fáceis de corrigir.

11.1. Qualquer ação do usuário pode ser revertida através da opção DESFAZER?

- ○ Sim
- ○ Não
- ○ Não se aplica

Figura 28: Checklist de usabilidade- Questões da seção 11

12. Consistência

Os procedimentos e rótulos são melhor reconhecidos quando seu formato são estáveis de uma tela para outra. Nessas condições, o sistema é mais previsível e a aprendizagem mais generalizável.

12.1. A organização em termos da localização das várias características das janelas é mantida consistente de uma tela para outra?

- ○ Sim
- ○ Não
- ○ Não se aplica

12.2. A localização dos dados é mantida consistente de uma tela para outra?

- ○ Sim
- ○ Não
- ○ Não se aplica

12.3. Os formatos de apresentação dos dados são mantidos consistentes de uma tela para outra?

- ○ Sim
- ○ Não
- ○ Não se aplica

12.4. Os rótulos estão na mesma posição em relação aos campos associados?

- ○ Sim
- ○ Não
- ○ Não se aplica

Figura 29: Checklist de usabilidade- Questões da seção 12

B Apêndice 2

B.1 Descrição de testes de aceitação

Essa estrutura é proposta da seguinte forma:

1. **Título:** Palavra-chave '*feature*' e um título curto que representa o objetivo da *feature*.

2. **Narrativa:** um texto curto que demonstre os cenários de execução, exatamente como a narrativa de uma história de usuário:

```
Feature: send work and emails to environment
  As an suer
  I want to send work with notification email
```

Figura 30: Descrição do título (*feature*) de um teste

3. **Pré condições:** representado pela palavra-chave *'background'*, define os passos precedem cada cenário de teste.

```
Background:
  Given "Work Assignment" plugin is enabled
  Given the following users
    | login     | name          |
    | joaosilva | Joao da Silva |
    | josesilva | Jose da Silva |
```

Figura 31: Descrição de pré condições (*background*) de um teste

4. **Cenários:** representam parte concreta de como o software deve se comportar, e sendo a parte essencial do teste realizado no cucumber. Após a palavra-chave *'scenario'* define-se o nome do cenário em questão:

5. **Passos:** Cada cenário possui uma série de passos que demonstram o seu comportamento, que são linhas simples iniciadas com as seguintes palavras-chaves: *Given, When, Then, And, But.*

6. **Given:** Indica uma condição inicial para que o cenário seja executado, trata-se das pré-condições do cenário.

7. **When:** Indica o evento do cenário

8. **Then:** Indica o que é esperado após o evento ocorrer.

```
@selenium
Scenario: Acess work assignment page
  Given I am logged in as "joaosilva"
  When I follow "My Community"
  And I go to mycommunity's control panel
  And I follow "Manage Content"
  And I follow "New content"
  And I follow "Work Assignment"
  Then I should be on /myprofile/mycommunity/cms/new
```

Figura 32: Descrição do cenário (*scenario*) de um teste

B.2 Descrição de funcionais e unitários

Os testes funcionais e unitários são escritos da seguinte forma:

Setup: indica as condições inciais dos testes, setando variáveis de ambiente e de configuração por exemplo.

```
def setup
  @controller = WorkAssignmentPluginCmsController.new
  @request    = ActionController::TestRequest.new
  @response   = ActionController::TestResponse.new

  @profile = create_user_with_permission('test_user', 'post_content')
  @person = @profile.user.person
  login_as :test_user
  @user = @profile.user
end
```

Figura 33: Descrição do setup de um teste

Título: título do teste iniciado com a palavra *'should'* e finalizado com *'do'*

Passos: código que define o comportamento do teste

Verificação: Assertiva que verifica se a ação foi realizada como esperada.

```
should 'verify plugins name' do
  plugin = WorkAssignmentPlugin.new
  value = "Work Assignment"
  assert_equal value, plugin.plugin_name
end
```

Figura 34: Código de teste

B.3 Cenários de Uso da release 1

Os cenários de uso foram a base de criação dos testes de aceitação. Durante a primeira release (release 1) foram desenvolvidas algumas histórias, dentre estas, a história de "Cadastro de Usuário", que possui os seguintes cenários de sucesso:

- **Cenário 01:** Cadastro com sucesso de apenas campos obrigatórios

[**Dado**] que não existe nenhum usuário com o nome de usuário “josesilva”

[**Quando**] eu clicar em cadastrar novo usuário

[**E**] eu preencho os seguintes campos:

nome de usuário: “josesilva”

e-mail: “jose@gmail.com”

senha: “123456”

confirmação da senha: “123456”

nome completo: “José da Silva”

país: “Brasil”

estado: “Distrito Federal”

cidade: “Brasília”

[**E**] eu clico em cadastrar

[**Então**] eu recebo uma confirmação de cadastro realizado com sucesso

- **Cenário 02:** Cadastro com sucesso de apenas campos obrigatórios de usuário governamental

[**Dado**] que não existe nenhum usuário com o nome de usuário “josesilva”

[**Quando**] eu clicar em cadastrar novo usuário

[**E**] eu preencho os seguintes campos:

nome de usuário: “josesilva”

e-mail: “jose@serpro.gov.br”

e-mail secundário: “jose@gmail.com”

senha: “123456”

confirmação da senha: “123456”

nome completo: “José da Silva”

cargo: “analista de TI”

país: “Brasil”

estado: “Distrito Federal”

cidade: “Brasília”

[**E**] eu seleciono “SERPRO” como instituição

[**E**] eu seleciono “teste” como unidade

[**E**] eu clico em cadastrar

[**Então**]eu recebo uma confirmação de cadastro realizado com sucesso

- **Cenário 3:** Cadastro com sucesso com todos os campos preenchidos, mesmo não obrigatórios

  [**Dado**] que não existe um usuário cujo email primário ou email secundário é "maria@gmail.com"

  [**Quando**] eu clicar em cadastrar novo usuário

  [**E**] eu preencho os seguintes campos:

  nome de usuário: "mariasilva"

  e-mail: "maria@gmail.com"

  e-mail secundário: "maria@yahoo.com"

  senha: "123456"

  confirmação da senha: "123456"

  nome completo: "Maria da Silva"

  cargo: "analista de TI"

  áreas de interesse: "Engenharia de Software"

  país: "Brasil"

  estado: "Distrito Federal"

  cidade: "Brasília"

  [**E**] eu seleciono "Outro" como instituição

  [**E**] eu clico em cadastrar

  [**Então**] eu recebo uma notificação de cadastro realizado com sucesso.

Os cenários de falha ocorrem nas seguintes situações:

- Email proposto existir como email de outro usuário;
- Email secundário proposto existir como email de outro usuário;
- Email secundário ser um email governamental e ao email primário não ser um email governamental;
- Não preenchimento de campos obrigatórios para usuário governamental

Quanto a história chamada "Manter Instituição" possui os seguintes cenários:

- **Cenário 01:** Cadastro de nova instituição com sucesso

  [**Dado**] que eu estou na página de cadastro de usuário

[**E**] que a seguinte instituição não existe:

nome: “Ministério do Planejamento, Orçamento e Gestão”

sigla: “MP”

poder: “executivo”

esfera: “federal”

tipo: “pública”

cnpj: “00.489.828/0002-36”

[**Quando**] eu clicar em “Cadastrar nova instituição”

[**E**] eu preencher os seguintes campos:

sigla: “MP”

poder: “executivo”

esfera: “federal”

tipo: “pública”

cnpj: “00.489.828/0002-36”

[**Então**] eu devo visualizar a mensagem “Instituição cadastrada com sucesso!”

- **Cenário 02:** Busca de instituição inexistente

[**Dado**] que eu estou na página de cadastro de usuário

[**E**] que a seguinte instituição não existe:

nome: “Ministério do Planejamento, Orçamento e Gestão”

sigla: “MP”

poder: “executivo”

esfera: “federal”

tipo: “pública”

cnpj: “00.489.828/0002-36”

[**Quando**] eu buscar MP

[**Então**] eu devo visualizar a mensagem “Instituição não cadastrada”

[**E**]eu devo visualizar a opção de cadastrar nova instituição

B.4 Cenários de Uso da release 1

Durante a release 2 a história de “Cadastro de Usuário” foi desenvolvida novamente com o seguinte cenário:

- **Cenário 01:** Cadastro com sucesso de apenas campos obrigatórios

  [**Dado**] que não existe nenhum usuário com o nome de usuário “josesilva”

  [**Quando**] eu clicar em “Cadastre-se”

  [**E**] eu preencho os seguintes campos:

  primeiro nome: “José”

  ultimo nome: “Silva”

  endereço de e-mail: “jose@gmail.com”

  usuário: “josesilva”

  [**E**] eu clico em “Cadastre-se”

  [**Então**] eu recebo uma confirmação de cadastro realizado com sucesso, com a seguinte mensagem: “Você deve se logar para seu perfil. Perfis não validados serão deletados em 24h.”

B.5 Cenários de uso da release 2

Durante a release 2 a história de “Novo Software” foi desenvolvida com os seguintes cenários:

- **Cenário 01:** Novo Software

  [**Dado**] que não existe nenhum software com o localhost “software”

  [**Quando**] eu clicar em “Novo Software”

  [**E**] eu preencho os seguintes campos:

  localhost: “software”

  finalidade: “Finalidade do software”

  licenca: “licença”

  [**E**] eu clico em “Salvar”

  [**Então**] eu recebo uma confirmação de cadastro realizado com sucesso, e encontro a pagina de edição de software

  Outros cenários de edição de software são: “Informações de Comunidade” e “Informações de Software”.

- **Cenário 02:** Informações de Comunidade

  [**Dado**] que “software” está cadastrado

  [**Quando**] eu clicar em “Cadastre-se”

[**E**] eu preencho os seguintes campos:

descricao: “Descrição do software”

tags: “software”

categorias: “categoria1”

[**E**] eu clico em “Salvar”

[**Então**] eu recebo uma confirmação de cadastro salvo com sucesso

- **Cenário 03:** Informações de Software

[**Dado**] que “software” está cadastrado e estou em Edição de software

[**Quando**] eu clicar em “Especifico”

[**E**] eu preencho os seguintes campos:

Sigla: “teste”

sistema operacional: “teste os”

funcionalidades: “testes”

categorias: “categoria1”

[**E**] eu clico em “Nova Biblioteca”

[**E**] eu preencho os seguintes campos:

nome: “teste”

versão: “teste”

licença: “teste”

[**E**] eu clico em “Novo Sistema Operacional”

[**E**] eu preencho os seguintes campos:

nome: “Debian”

versão: “teste”

[**E**] eu clico em “Nova linguagem”

[**E**] eu preencho os seguintes campos:

nome: “C++”

versão: “teste”

sistema operacional: “Debian”

[**E**] eu clico em “Novo Banco de Dados”

[**E**] eu preencho os seguintes campos:

nome: “apache”

versão: “teste”

sistema operacional: “Debian”

[**E**] eu clico em “Salvar”

[**Então**] eu recebo uma confirmação de cadastro salvo com sucesso

C Apêndice 3

C.1 Pesquisa com profissionais de software livre e métodos ágeis

Foi realizada uma pesquisa para entender o que profissionais que trabalham com software livre e métodos ágeis entendem sobre as técnicas de usabilidade. O resultado se encontram abaixo:

A grande maioria dos profissionais pesquisados eram desenvolvedores, seguidos de designs, arquitetos de informação, especialistas de usabilidade e gerentes de projetos. Alguns deles realizam mais de um papel em sua equipe de trabalho.

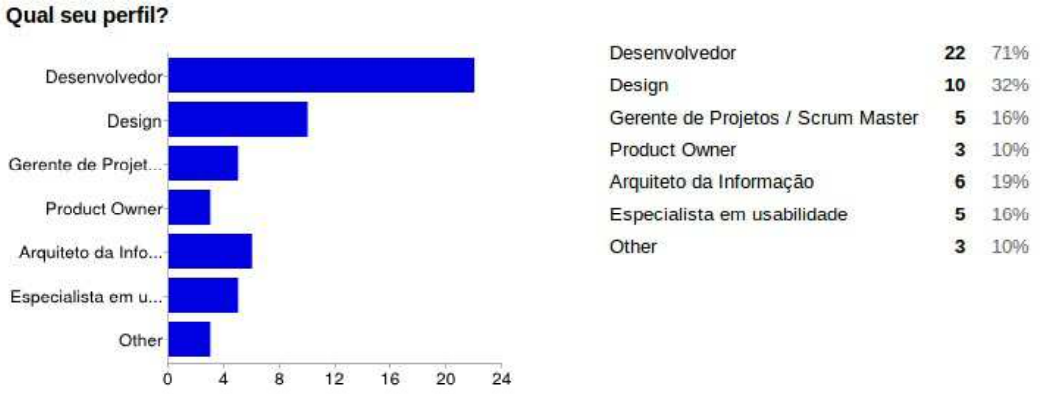


Figura 35: Perfil dos entrevistados

Como a pesquisa foi realizada com profissionais diversos, alguns não estavam envolvidos com software livre e com métodos ágeis simultâneamente. Dentre os pesquisados a grande maioria, 48% tinham entre 1 e 2 anos de experiência com métodos ágeis. Em relação à software livre, 19% entre 1 e 2 anos, 19% de 5 a 10 anos e 19% não trabalhavam com software livre.

Para 61% dos entrevistados, investir em usabilidade deve ser feito em todo o ciclo de desenvolvimento e realizada não somente pelos especialistas de usabilidade, mas também pelos desenvolvedores e designs.

Figura 36: Tempo de trabalho

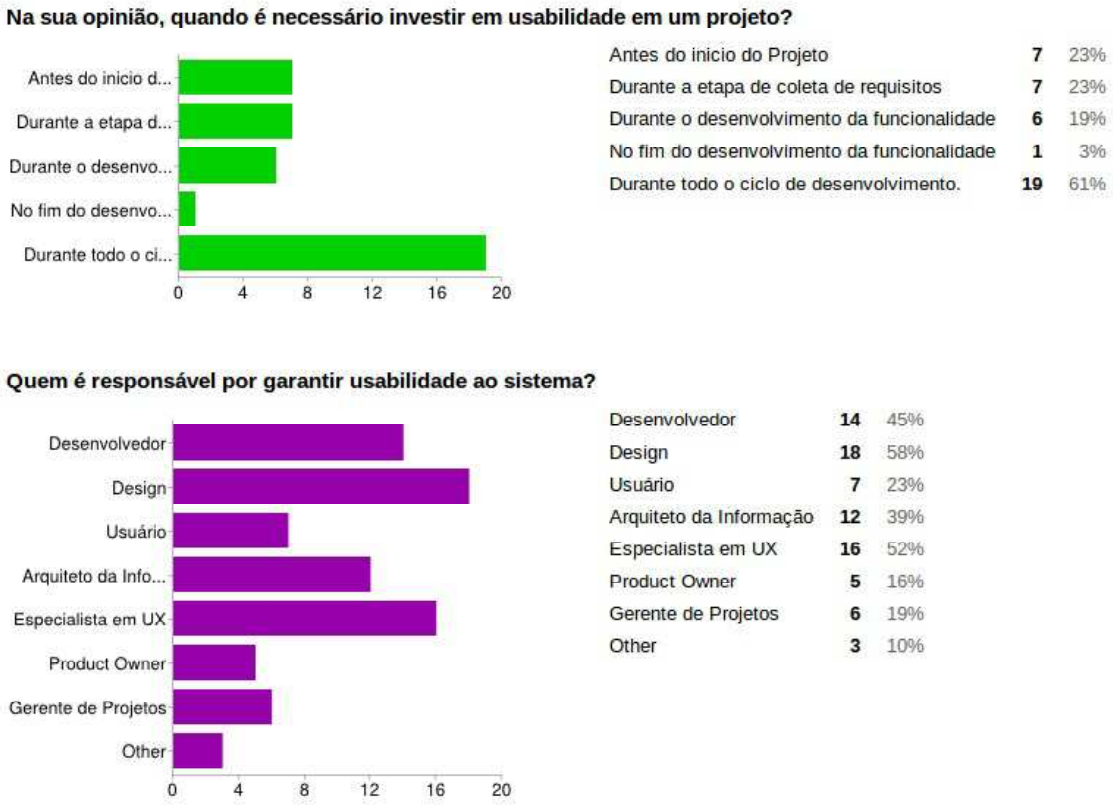


Figura 37: Quando e quem é responsável por garantir a usabilidade

Os resultados abaixo mostram as técnicas utilizadas pelos profissionais para avaliar, análisar e conceber interfaces com usabilidade.

Uma das técnicas mais utilizadas para análisar o contexto de uso do sistema são

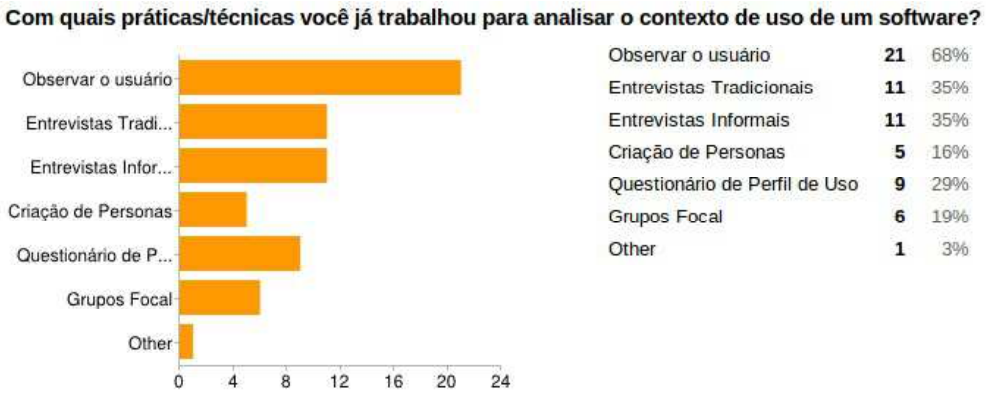


Figura 38: Técnicas de contexto de uso

as observações de usuários, com 68%, as entrevistas estão em seguidas com 35% e os questionários de perfil de uso com 29%.

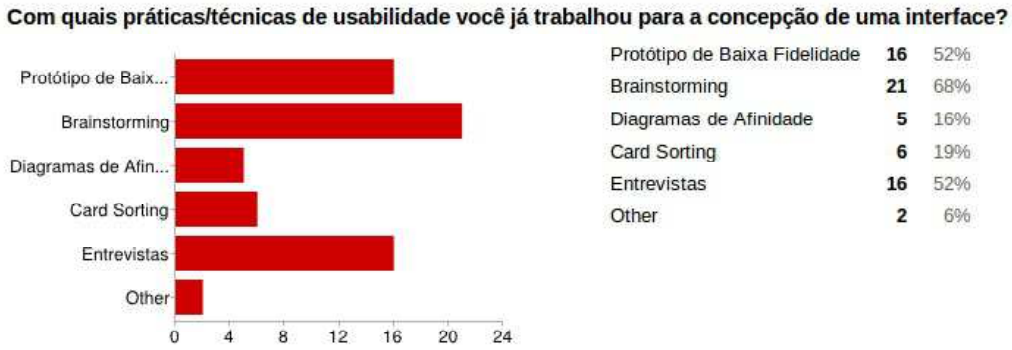


Figura 39: Técnicas de concepção de interfaces

A técnica de Braistorming é bastante utilizada por 68% dos pesquisados para concepção de novas interfaces. Os protótipos de baixa fidelidade são bastante utilizados pelos pesquisados. 52% também realizam entrevistas com usuários quando necessitam criar uma nova interface.

Figura 40: Técnicas de avaliação utilizadas

Os testes com usuários é uma das técnicas mais utilizadas pelos profissionais com 58%, seguido pelas técnicas de avaliação heurísticas e checklists (listas de verificação).

Os resultados abaixo mostram qual a importância de cada técnica de concepção de uma interface de acordo com a percepção dos pesquisados.

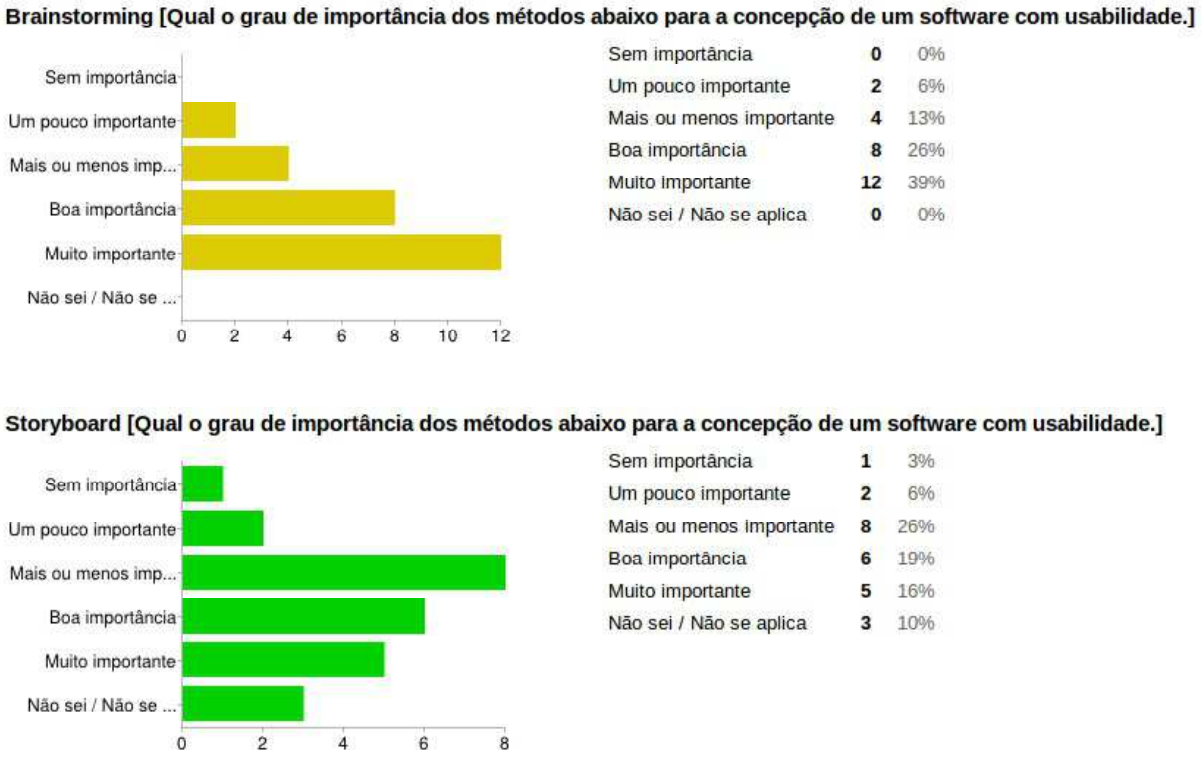


Figura 41: Técnicas de concepção - Brainstorming e Storyboard

Em relação ao Brainstorming, 39% dos entrevistados concordam que a técnica é muito importante, já em relação ao storyboard a maioria, 26% informa que é mais ou menos importante.

Em relação as técnicas de Card Sorting e Diagramas de afinidade, 26 por cento dos pesquisados não sabiam o que eram cada técnica.

As técnicas de prototipação seja de baixa fidelidade ou geral obtiveram uma boa aceitação por parte dos pesquisados.

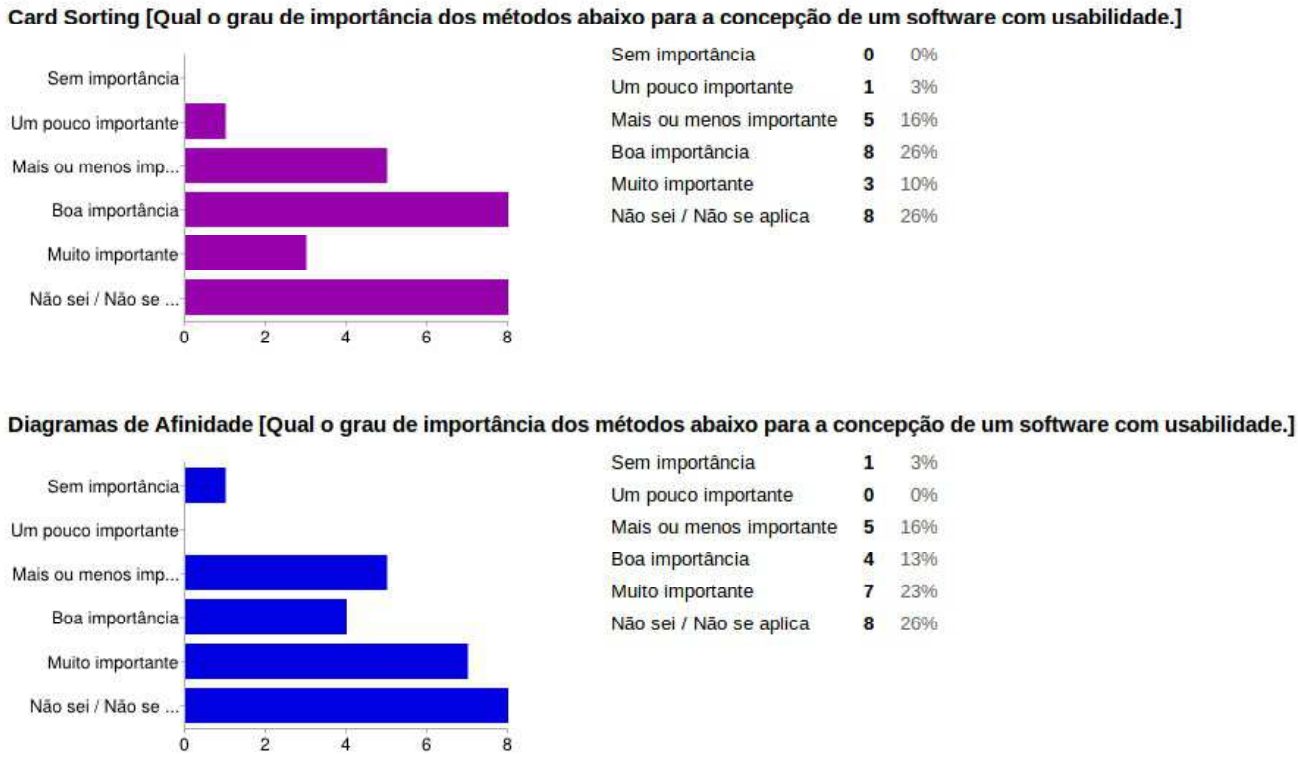


Figura 42: Técnicas de concepção - Card Sorting e Diagramas de Afinidade

Os resultados abaixo mostram a importância das técnicas de análise.

As técnica de observar o usuário obteve uma grande aceitação por parte dos pesquisados, sendo 65% tendo muita importância.

As entrevistas tanto tradicionais, como as informais obtiveram resultados próximos, mais de 50% informaram que têm boa ou muita importância.

32% dos pesquisados informaram que a técnica de Eytracking é muito importante e 16% não sabiam ou não aplicavam tal técnica.

Os questionários de perfil de uso obtiveram boa importância, 32%, mas alguns pesquisados, 6% informaram que não são muito importante para realização de análises no contexto que estão inseridos.

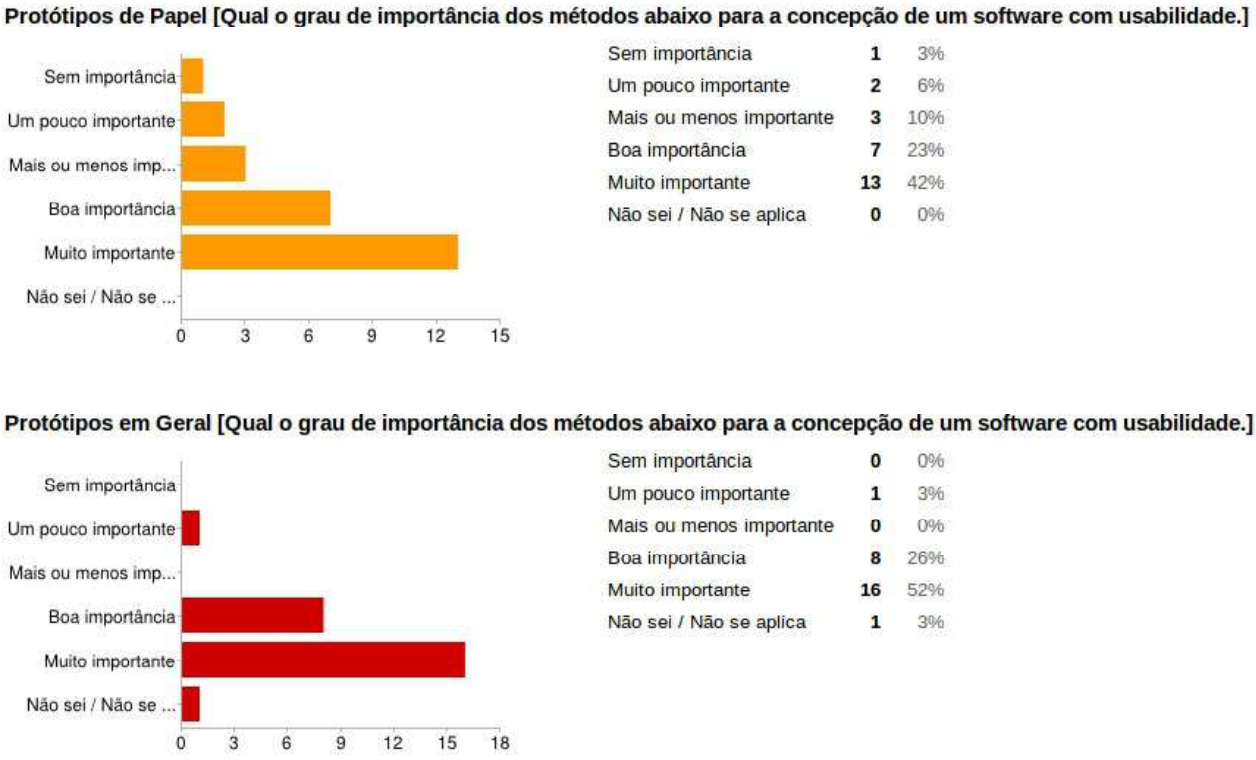


Figura 43: Técnicas de concepção - Protótipos

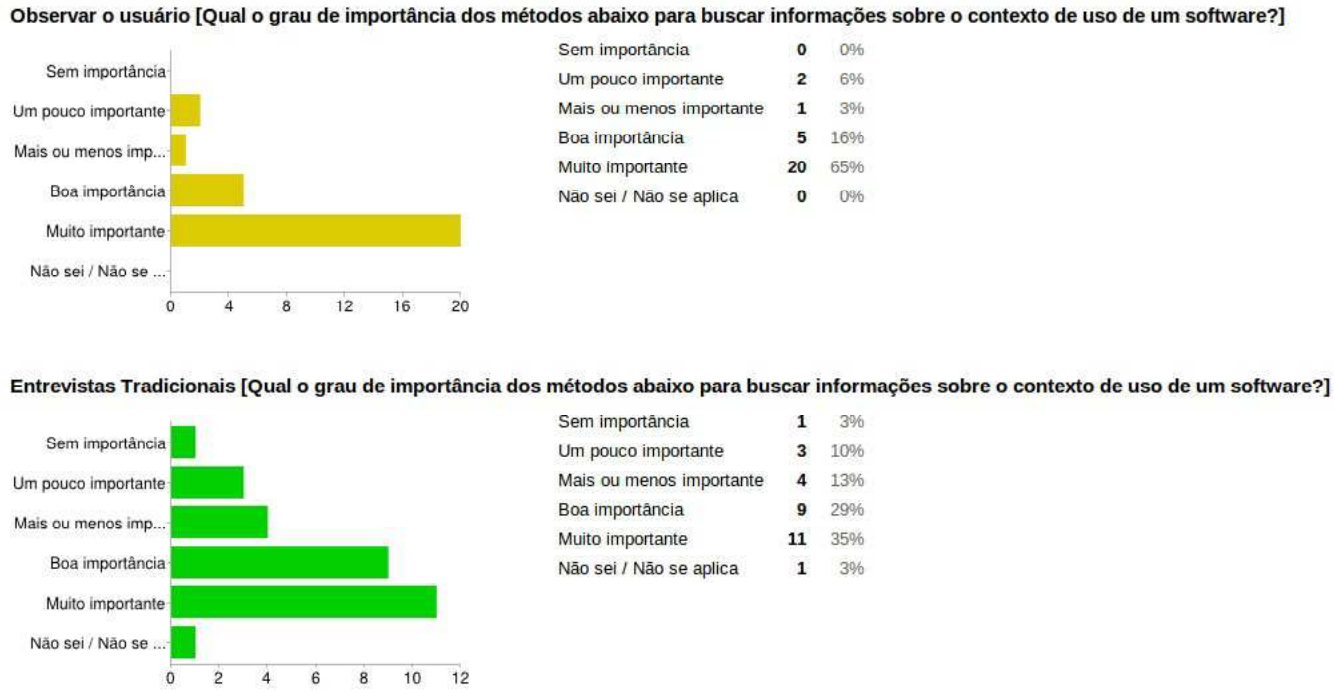


Figura 44: Técnicas de análise - Observação e entrevistas tradicionais

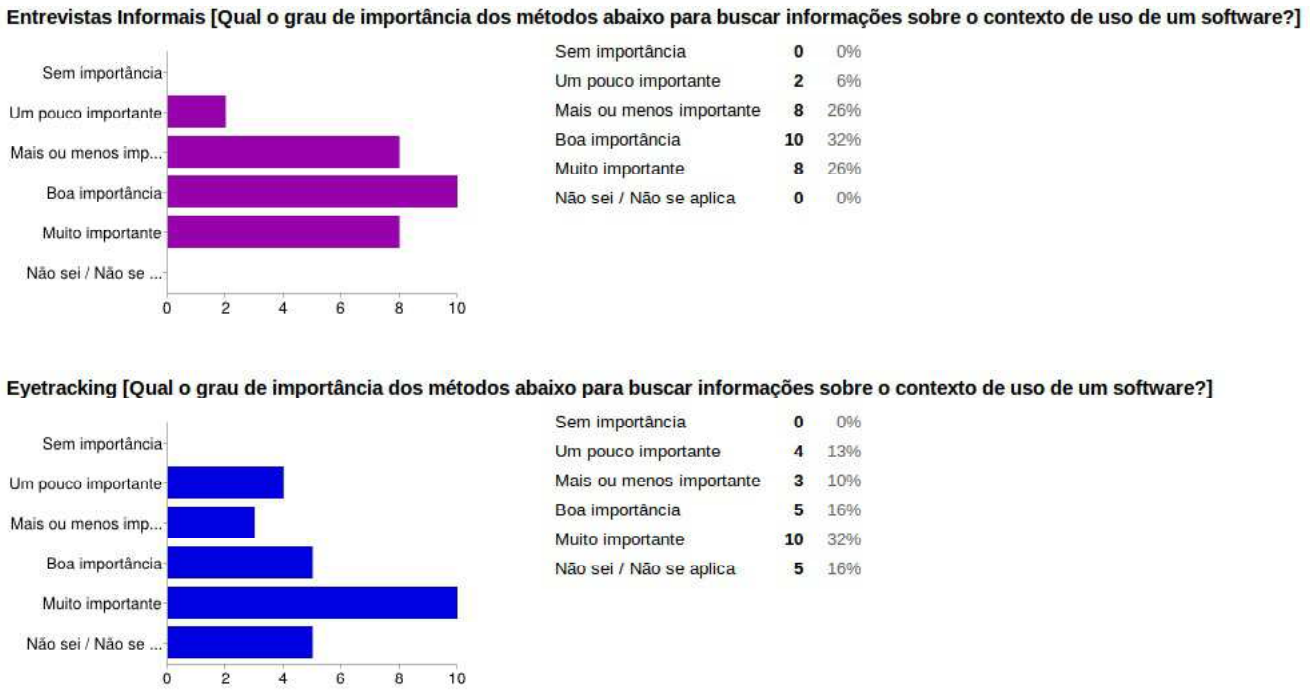


Figura 45: Técnicas de análise - Entrevistas Informais e Eytracking

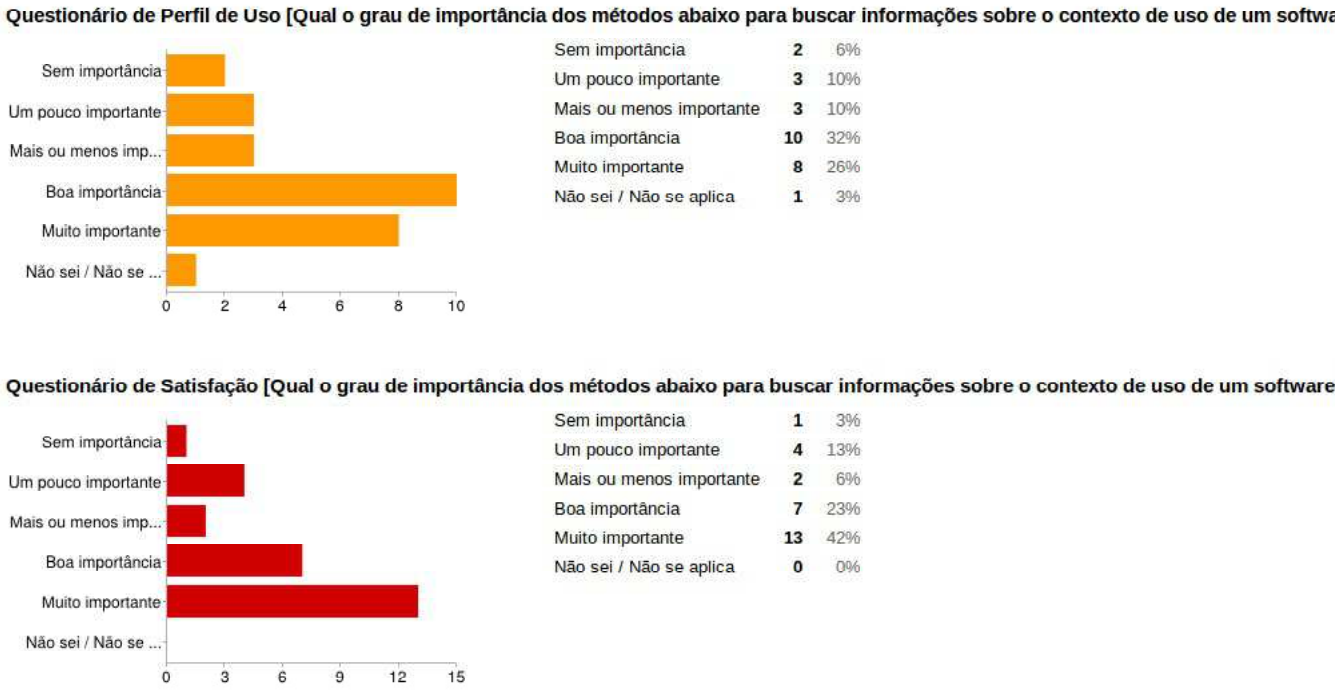


Figura 46: Técnicas de análise 'Questionários de perfil de uso e de satisfação

Os resultados abaixo são referentes as técnicas de avaliação da usabilidade.

Mais de 60% informaram que as avaliações feitas por heurísticas e listas de verifi-

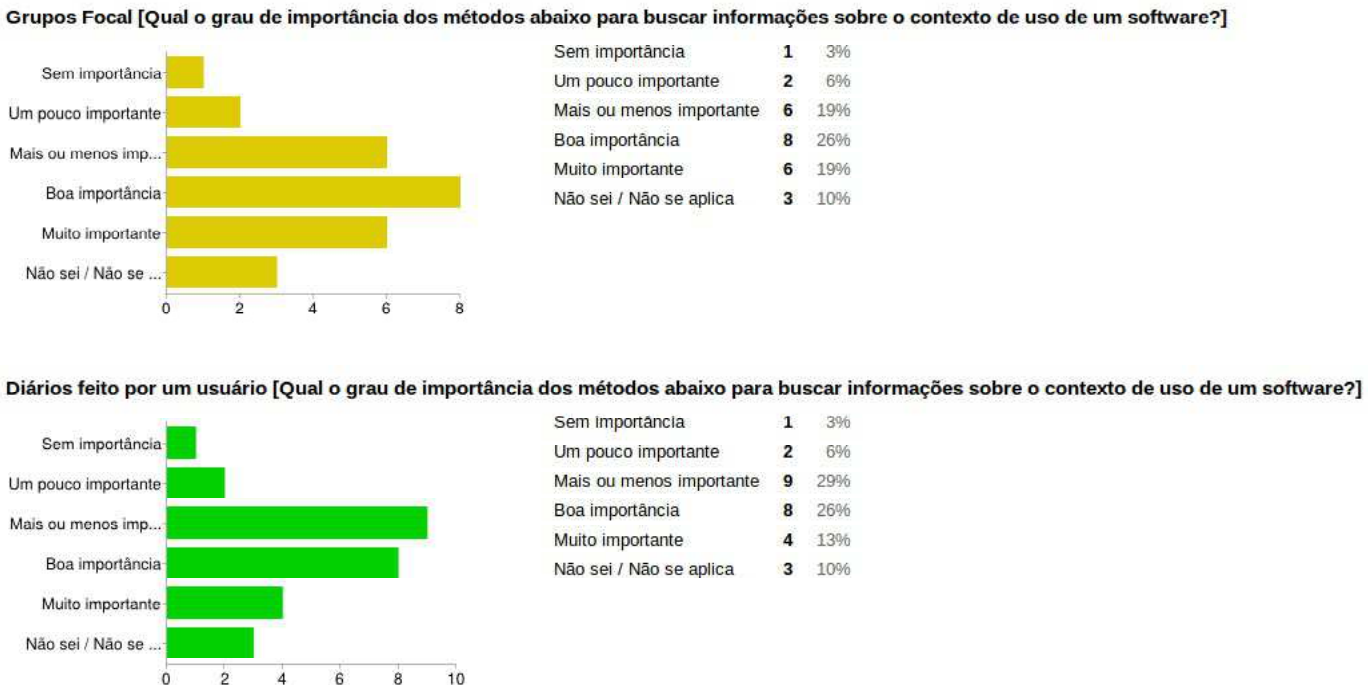


Figura 47: Técnicas de análise - Grupo Focal e Diários

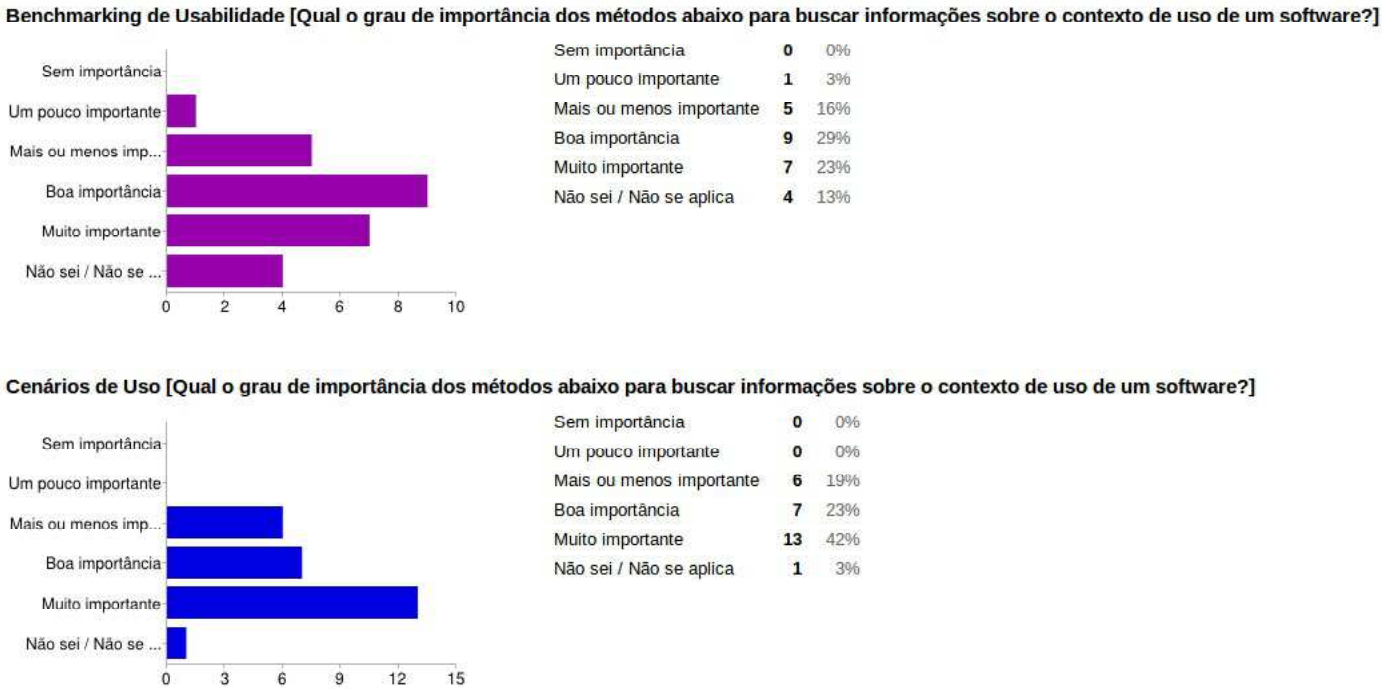


Figura 48: Técnicas de análise - Benchmarking e Cenários de uso

cação são de boa ou muita importância.

Os Testes de Usabilidade foram o que mais apresentaram importância para os

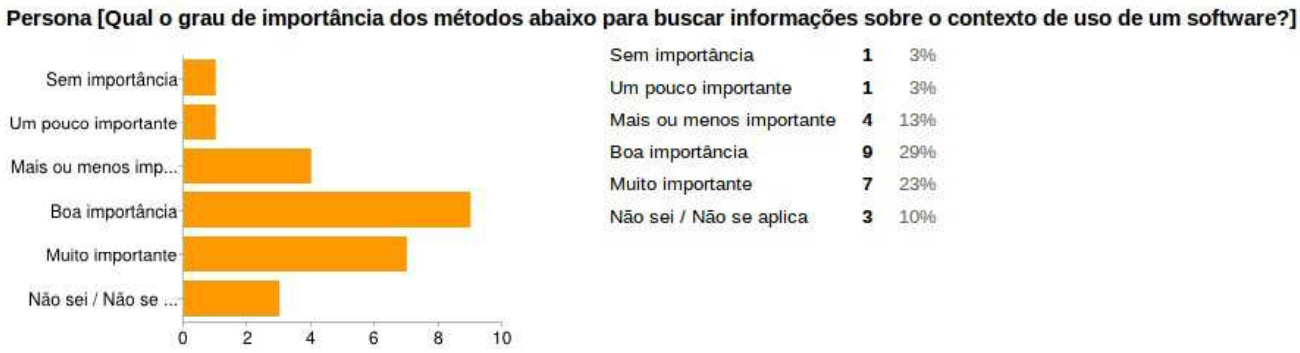


Figura 49: Técnicas de análise - Persona

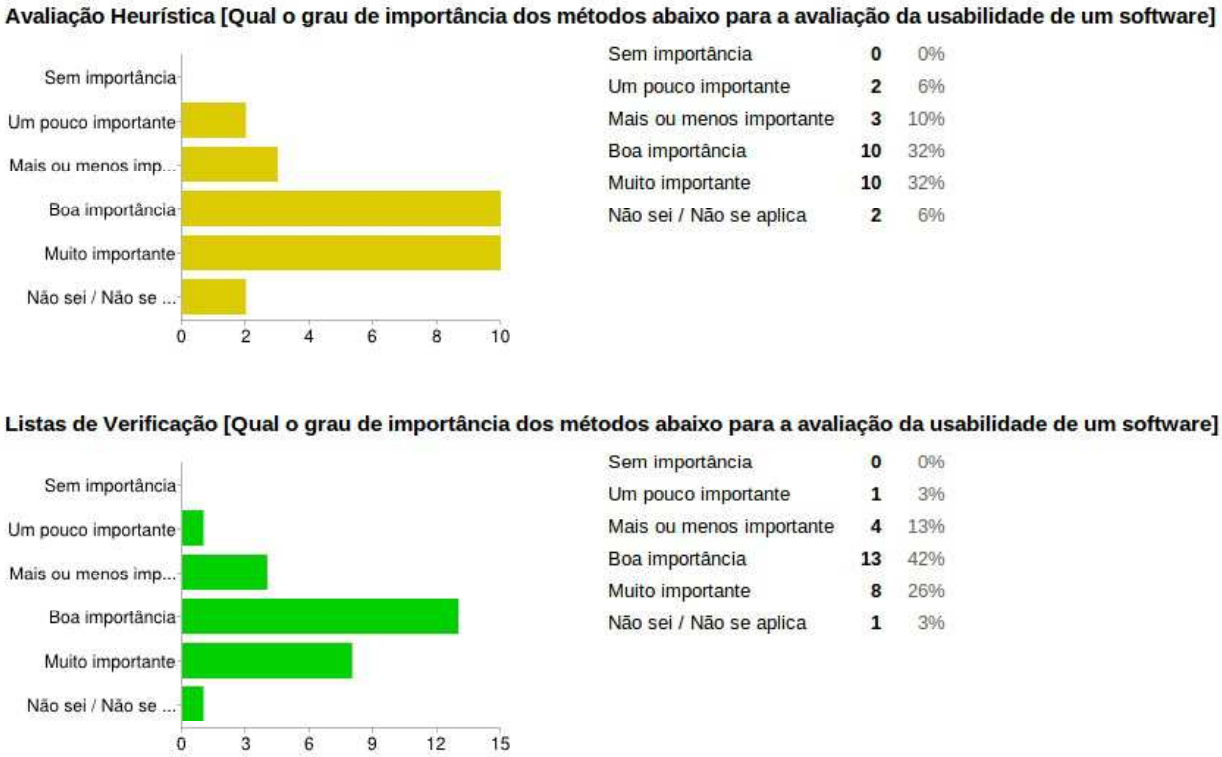


Figura 50: Técnicas de avaliacao - Heurísticas e Listas de Verificação

pesquisados, sendo 77% muito importante e 10% boa importância.

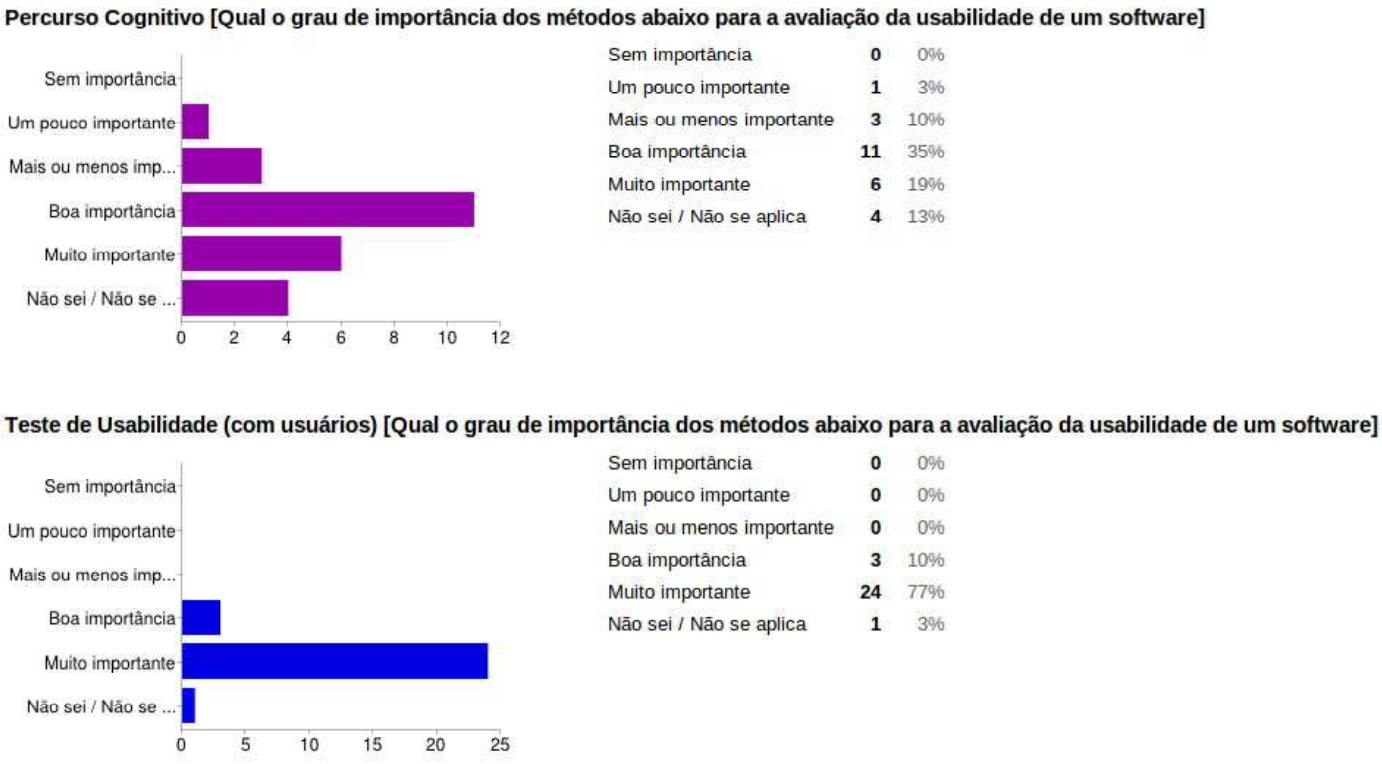


Figura 51: Técnicas de avaliação - Percursos Cognitivos e Testes de Usabilidade

Referências

BARBOSA, D. F.; FURTADO, E. S.; GOMES, A. S. Uma estratégia de apoio à institucionalização da usabilidade em ambientes de desenvolvimento ágil. In: SOCIEDADE BRASILEIRA DE COMPUTAçãO. *Proceedings of the VIII Brazilian Symposium on Human Factors in Computing Systems*. [S.l.], 2008. p. 214–223. Citado na página 38.

BECK, K. *Test-Driven Development by Example*. [S.l.]: Addison-Wesley Prefessional, 2002. Citado 3 vezes nas páginas 11, 28 e 29.

BERNARDO, P. C. *Padrões de testes automatizados*. Dissertação (Mestrado) — Instituto de Matemática e Estatística – Universidade de São Paulo, 2011. Disponível em: <http://www.teses.usp.br/teses/disponiveis/45/45134/tde-02042012-120707/>. Citado 2 vezes nas páginas 27 e 29.

BERVIAN, P. A.; CERVO, A. L.; SILVA, R. d. Metodologia científica. *São Paulo*, 2002. Citado na página 60.

BROOKE, J. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, London: Taylor and Francis, v. 189, p. 194, 1996. Citado na página 67.

CHAUí, M. *Convite a filosofia*. [S.l.: s.n.], 2003. Citado na página 23.

CHELIMSKKY, D. et al. *The RSpeec Book: Behaviour-Driven Development with RSpec, Cucumber, and Friends*. [S.l.: s.n.], 2010. Citado 2 vezes nas páginas 27 e 30.

CONSTANTINE, L. L.; LOCKWOOD, L. Process agility and software usability: Toward lightweight usage-centered design. *Information Age*, v. 8, n. 8, p. 1–10, 2002. Citado na página 39.

CORBUCCI, H. *Métodos ágeis e software livre: um estudo da relação entre estas duas comunidades*. Tese (Doutorado) — Universidade de São Paulo, 2011. Citado na página 24.

CYBIS, W.; BETIOL, H.; FAUST, R. *Ergonomia e Usabilidade: Conhecimentos, Métodos e Aplicações*. [S.l.: s.n.], 2010. Citado 8 vezes nas páginas 33, 34, 35, 38, 59, 60, 62 e 63.

DIAS, C. *Usabilidade na Web: Criando websites mais acessíveis*. [S.l.: s.n.], 2006. Citado na página 31.

EASON, K. D. User-centred design: for users or by users? 2005. Citado na página 37.

EVERETT, G. D. et al. *Software Testing*. [S.l.: s.n.], 2007. Citado na página 27.

GARRET, J. *The Elements of user experience*. [S.l.: s.n.], 2003. Citado na página 32.

GUIMARAES, C. P.; CHAVES, C. v. F. G. Xplus: Integrando o design de interfaces centrado na experiência do usuário ao processo de desenvolvimento de software com extreme programming. Citado 2 vezes nas páginas 11 e 39.

HARING, R. Behavior driven development: Beter dan test driven development. Java Magazine, 2011. Citado na página 29.

HIX, D.; HARTSON, H. R. *Developing User Interfaces: Ensuring Usability Through Product &Amp; Process*. New York, NY, USA: John Wiley & Sons, Inc., 1993. ISBN 0-471-57813-4. Citado 2 vezes nas páginas 37 e 59.

HODGETTS, P. Experiences integrating sophisticated user experience design practices into agile processes. In: IEEE. *Agile Conference, 2005. Proceedings*. [S.l.], 2005. p. 235–242. Citado na página 38.

ISO/IEC 13407. *ISO/IEC 13407: Human-centered design processes for interactive system*. [S.l.], 1999. Citado 3 vezes nas páginas 11, 37 e 38.

ISO/IEC 9241-11. *NBR ISO/IEC 9241-11: Requisitos ergonômicos para o trabalho com dispositivos de interação visual Parte 11: Orientações sobre usabilidade*. [S.l.], 2003. Citado 2 vezes nas páginas 31 e 60.

KIRAKOWSKI, J.; CORBETT, M. Measuring user satisfaction. In: CAMBRIDGE UNIVERSITY PRESS. *Proceedings of the Fourth Conference of the British Computer Society on People and computers IV*. [S.l.], 1988. p. 329–338. Citado na página 67.

KOSKELA, L. *Test Driven: Pratical TDD and Acceptance TDD for Java Developers*. [S.l.]: Manning Publications, 2007. Citado na página 28.

KRUG, S. *Simplificando coisas que parecem complicadas*. [S.l.: s.n.], 2010. Citado 2 vezes nas páginas 64 e 65.

LEWIS, J. R. Ibm computer usability satisfaction questionnaires: psychometric evaluation and instructions for use. *International Journal of Human-Computer Interaction*, Taylor and Francis, v. 7, n. 1, p. 57–78, 1995. Citado 3 vezes nas páginas 67, 68 e 70.

LOWDERMILK, t. *Design Centrado no Usuário: Um guia para o desenvolvimento de aplicativos amigáveis*. [S.l.: s.n.], 2013. Citado 2 vezes nas páginas 33 e 37.

MASSOL, V.; HUSTED, T. *JUnit in Action*. [S.l.]: Manning Publications, 2003. Citado na página 29.

MAYHEW, D. *The usability engineering lifecycle: a practitioner's handbbok for user interface design*. [S.l.: s.n.], 1999. Citado 2 vezes nas páginas 37 e 59.

MESZAROS, G.; WESLEY, A. *XUnit Test Patterns: Refactoring Test Code*. [S.l.: s.n.], 2007. Citado na página 27.

MOREIRA, T. et al. Proposta de processo de desenvolvimento distribuído de software livre com usabilidade. 2012. Citado na página 19.

NAJAFI, M.; TOYOSHIBA, L. Two case studies of user experience design and agile development. In: *Agile, 2008. AGILE '08. Conference*. [S.l.: s.n.], 2008. p. 531–536. Citado na página 42.

NERUR, S.; MAHAPATRA, R.; MANGALARAJ, G. *Challenges of migrating to agile methodologies*. [S.l.: s.n.], 2005. Citado na página 24.

NIELSEN, J. *Usability engineering*. [S.l.: s.n.], 1994. Citado 3 vezes nas páginas 31, 35 e 65.

NIELSEN, J.; LORANGER, H. *Usabilidade na web: Projetando websites com qualidade*. [S.l.: s.n.], 2007. Citado na página 31.

NORMAN, D. *O design do dia-a-dia*. Rocco, 2006. ISBN 9788532520838. Disponível em: <http://books.google.com.br/books?id=8zd8PgAACAAJ>. Citado na página 32.

POTEL, M.; COTTER, S. *Inside Taligent Technology*. [S.l.]: Taligent Press, 1995. Citado 2 vezes nas páginas 19 e 27.

PREECE, J.; ROGERS, Y.; SHARP, H. *Design de Interação: Além do homem computador*. [S.l.: s.n.], 2007. Citado 4 vezes nas páginas 19, 31, 32 e 59.

SAFFER, D. *Designing for Interaction: Creating Innovative Applications and Devices*. New Riders, 2010. (Voices that matter). ISBN 9780321643391. Disponível em: <http://books.google.com.br/books?id=k28yVW3SEyYC>. Citado na página 32.

SANTOS, A. P. *Aplicação de práticas de usabilidade ágil em software livre*. 2012. Citado 7 vezes nas páginas 19, 20, 35, 38, 42, 64 e 65.

SATO, D. T. *Uso eficaz de métricas em métodos Ágeis de desenvolvimento de software*. 2007. Citado na página 25.

SCHWABER, K. *Agile Project Management with Scrum*. [S.l.]: Microssoft Press, 2004. Citado na página 23.

SPENDOLINI, M. J. *Benchmarking*. [S.l.: s.n.], 1994. Citado na página 63.

STALLMAN, R. M. *Free Software: Freedom and Cooperation*. 2001. Disponível em: <http://www.gnu.org/events/rms-nyu-2001-transcript.txt>. Citado na página 23.

SY, D. Adapting usability investigations for agile user-centered design. *Journal of usability Studies*, New Riders Press, Pearson Education, v. 2, n. 3, p. 112–132, 2007. Citado na página 42.

UNGER, R.; CHANDLER, C. *O Guia para projetar a experiência do usuário (UX) para projetistas de contéudo digital, aplicações e web sites*. [S.l.: s.n.], 2009. Citado 2 vezes nas páginas 64 e 65.

VASCONCELOS, C.; GARCIA, F.; TURNELL, M. Integrando usabilidade e engenharia de software: um modelo para o desenvolvimento de sistemas centrado no usuário. *WIHC-ES2003, Rio de Janeiro*, 2003. Citado na página 39.

VICENTE, A. A. *Definição e gerenciamento de métricas de teste no contexto de métodos ágeis*. Dissertação (Mestrado) — USP - Universidade de São Paulo, 2010. Citado na página 19.