UNIVERSIDADE FEDERAL DE PERNAMBUCO
GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO
CENTRO DE INFORMÁTICA

DESIGN INTERATIVO EM PROCESSOS ÁGEIS DE DESENVOLVIMENTO DE SOFTWARE

Recife, 11 de março de 2005.

Aluno: **Cynthia Belleza Bernardino** (cpbb@cin.ufpe.br)
Orientador: **Alex Sandro Gomes** (asg@cin.ufpe.br)
Co-orientador: **Alexandre Vasconcelos** (amlv@cin.ufpe.br)

”Se o conhecimento pode criar problemas,
não é através da ignorância que podemos solucioná-los.”
(Isaac Asimov)

RESUMO

Este trabalho apresenta a proposta de uma metodologia para o desenvolvimento de sistemas de software levando em consideração a característica de qualidade usabilidade e valores de metodologias ágeis de desenvolvimento de software, o AgileUse©. Trata-se de uma conduta de desenvolvimento leve, iterativa e incremental que pretende focar nos usuários finais do sistema sem exigir recursos excessivos ou conhecimentos pouco acessíveis. Em AgileUse©, práticas do design interativo são evidenciadas como uma característica fundamental na geração de produtos com qualidade. A AgileUse© é composta de fases que podem ser facilmente entendidas e utilizadas por diferentes empresas e diferentes metodologias de desenvolvimento de software. A base da AgileUse© encontra-se na estrutura básica de processos de desenvolvimento unida a especificações das normas ISO 13407 e ISO TR 18529, bem como práticas e valores das Metodologias Ágeis de desenvolvimento de software.

Palavras-chave: Usabilidade, Design Interativo, Engenharia de Software, Processos de Desenvolvimento de Software, Qualidade de Software, Metodologia Ágil, ISO 13407, ISO TR 18529.

ABSTRACT

This work presents the proposal of a methodology for the development of software systems leading in consideration the quality characteristic usability and values of agile methodologies of software development, the AgileUse ©. It is about a behavior of light, iterative and incremental development that it intends to focus in the final users of the system without demanding extreme resources or little accessible knowledge. In AgileUse ©, practical of design interactive are evidenced as a basic characteristic in the generation of products with quality. The AgileUse © is composed of phases that can easily be understood and be used by different companies and different methodologies of software development. The base of the AgileUse © meets in the basic structure of processes of joined development the specifications of norms ISO 13407 and ISO TR 18529, as well as practical and values of the Agile Methodologies of software development.

Keywords: Usability, Interactive Design, Software Engineering, Software Development Processes, Software Quality, Agile Methodology, ISO 13407, ISO TR 18529.

Dedico este trabalho a minha família e amigos.

AGRADECIMENTOS

Ao meu orientador, Alex, pela oportunidade de contar com seu apoio e paciência em cada passo na construção deste trabalho.

Ao meu co-orientador, Alexandre, pela experiência e boa vontade em me tirar as várias dúvidas, mesmo estando tão ocupado.

De maneira especial, aos meus pais e irmãs, pela presença diferencial em minha vida, bem como enorme compreensão em meus momentos de stress com a conclusão do trabalho.

Aos amigos como César, Popis, Rafa, Lipe, Alessandro, Inhow, Paulo, Pinhow, Badu, Manno, XBacon e tantos outros que me ajudaram com o trabalho ou mesmo pela imensa torcida.

À equipe do Laboratório de Avaliação de Produtos de Software (LAPS) por me motivar e permitir que eu agregasse mais conhecimentos na área de qualidade, usabilidade e processos. Obrigada pelas experiências e amizade.

SUMÁRIO

LISTA DE IMAGENS

LISTA DE TABELAS

1 Introdução

Ao longo dos anos, com a popularização dos computadores e a evolução e maior acesso a tecnologias de informação e comunicação (ex: Internet, Caixas de Bancos 24hs etc.), houve o crescimento do número de pessoas (usuários[1]) a interagir com produtos e sistemas informatizados em diversas áreas da atividade humana. Como John Karat [Karat, 1997] descreve, a interação das pessoas com produtos de software tomou novo caráter. Agora, a aceitação de qualquer produto de software não é mais vista como unicamente dependente das características da interface do usuário, mas no modo como o sistema adequa-se dentro do contexto de uso. Para isso, nós temos de entender as necessidades dos usuários como o meio de comunicar o processo de design. A comunidade de Interação Homem-Computador (IHC) tem, geralmente, adotado o termo 'design centrado no usuário' para essa visão de desenvolvimento de sistemas.

Como Bevan & Bogomolni [Bevan & Bogomolni, 2000] apontam, o valor de mercado de um sistema computacional é em função de sua qualidade em uso[2] – o quão se adequa à sua finalidade. Esse aspecto também é referenciado na norma ISO/IEC 14598-1 [ISO 14598-1, 1998] (Validação de Produtos de Software), que coloca a qualidade em uso como o objetivo principal do desenvolvimento de um software.

---

[1] De acordo com [Vasconcelos et al., 2003], entende-se por usuários os seres humanos que irão, de fato, interagir com o sistema. É o usuário que, através dos seus sentidos, percebe e armazena as informações apresentadas para, em seguida, processá-las utilizando um raciocínio lógico. É um ser tomador de decisões. No âmbito de um projeto de software, o usuário pode ser ainda o cliente do projeto ou não.

[2] De acordo com [Bevan & Bogomolni, 2000], o termo qualidade em uso reconhece que o software não existe de forma isolada, devendo adequar-se ao ambiente de trabalho sócio-tecnológico espera-se que funcione na prática.

Existem três abordagens principais para a obtenção da qualidade em uso [Bevan & Bogomolni, 2000]:

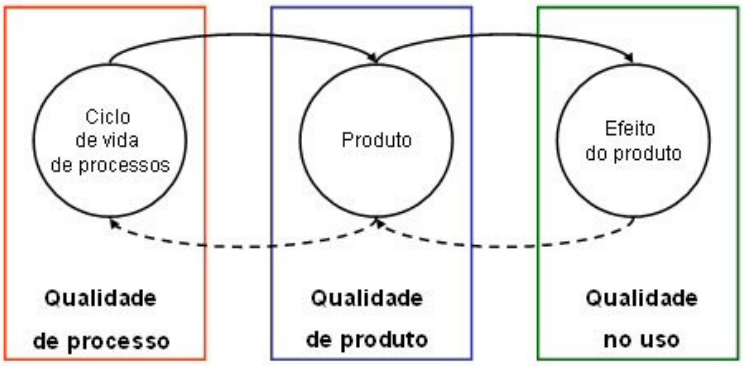

Figura 1. *Abordagens para obtenção da qualidade em uso.*

- Melhoria da qualidade dos processos de desenvolvimento de software através da incorporação de atividades centradas no usuário.
- Melhoria da qualidade do software através da incorporação da qualidade da interface com o usuário.
- Melhoria da qualidade em uso através da certeza que o software atende as necessidades do usuário por efetividade, produtividade e satisfação em uso.

A primeira abordagem mostra-se mais vantajosa devido ao custo de efetuar mudanças crescer com o andamento do projeto e, ainda, por melhorias no processo de desenvolvimento acarretarem na melhor qualidade do produto e, conseqüentemente, no uso.

Nesse contexto, Gould & Lewis [apud Rubin, 1994] apontam três princípios do projeto centrado no usuário:

1 A ênfase no usuário e na tarefa deve ocorrer desde o início;
2 Medidas empíricas da usabilidade[3] do produto devem ser aplicadas;
3 Design interativo deve ocorrer enquanto o produto é projetado, modificado e testado repetidas vezes.

Essa atual ênfase na interface do usuário exige, cada vez mais, uma postura de maior preocupação com a interação do usuário com a interface e, conseqüentemente, sua qualidade. Deste ponto, surge a necessidade de incluir aspectos de design interativo no ciclo de desenvolvimento de produtos de software.

1.1 Problemática

Ao longo dos anos, a ES tem tradicionalmente construído sistemas de software tendo como foco principal suas funcionalidades ou, ainda, suas estruturas internas – um processamento lógico correto, desempenho adequado, entre outros aspectos [Constantine & Lockwood, 2001]. Conseqüentemente, a qualidade destes tipos de sistemas tem sido mensurada através de fatores como reusabilidade, portabilidade, modularidade, acoplamento, robustez – aspectos os quais a maioria dos usuários finais do produto praticamente desconhece. Em contrapartida, a interação do usuário com o sistema tem sido, muitas vezes, vista como um assunto de pouca relevância na construção de softwares. Se, por um lado, alguns desenvolvedores de software percebem o desenvolvimento de sistemas desta forma, por outro, os engenheiros de usabilidade vislumbram o usuário como a peça chave durante todo o processo de desenvolvimento do produto. Eles direcionam seu foco no modo como ele interage com o sistema, o contexto de uso e quais são as tarefas desempenhadas pelos mesmos. Estes engenheiros não estão interessados em metodologias para, por exemplo, fazer com que o sistema em desenvolvimento seja reutilizável, estruturado ou, ainda, fracamente acoplado.

---

[3] De acordo com a ISO 9241-11 [ISO 9241-11, 1998] (Requisitos Ergonômicos para Trabalho de Escritórios com Computadores), usabilidade corresponde até que ponto um produto pode ser utilizado por usuários específicos para alcançar metas também especificadas com eficácia, eficiência e satisfação, em um determinado contexto de uso.

Em um mercado onde a redução cada vez maior do *time-to-market* dos produtos de software constitui um fator chave na sobrevivência de muitas organizações, muitos ainda percebem o emprego de técnicas de usabilidade na confecção de seus produtos como um atraso no cronograma de seus projetos [Ferré, 2001].

Diante desses fatos, empresas de software, em especial as pequenas e médias, fecham-se para a melhoria de seus processos através da interação com o usuário, utilizando-se, para isso, de justificativas como alto custo de utilização das técnicas, restrições de prazo e dificuldade de implantação de uma cultura centrada no usuário. Essas são algumas questões que norteiam e fornecem inspiração para a conclusão desse trabalho de graduação.

1.2 Objetivos da Monografia

1.2.1 Objetivo Principal

O principal objetivo deste trabalho é propor um modelo para o desenvolvimento de sistemas de software que se utilize do design interativo como fator de melhoria da interação do usuário com o produto.

1.2.2 Objetivos Específicos

O modelo proposto deverá, ainda, contemplar alguns objetivos específicos:

- Ser viável a empresas de pequeno e médio porte.
- Provar que a aplicação do design interativo não precisa ser um processo árduo, lento e custoso.
- Considerar a usabilidade não, necessariamente, como o aspecto mais importante de um software, mas um importante fator de qualidade deste.

1.3 Relevância

A relevância deste trabalho é fundamentada na ausência de um modelo que satisfaça as necessidades de pequenas e médias empresas (que possuem orçamento e tempo como fatores restritivos) no quesito usabilidade como fator de qualidade.

A iniciativa de promover o design interativo junto às empresas de software contribui para o crescimento de um foco mais preocupado com o usuário final do produto, deixando de se ater apenas ao cliente e desenvolvedores. Levando a pública que a usabilidade de um sistema não se prende a fatores como beleza e cores, mas à interação do usuário com a interface desde seus objetivos à como as informações são apresentadas em tela.

1.4 Metodologia de Trabalho

A metodologia utilizada no desenvolvimento deste trabalho constitui das seguintes fases:

1. **Revisão bibliográfica das técnicas de design interativo –** primeiramente, procurou-se analisar algumas das principais técnicas atualmente adotadas na área de design interativo. Fez-se necessário compreender o contexto em que são aplicadas, considerando o retorno proporcionado por estas.

2. **Revisão bibliográfica dos processos de desenvolvimento de software –** esta atividade contemplou a investigação dos processos de desenvolvimento difundidos localmente em pequenas e médias empresas. Após um estudo inicial, decidiu-se por aprofundar-se nas metodologias de processos ágeis de desenvolvimento de software.

3. **Revisão da literatura sobre normas de qualidade sobre processos de desenvolvimento de software –** consistiu na pesquisa e entendimento de normas de qualidade sobre processos de desenvolvimento de software. Sendo consideradas, em especial, as normas ISO com integração de aspectos centrados no usuário.

4. **Proposta da metodologia –** neste momento, deu-se a transição do arcabouço para uma versão mais estruturada da metodologia. Buscou-se especificar e estruturar a metodologia integrando os aspectos e técnicas que foram levantados em cada um dos dois universos investigados – Design Interativo e ES.

1.5 Organização da Monografia

A monografia encontra-se organizada em 5 (cinco) capítulos:

O Capítulo 2 apresenta ao leitor um panorama no que se refere à utilização de técnicas de design interativo em projetos de sistema de software. Alguns conceitos fundamentais relacionados à área, bem como o impacto e importância que as atividades de design interativo exercem sobre a qualidade de um produto de software são abordados neste capítulo. Também são apresentadas normas de qualidade acerca de aspectos centrados no usuário no contexto da ES.

O Capítulo 3 provê um panorama do mercado de software atual e relata a boa aceitação que as metodologias ágeis de desenvolvimento de software têm encontrado neste cenário. Também é citada a relação dessas com o design interativo, levando em consideração princípios comuns, lacunas nas metodologias ágeis e cenários de integração.

No Capítulo 4 é fornecida a apresentação do modelo. Aspectos relacionados à sua estrutura são esclarecidos neste capítulo. São descritas suas fases e respectivas atividades, atores, artefatos e técnicas sugeridas.

No Capítulo 5 são apresentadas as conclusões deste trabalho, contribuições providas pelo mesmo, bem como as atividades futuras sugeridas.

2 O Design Interativo no Desenvolvimento de Software

2.1 Design Interativo e Usabilidade

Como descreve Fleming [Fleming, 1998] ao falar da interatividade em sites web, esta não se resume à navegação não-linear ou animações na tela. Ela aborda o que as pessoas podem fazer no site, em que elas podem interagir e o que o site faz para atingir suas necessidades, interesses, objetivos e habilidades.

O design interativo surge como a arte eficaz de criar experiências interessantes para os usuários [Shedroff, 1999], tratando do genuíno engajamento humano como uma das medidas para o sucesso de um software interativo.

Hix e Hartson [Hix & Hartson, 1993] descrevem o designer interativo como o responsável por desenvolver o conteúdo, contexto e aparência do design interativo. Pessoas com essa função são diretamente responsáveis por assegurarem a usabilidade, incluindo o desempenho e satisfação do usuário. A maior parte do trabalho destas está em preocupar-se com a determinação de especificações mensuráveis de usabilidade, validação de designs interativos com os usuários e redesign baseado nas análises dos dados obtidos nas avaliações de uma interface.

O design interativo em questão é abordado na forma de uma metodologia que considera um ciclo de coleta de dados, análise e prototipação através da interação com os usuários, levando em consideração a característica de usabilidade do software.

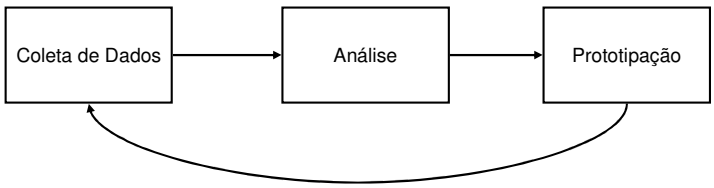

Figura 2. *Ciclo de design interativo.*

2.2 Design Interativo e Engenharia de Software

Na década de 90, a IEEE (*Institute of Eletrical and Eletronics Engineers*) [IEEE, 1990] apresentou uma definição para a ES que tornou-se bastante conhecida pela comunidade de desenvolvimento de sistemas:

"A Engenharia de Software é a aplicação de um processo sistemático, disciplinado e quantificado à concepção, implementação e manutenção do software".

Segundo Pressman [Pressman, 1995], esses métodos que refletem detalhes de "como agir" para se construir um determinado software envolvem um amplo conjunto de tarefas que incluem: (1) planejamento e estimativas de projeto, (2) análise de requisitos de software, (3) projeto de estrutura de dados e arquitetura de programas, (4) codificação, (5) testes e (6) manutenção. Nota-se, portanto, a existência de uma clara e conhecida visão do ciclo de desenvolvimento de um sistema.

2.2.1 Visão Tradicional da Usabilidade

Ferré [Ferré, 2001], ao abordar a integração e utilização das técnicas de usabilidade em processos da ES, afirma que, muitas vezes e equivocadamente, a usabilidade é vista por engenheiros de software e desenvolvedores como sendo somente o processo de construção da interface com o usuário (cores, fontes etc.), o qual, segundo eles, pode ser realizado após as funcionalidades internas dos sistemas terem sido implementadas.

Em contrapartida a essa visão, a ISO 9241-11 [ISO 9241-11, 1998] define usabilidade como: "Até que ponto um produto pode ser utilizado por usuários específicos para alcançar metas também especificadas com eficácia, eficiência e satisfação em um determinado contexto de uso".

Essa visão discrepante sobre o papel da usabilidade na qualidade do produto de software é possivelmente ligada a uma falta de cultura, inclusive sobre o real papel do usuário no retorno de investimento sobre o produto. De acordo com Endler e Pimenta [Endler & Pimenta, 2004], é preciso ter paciência no processo de mudança cultural no desenvolvimento de interfaces de sistemas, pois leva

tempo até que as pessoas assimilem os conhecimentos e consigam colocar em prática novos métodos e técnicas. Para uma integração da cultura de IHC, é necessária a integração entre métodos: os métodos de IHC devem estar integrados aos métodos existentes de desenvolvimento de sistemas, e vice-versa.

2.2.2 Relação da ES com a IHC

Seffah [Seffah, 2002] aborda a relação da ES com a IHC, citando algumas diferenças entre o desenvolvimento de software tradicional e aquele centrado no usuário tomando, como cenário, o atual mercado de software. Pode-se observar, na Figura 3, as principais divergências. Enquanto a visão tradicional da ES trabalha sob um aspecto fundamentalmente técnico centrado na construção de código, a IHC possui como característica a interação entre equipes multidisciplinares, baseados em metodologias advindas da Psicologia, Sociologia, Design Industrial, Comunicação e assim por diante.

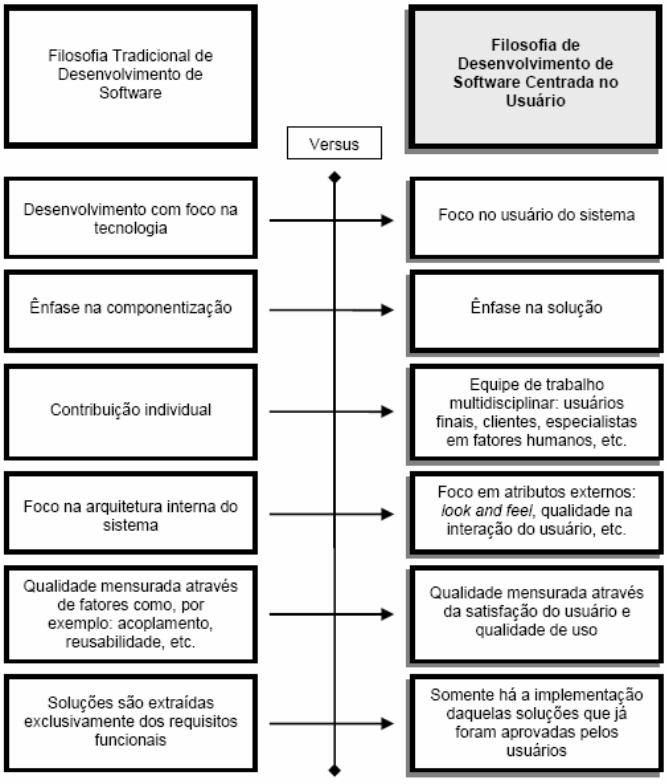

Figura 3. *Relação da ES com a IHC.*

Embora reconhecendo as diferenças, é importante enfatizar que não se pode privilegiar apenas a ES ou a IHC. Ambas devem ser consideradas, havendo uma relação inclusiva e não excludente [Cybis et al., 1998]. O ideal é conscientizar o engenheiro de software dos benefícios oriundos das técnicas de usabilidade e o impacto positivo que elas detêm na qualidade e aceitação de um produto de software por parte de seus usuários.

2.3 Importância das Técnicas para a Obtenção de Usabilidade

Marcus [Marcus, 2002] cita, em seu ensaio, 42 artigos mostrando diferentes aspectos do impacto financeiro da usabilidade. Ele mede-os através de três categorias e suas respectivas subdivisões: desenvolvimento, vendas e uso. As categorias citadas e suas subdivisões estão listadas na Tabela 1.

Desenvolvimento	**Vendas**	**Uso**
Redução de Custos	Aumentar rendimento	Aumentar eficácia
Poupar desenvolvimento de custos	Aumentar transações/compras	Aumentar taxa de sucesso
Poupar o desenvolvimento de tempo	Aumentar venda de produtos	Reduzir o erro do usuário
Reduzir a manutenção de custos	Aumentar tráfego	Aumentar a produtividade do usuário
Poupar custos de redesign	Reter clientes	Aumentar a satisfação do usuário
	Atrair uma parcela de mercado	Aumentar a satisfação do empregado
		Aumentar a facilidade de uso
		Aumentar facilidade de aprender
		Aumentar a confiança no sistema
		Diminuir custo de suporte
		Diminuir custo de treinamento

Tabela 1. *Categorias e subcategorias para medição do Retorno de Investimento da Usabilidade.*

Percebe-se que muitos são os benefícios ganhos ao empregar-se as técnicas para a obtenção de usabilidade em um processo de desenvolvimento. Esta pode, ainda, ser vista como fator crítico para a aceitação de um sistema de software

pelo usuário. Se o produto não provê suporte à realização de suas tarefas, provavelmente será rejeitado. É possível que haja também uma subutilização dos recursos oferecidos pelo sistema, insatisfação dos usuários e, até mesmo, o fracasso de um projeto como um todo. Vale ressaltar que, mesmo que um produto esteja sendo utilizado, isso não significa necessariamente que o software seja adequado com relação à usabilidade. É importante destacar que existem outros aspectos que condicionam o uso de um produto de software, como a capacidade de escolha de seus usuários, opções de produtos similares e custos associados.

Assim, de um modo geral, a principal motivação para a aplicação de práticas para a obtenção de usabilidade nos processos de desenvolvimento de software é aumentar a eficiência, satisfação e conseqüente produtividade do usuário.

2.4 Normas de Qualidade sobre Processos Centrados no Usuário

Hoje são numerosos os livros, revistas, normas e relatórios técnicos que nos apresentam métodos, técnicas e ferramentas para a montagem de uma capacidade em termos de usabilidade nas empresas.

Existem publicações que orientam a como especificar, construir e testar a usabilidade, como qualidade de uso e qualidade externa de um sistema de software interativo. Outras nos informam de centenas de técnicas participativas ou documentais para o projeto e a avaliação da usabilidade de interfaces. Livros e normas orientam a montagem de um processo de desenvolvimento centrado no usuário, como a norma ISO 13407 e a ISO TR 18529.

O objetivo desses dois padrões é assegurar que o desenvolvimento e uso de sistemas interativos levem em consideração as necessidades dos usuários bem como as dos desenvolvedores e *stakeholders*[4] em geral

[4] O termo *stakeholder* foi introduzido para nomear a todos os envolvidos no projeto, diretamente ou indiretamente, ou que tenha interesse no resultado do projeto.

2.4.1 ISO 13407: Processo Centrado no Usuário

A ISO 13407 [ISO 13407, 1999] provê instruções sobre como conseguir qualidade no uso incorporando atividades de projeto centrado no usuário através do ciclo de vida de sistemas interativos computacionais (vide Apêndice A). Ela descreve o projeto centrado no usuário como uma atividade multidisciplinar, a qual incorpora fatores humanos, conhecimento ergonômico e técnicas com o objetivo de acentuar a efetividade e a produtividade, melhorando as condições de trabalho humano e contra-atacando possíveis efeitos adversos do uso na saúde, segurança e performance humana.

Nela existem quatro atividades de projeto centrado no usuário que precisam ser iniciadas nos estágios iniciais de um projeto. São elas:

- Entender e especificar o contexto de uso;
- Especificar os requisitos do usuário e da organização;
- Produção de soluções de projeto;
- Validação dos projetos baseada nos requisitos.

A natureza iterativa dessas atividades é ilustrada na Figura 4. O processo envolve iterações até que os objetivos sejam satisfeitos.

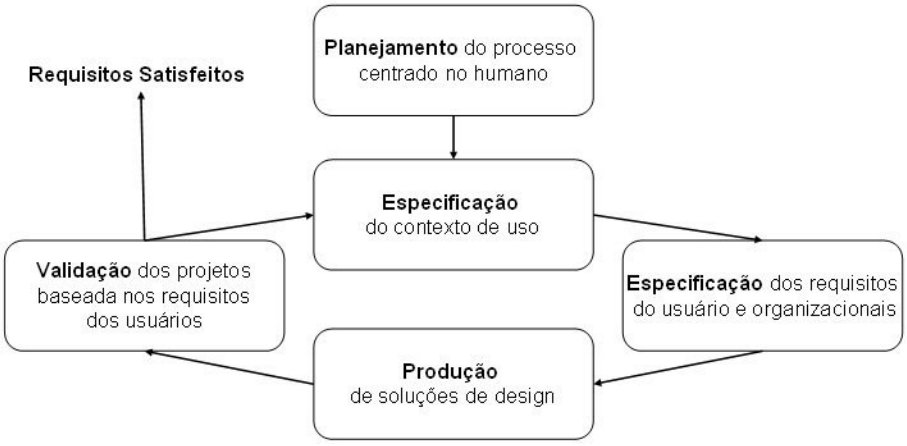

Figura 4. *A interdependência das atividades do projeto centrado no usuário.*

A seqüência em que elas são executadas e o nível de esforço e detalhe apropriado varia de acordo com o ambiente do projeto, o estágio do processo de projeto e objetivos almejados no quesito usabilidade.

2.4.2 ISO TR 18529: Descrição do Processo de Ciclo de Vida Centrado no Humano

A INUSE desenvolveu uma definição estruturada e formalizada de processos centrados no humano descritos na ISO 13407. Uma versão melhorada foi, subseqüentemente, publicada como ISO TR 18529 [ISO TR 18529, 2000]. Sua intenção é tornar o conteúdo da ISO 13407 acessível aos especialistas em avaliação de processos de software e aos familiarizados com ou envolvidos na modelagem de processos. A ISO TR 18529 pode ser usada na especificação, avaliação e melhoria de processos centrados no humano no desenvolvimento e operação de sistemas.

O modelo consiste de sete conjuntos de práticas básicas (vide Apêndice B). Essas práticas descrevem o que deve ser feito em ordem para representar e incluir os usuários de um sistema durante o ciclo de vida. O modelo usa o formato comum de modelos de avaliação de processos. Esses modelos descrevem os processos que devem ser executados por uma organização para alcançar objetivos técnicos definidos. Os processos nesse modelo são descritos no formato definido pela ISO 15504 - Avaliação de Processos de Software [ISO 15504, 1998]. Embora o uso primário de um modelo de avaliação de processo seja para a medição de quão bem uma organização cumpre com os processos incluídos no modelo, tais modelos podem ser usados como uma descrição do que é requerido com o intuito de projetar e desenvolver processos de projeto e organizacionais efetivos.

3 Metodologias Ágeis de Desenvolvimento

3.1 Contexto

Aceitando que a teoria de custos diferenciais no ciclo de vida de Boehm [Boehm, 1981] – o custo da mudança cresce através do ciclo de vida de desenvolvimento de software – mantém-se válida, a questão atual não é como deter mudanças em um projeto, mas como adequar-se da melhor forma a mudanças inevitáveis.

Como Cockburn & Highsmith [Cockburn & Highsmith, 2001] apontam, métodos tradicionais evitam as mudanças através de listas de riscos e definição antecipada de requisitos, tentando, assim, reduzir os custos pela eliminação das mudanças. Eles assumem que as variações são resultantes de erros no planejamento. Entretanto, atualmente, enquanto problemas no processo certamente causam alguns erros, mudanças no ambiente externo causam variações críticas, devendo ser aceitas e não ignoradas ou evitadas.

É nesse panorama que as chamadas "metodologias ágeis" se inserem. A necessidade de processos minimalistas – com pouca burocracia envolvida no desenvolvimento – e de equipes ágeis de software que sejam adaptáveis, capazes de reagir rapidamente a mudanças e de agregar valor ao cliente em um ritmo constante é muito grande. Segundo Beck [Beck, 2000], a abordagem tradicional de desenvolvimento de software é pouco aderente ao contexto dinâmico e imprevisível do atual mercado e suas tecnologias. Assim sendo, as metodologias ágeis buscam uma abordagem adequada a esta realidade: focar em pessoas ao invés de processos, gerar rapidamente valor para o cliente e, sobretudo, "abraçar" as freqüentes mudanças de requisitos.

3.2 Metodologias Ágeis

3.2.1 Ágil versus Tradicional

Tanto as metodologias ágeis quanto as tradicionais (caracterizadas por serem guiadas por planejamento) possuem seus pontos fortes e fraquezas. Uma comparação pode ser feita baseada em algumas áreas principais (vide Tabela 2).

Principais Áreas	Metodologias Ágeis	Metodologias guiadas por Planejamento
Desenvolvedores	Ágil, voltado ao conhecimento, arranjado e colaborativo.	Orientado ao planejamento, habilidades adequadas, acesso ao conhecimento externo.
Clientes	Dedicado, voltado ao conhecimento, arranjado, colaborativo, representativo e com poder.	Acesso ao conhecimento, colaborativo, representativo e clientes com poder.
Requisitos	Largamente emergentes, mudanças rápidas.	Conhecidos previamente, largamente estáveis.
Arquitetura	Projetada para requisitos atuais.	Projetada para requisitos atuais e futuros.
Refatoração	Barata.	Cara.
Tamanho	Pequenos times e produtos.	Grandes times e produtos.
Objetivo Primário	Valor rápido.	Garantia elevada.

Tabela 2. *Principais áreas para metodologias ágeis e guiadas por planejamento.* [Boehm, 2002]

Desenvolvedores. Cockburn & Highsmith [Cockburn & Highsmith, 2001b] enfatizam vários fatores críticos sobre a equipe para metodologias ágeis: amizade, talento, habilidade e comunicação. Eles consideram que, quando o trabalho é executado por pessoas competentes, poucas pessoas podem produzir melhor do que se estivessem sozinhas. A grande diferença é que as metodologias ágeis baseiam-se muito mais na agilidade ao confiar no conhecimento tácito existente na equipe do que em escrever o conhecimento em planos. Por outro lado, as metodologias tradicionais reduzem o risco de ocorrerem erros por problemas na transferência desse conhecimento, utilizando-se de arquiteturas de ciclo de vida e planos, facilitando revisões por especialistas externos.

Clientes. Como constatado por Deursen [Deursen, 2001], as metodologias ágeis funcionam melhor quando os clientes colaboram com o time de desenvolvimento e quando seu conhecimento tácito é suficiente para o completo entendimento da aplicação. Novamente, ainda há o risco da má-compreensão desse conhecimento, o qual é reduzido pelas metodologias tradicionais através de documentação, arquiteturas e apoio de especialistas externos ao projeto.

Requisitos. De acordo com o ponto de vista de Highsmith e Cockburn [Highsmith e Cockburn, 2001], organizações são sistemas complexos adaptativos, onde os requisitos são mais emergentes que previsíveis. Entretanto, para componentes de softwares onde a estabilidade e a segurança são vitais, metodologias tradicionais podem evitar erros.

Arquitetura. Dentre os valores das metodologias ágeis está o trabalho através de documentação compreensível, enfatizando a simplicidade: maximizando a quantidade de trabalho não feito. Entretanto, em situações onde os requisitos futuros são previsíveis, essa prática não apenas dispensa arquiteturas de valor que os suportariam como cria problemas com clientes que desejam acreditar que suas prioridades e requisitos de evolução estão sendo acomodados no projeto.

Refatoração. Com desenvolvedores competentes e pequenos sistemas, a suposição de que a refatoração é essencialmente barata é válida. Entretanto, evidências empíricas indicam que, fora dessa realidade, o esforço para a refatoração cresce com o número de requisitos.

Tamanho. Constantine [Constantine, 2001b] cita que metodologias ágeis são mais atrativas para pequenos projetos pela necessidade de alta-coordenação da equipe, sendo aconselhável a empresas burocráticas e com grandes projetos o uso de metodologias tradicionais. Entretanto, Cockburn & Highsmith [Cockburn & Highsmith, 2001b] citam sucessos ocasionais em projetos ágeis de grande porte com mais de 250 pessoas.

Objetivos Primários. Um dos princípios ágeis é o de dar alta-prioridade à satisfação do cliente através da entrega contínua e em pouco tempo de software de valor. Mas em sistemas de grande porte, essa atitude pode levar a maior retrabalho quando a arquitetura não possui escalabilidade. Nesses casos, um bom planejamento é necessário.

3.2.2 O Movimento Ágil

O "Movimento Ágil" na indústria de software teve como guia o Manifesto Ágil [Manifesto, 2001] publicado por um grupo de desenvolvedores e consultores no ano de 2001. Esse manifesto declara que, em ambientes ágeis, se preza:

- **Indivíduos e interações** mais que processos e ferramentas;
- **Software funcionando** mais que documentação extensiva e detalhada;
- **Colaboração do cliente** mais que negociações contratuais;
- **Responder às mudanças** mais que seguir um plano.

Esses valores centrais a qual a comunidade ágil adere são assim detalhados:

Primeiro, o movimento ágil enfatiza o relacionamento e comunalidade de desenvolvedores de software e o papel humano refletido nos contratos, como oposição aos processos institucionalizados e ferramentas de desenvolvimento. Nas práticas ágeis existentes, isto se manifesta em relacionamentos próximos da equipe, organização do ambiente de trabalho, e outros procedimentos para impulsionar o espírito de equipe.

Segundo, o objetivo vital do time de software é testar, continuamente, o que foi produzido. Novas versões são produzidas em intervalos freqüentes - em alguns métodos a cada hora ou diariamente, mas usualmente a cada dois meses ou mensalmente. Os desenvolvedores são chamados a manter o código simples, objetivo, e tecnicamente da forma mais avançada possível, assim diminuindo a carga de documentação a um nível apropriado.

Terceiro, o relacionamento e cooperação entre os desenvolvedores e os clientes é dado como preferencial em detrimento a contratos estritos, embora a importância de contratos bem esboçados cresça ao mesmo passo que o tamanho do projeto de software. O processo de negociação em si deve ser visto como um meio de firmar e manter um relacionamento viável. De um ponto de vista de negócio, o desenvolvimento ágil é focado na entregas mediante ao começo do projeto, reduzindo, assim, os riscos de não-cumprimento a respeito do contrato.

Quarto, o grupo de desenvolvimento, compreendendo tanto desenvolvedores de software e representantes do cliente, deve estar bem-informado, competente e autorizado a considerar a necessidade de possíveis ajustes que surjam durante o processo de desenvolvimento. Isso significa que os participantes são preparados para fazer mudanças e que os contratos existentes são formados com ferramentas que suportam e permitem que esses realces sejam feitos.

O manifesto expõe, ainda, doze princípios fundamentais:

1. "*nossa maior prioridade é satisfazer o cliente através da entrega rápida e contínua de software de valor*";
2. "*mudanças nos requisitos do sistema são bem vindas, mesmo que de última hora*";
3. "*forneça versões de pequenas porções de software freqüentemente, em poucas semanas*";
4. "*cliente e desenvolvedor devem trabalhar juntos ao longo de todo o projeto*";
5. "*faça projetos com pessoas motivadas, disponibilize o ambiente e o suporte necessário, então confie que elas possam fazer o trabalho*";
6. "*o método mais eficiente para colher informações sobre o projeto é através de conversas face-a-face com o cliente*";
7. "*a principal medida de progresso de um projeto é trabalhar em cima de software e não de documentação*";
8. "*processos ágeis promovem desenvolvimento sustentável. Clientes e desenvolvedores devem descobrir seu ritmo de trabalho e mantê-lo constantemente*";
9. "*a atenção contínua às boas práticas e a um bom design*[5] *promovem agilidade no projeto*";
10. "*simplicidade ao longo do projeto é essencial*";
11. "*as melhores arquiteturas, requisitos e design emergem de times auto-organizados*";
12. "*em intervalos regulares, o time de desenvolvimento refletirá como se tornar mais efetivo. Então poderá ajustar seu comportamento conforme as necessidades adequadamente*".

[5] Nesse ponto, no sentido de projeto.

Serão apresentadas, a seguir, algumas metodologias de desenvolvimento de software ágeis.

3.2.3 Metodologias Mais Utilizadas

3.2.3.1 Extreme Programming (XP)

A *Extreme Programming* (XP) é uma metodologia ágil para equipes pequenas e médias que desenvolvem software baseado em requisitos vagos e que se modificam rapidamente [Beck, 2000]. Dentre as principais diferenças da XP em relação a outras metodologias estão:

- *Feedback* constante;
- Abordagem incremental;
- A comunicação entre as pessoas é encorajada.

O primeiro projeto a usar XP foi o C3, da Chrysler. Após anos de fracasso utilizando metodologias tradicionais, com o uso da XP o projeto ficou pronto em pouco mais de um ano [Highsmith & Cockburn, 2000].

A maioria das regras da XP causa polêmica à primeira vista e muitas não fazem sentido se aplicadas isoladamente. É a sinergia de seu conjunto que sustenta o sucesso de XP, encabeçando uma verdadeira revolução no desenvolvimento de software.

A XP enfatiza o desenvolvimento rápido do projeto e visa garantir a satisfação do cliente, além de favorecer o cumprimento das estimativas. As regras, práticas e valores da XP proporcionam um agradável ambiente de desenvolvimento de software para os seus seguidores, que são conduzidos por quatro valores: comunicação, simplicidade, *feedback* e coragem [Beck, 2000].

A finalidade do princípio de comunicação é manter o melhor relacionamento possível entre clientes e desenvolvedores, preferindo conversas pessoais a outros meios de comunicação. A comunicação entre os desenvolvedores e o gerente do projeto também é encorajada. A simplicidade visa permitir a criação de código simples que não deve possuir funções desnecessárias. Por código simples

entende-se implementar o software com o menor número possível de classes e métodos. Outra idéia importante da simplicidade é procurar implementar apenas requisitos atuais, evitando-se adicionar funcionalidades que podem ser importantes no futuro. A aposta da XP é que é melhor fazer algo simples hoje e pagar um pouco mais amanhã para fazer modificações necessárias do que implementar algo complicado hoje que talvez não venha a ser usado, sempre considerando que requisitos são mutáveis.

A prática do *feedback* constante significa que o programador terá informações constantes do código e do cliente. A informação do código é dada pelos testes constantes, que indicam os erros tanto individuais quanto do software integrado. Em relação ao cliente, o *feedback* constante significa que ele terá freqüentemente uma parte do software totalmente funcional para avaliar. O cliente então, constantemente, sugere novas características e informações aos desenvolvedores. Eventuais erros e não conformidades são rapidamente identificados e corrigidos nas próximas versões. Desta forma, a tendência é que o produto final esteja de acordo com as expectativas reais do cliente.

É necessário coragem para implantar os três valores anteriores. Por exemplo, não são todas as pessoas que possuem facilidade de comunicação e têm bom relacionamento. A coragem também dá suporte à simplicidade, pois assim que a oportunidade de simplificar o software é percebida, a equipe pode experimentar. Além disso, é preciso coragem para obter *feedback* constante do cliente.

A XP baseia-se nas 12 (doze) práticas [Beck, 2000] a seguir:

- **Planejamento**: consiste em decidir o que é necessário ser feito e o que pode ser adiado no projeto. A XP baseia-se em requisitos atuais para desenvolvimento de software, não em requisitos futuros. Além disso, a XP procura evitar os problemas de relacionamento entre a área de negócios (clientes) e a área de desenvolvimento. As duas áreas devem cooperar para o sucesso do projeto, e cada uma deve focar em partes específicas do projeto. Desta forma, enquanto a área de negócios deve decidir sobre o escopo, a composição das versões e as datas de entrega, os desenvolvedores devem decidir sobre as estimativas de prazo, o

processo de desenvolvimento e o cronograma detalhado para que o software seja entregue nas datas especificadas.

- **Entregas freqüentes**: visa à construção de um software simples, e conforme os requisitos surgem, há a atualização do software. Cada versão entregue deve ter o menor tamanho possível, contendo os requisitos de maior valor para o negócio. Idealmente devem ser entregues versões a cada mês ou, no máximo, a cada dois meses, aumentando a possibilidade de *feedback* rápido do cliente. Isto evita surpresas caso o software seja entregue após muito tempo, melhora as avaliações e o *feedback* do cliente, aumentando a probabilidade do software final estar de acordo com os requisitos do cliente.
- **Metáfora**: são as descrições de um software sem a utilização de termos técnicos, com o intuito de guiar o desenvolvimento do software.
- **Projeto simples**: o programa desenvolvido pelo método XP deve ser o mais simples possível e satisfazer os requisitos atuais, sem a preocupação de requisitos futuros. Eventuais requisitos futuros devem ser adicionados assim que eles realmente existirem. Esta forma de raciocínio se opõe ao "implemente para hoje e projete para amanhã".
- **Testes**: a XP focaliza a validação do projeto durante todo o processo de desenvolvimento. Os programadores desenvolvem o software criando primeiramente os testes.
- **Refatoração**: focaliza o aperfeiçoamento do projeto do software e está presente em todo o desenvolvimento. A refatoração deve ser feita apenas quando é necessário, ou seja, quando um desenvolvedor da dupla, ou os dois, percebe que é possível simplificar o módulo atual sem perder nenhuma funcionalidade.
- **Programação em pares**: a implementação do código é feita em dupla, ou seja, dois desenvolvedores trabalham em um único computador. O desenvolvedor que está com o controle do teclado e do mouse implementa o código, enquanto o outro observa continuamente o trabalho que está sendo feito, procurando identificar erros sintáticos e semânticos e pensando estrategicamente em como melhorar o código que está sendo implementado. Esses papéis podem e devem ser alterados continuamente. Uma grande vantagem da programação em dupla é a possibilidade dos desenvolvedores estarem continuamente aprendendo um com o outro.

- **Propriedade coletiva**: o código do projeto pertence a todos os membros da equipe. Isto significa que qualquer pessoa que percebe que pode adicionar valor a um código, mesmo que ele próprio não o tenha desenvolvido, pode fazê-lo, desde que faça a bateria de testes necessária. Isto é possível porque na XP todos são responsáveis pelo software inteiro. Uma grande vantagem desta prática é que, caso um membro da equipe deixe o projeto antes do fim, a equipe consegue continuar o projeto com poucas dificuldades, pois todos conhecem todas as partes do software, mesmo que não seja de forma detalhada.
- **Integração contínua**: interagir e construir o sistema de software várias vezes por dia, mantendo os programadores em sintonia, além de possibilitar processos rápidos. Integrar apenas um conjunto de modificações de cada vez é uma prática que funciona bem porque fica óbvio quem deve fazer as correções quando os testes falham: a última equipe que integrou código novo ao software. Esta prática é facilitada com o uso de apenas uma máquina de integração, que deve ter livre acesso a todos os membros da equipe.
- **40 horas de trabalho semanal**: a XP assume que não se deve fazer horas extras constantemente. Caso seja necessário trabalhar mais de 40 horas pela segunda semana consecutiva, existe um problema sério no projeto que deve ser resolvido não com aumento de horas trabalhadas, mas com melhor planejamento, por exemplo. Esta prática procura ratificar o foco nas pessoas e não em processos e planejamentos. Caso seja necessário, os planos devem ser alterados, ao invés de sobrecarregar as pessoas.
- **Cliente presente**: é fundamental a participação do cliente durante todo o desenvolvimento do projeto. O cliente deve estar sempre disponível para sanar todas as dúvidas de requisitos, evitando atrasos e até mesmo construções erradas. Uma idéia interessante é manter o cliente como parte integrante da equipe de desenvolvimento.
- **Código padrão**: padronização na arquitetura do código, para que este possa ser compartilhado entre todos os programadores.

3.2.3.2 Scrum

Outra metodologia ágil que apresenta uma comunidade grande de usuários é a Scrum [Schwaber & Beedle, 2002]. Seu objetivo é fornecer um processo conveniente para projeto e desenvolvimento orientado a objeto. A Scrum apresenta uma abordagem empírica que aplica algumas idéias da teoria de controle de processos industriais para o desenvolvimento de softwares, re-introduzindo as idéias de flexibilidade, adaptabilidade e produtividade. O foco da metodologia é encontrar uma forma de trabalho dos membros da equipe para produzir o software de forma flexível e em um ambiente em constante mudança.

A idéia principal da Scrum é que o desenvolvimento de softwares envolve muitas variáveis técnicas e do ambiente, como requisitos, recursos e tecnologia, que podem mudar durante o processo. Isto torna o processo de desenvolvimento imprevisível e complexo, requerendo flexibilidade para acompanhar as mudanças. O resultado do processo deve ser um software que é realmente útil para o cliente [Schwaber, 1995].

A metodologia é baseada em princípios semelhantes aos da XP: equipes pequenas, requisitos pouco estáveis ou desconhecidos e iterações curtas para promover visibilidade para o desenvolvimento. No entanto, as dimensões em Scrum diferem de XP.

A Scrum divide o desenvolvimento em iterações (*sprints*) de trinta dias. Equipes pequenas, de até dez pessoas, são formadas por projetistas, programadores, engenheiros e gerentes de qualidade. Estas equipes trabalham em cima de funcionalidades (os requisitos, em outras palavras) definidas no início de cada *sprint*. A equipe é responsável pelo desenvolvimento desta funcionalidade.

Na Scrum existem reuniões de acompanhamento diárias. Nessas reuniões, que são preferencialmente de curta duração (aproximadamente quinze minutos), são discutidos pontos como o que foi feito desde a última reunião e o que precisa ser feito até a próxima. As dificuldades encontradas e os fatores de impedimento (*bottlenecks*) são identificados e resolvidos.

O ciclo de vida da Scrum é baseado em três fases principais, divididas em subfases:

1 **Pré-planejamento** (*Pre-game phase*): os requisitos são descritos em um documento chamado *backlog*. Posteriormente eles são priorizados e são feitas estimativas de esforço para o desenvolvimento de cada requisito. O planejamento inclui também, entre outras atividades, a definição da equipe de desenvolvimento, as ferramentas a serem usadas, os possíveis riscos do projeto e as necessidades de treinamento. Finalmente é proposta uma arquitetura de desenvolvimento. Eventuais alterações nos requisitos descritos no *backlog* são identificadas, assim como seus possíveis riscos.

2 **Desenvolvimento** (*Game phase*): as muitas variáveis técnicas e do ambiente identificadas previamente são observadas e controladas durante o desenvolvimento. Ao invés de considerar essas variáveis apenas no início do projeto, como no caso das metodologias tradicionais, na Scrum o controle é feito continuamente, o que aumenta a flexibilidade para acompanhar as mudanças. Nesta fase o software é desenvolvido em ciclos (*sprints*) em que novas funcionalidades são adicionadas. Cada um desses ciclos é desenvolvido de forma tradicional, ou seja, primeiramente faz-se a análise, em seguida o projeto, implementação e testes. Cada um desses ciclos é planejado para durar de uma semana a um mês.

3 **Pós-planejamento** (*Post-game phase*): após a fase de desenvolvimento são feitas reuniões para analisar o progresso do projeto e demonstrar o software atual para os clientes. Nesta fase são feitas as etapas de integração, testes finais e documentação.

3.3 Metodologias Ágeis e Design Interativo

Patton [Patton, 2002] cita que a utilização de metodologias ágeis permite a entrega de software de qualidade em pouco tempo, enquanto que o design interativo exerce o papel de garantir um grau satisfatório de empatia com o usuário final.

3.3.1 Princípios Comuns

Como Nikula & Sajaniemi [Nikula & Sajaniemi, 2002] citam, ambos tentam destilar os elementos essenciais de suas respectivas disciplinas e focar nesses princípios.

Interações e Indivíduos. Entender os usuários e seus relacionamentos com o software e ambiente são o coração do design interativo a fim de tornar o sistema mais prazeroso e efetivo.

Software Funcionando. Para o design interativo, software funcionando significa ser possível compreender a interface entre o sistema e o usuário.

Colaboração com Cliente. Além de poderem garantir acesso aos usuários, a cooperação com o cliente é importante no ganho de flexibilidade ao construir softwares mais usáveis.

Resposta à Mudança. No design interativo, a resposta à mudança diz respeito a como o time adapta-se às diferenças de suposições entre desenvolvedores e usuários através de mecanismos de *feedback*. É não apenas responder às mudanças para tornar o software mais usável, mas também servir como um catalisador para mudanças, fazendo aspectos de usabilidade mais evidentes no processo de desenvolvimento.

3.3.2 Possíveis Lacunas nas Metodologias Ágeis

Constantine [Constantine, 2001c] observa que, infelizmente, o projeto da interface com o usuário e a usabilidade são largamente negligenciados pelas metodologias ágeis. Com algumas possíveis exceções, usuários e suas interfaces são todos ignorados. XP e outras metodologias leves são leves também no lado do usuário do software. Eles parecem dar o seu melhor em aplicações que não necessitam de trabalho intensivo com a interface com o usuário.

Algumas lacunas são apontadas por Kane [Kane, 2003] como oportunidades para considerar aspectos de usabilidade. São elas:

***Feedback* do Usuário.** Todas as metodologias ágeis consideram o *feedback* do usuário como um elemento-chave. XP, por exemplo, funciona com um cliente no local de trabalho a fim de prover *feedback* ao time de desenvolvimento. Entretanto, mesmo esse cliente, tipicamente, sendo um especialista na área do software, ele acaba por se tornar um especialista no software sendo desenvolvido

por conta de seu trabalho regular com o time de desenvolvimento. Isso é de valor, mas é difícil para esse tipo de usuário antecipar interesses potenciais de diferentes grupos de usuários.

Teste de Aceitação. Esses testes são usados em metodologias ágeis para verificar quando uma porção particular do trabalho foi completada, e para confirmar que as últimas versões não incorrerão em erros. Enquanto esses testes de aceitação definem o comportamento externo do software, é muito difícil (se não, impossível), através desses testes, refletir o entendimento humano que constitui a usabilidade do software.

Guias de Estilo. Tipicamente, essas guias de estilo tomam a forma de guias focados na programação em si, e não na interface com o usuário.

Padrões. Utilizados e citados, em sua maioria, com foco em decisões de projeto [Gamma et al., 1994], os padrões de software ajudam o engenheiro de software a tomar melhores decisões. Bem como esses padrões capturam os *tradeoffs* de decisões de projeto de software, padrões de interface com o usuário podem ajudar desenvolvedores a fazerem melhores escolhas sobre os *tradeoffs* relacionados à usabilidade [Pertzel & Kane, 1999].

3.3.3 Cenários de Integração

Através de pesquisa bibliográfica, foram encontradas algumas tentativas de aliar a característica de usabilidade ao desenvolvimento de software. Dentre elas, podemos citar o trabalho de Bankston [Bankston, 2002], Constantine [Constantine, 1999] e Vasconcelos [Vasconcelos et al., 2004].

Bankston – Usabilidade e Projeto da Interface do Usuário em XP

Bankston baseou-se, em especial, na idéia de que metodologias ágeis são perfeitamente adequadas a testes de usabilidade ao permitir que as equipes tomem ações imediatas baseadas no *feedback* recebido dos usuários, maximizando os benefícios de cada iteração e reduzindo os custos a longo prazo. Através dessa idéia de ciclo de testes de aceitação, ele apresentou o modelo que se segue (Figura 5):

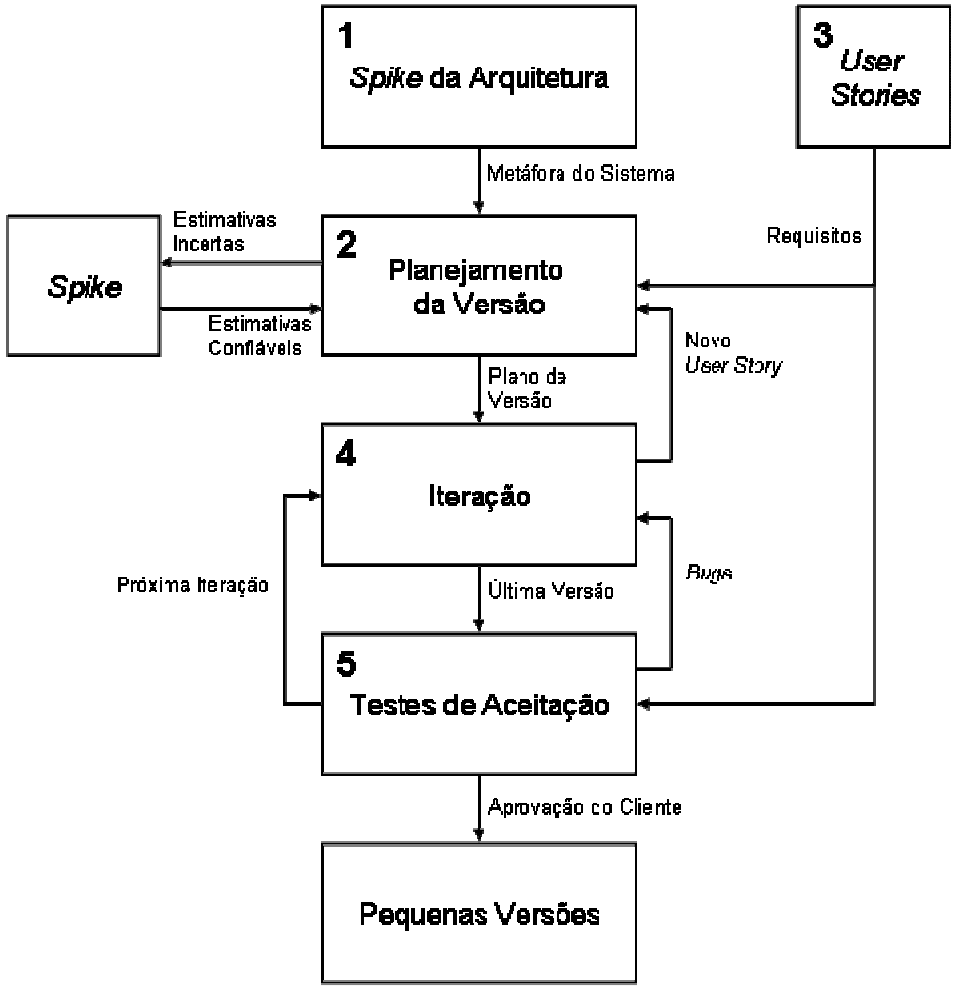

Figura 5. *O modelo proposto por Bankston.*

Correspondendo a numeração a:

1. Engajamento com usuários finais e definição inicial das tarefas e regras do usuário;
2. Priorização e solidificação das regras dos usuários e *task cards*;
3. Tradução dos requisitos na interface com o usuário, para então defini-los em *user stories*;
4. Trabalho com os desenvolvedores a fim de implementar a interface a ser visualizada e
5. Condução de testes de usabilidade com usuários finais.

Constantine – Agile Usage-centered Design

Considerando um time que inclui tanto designers como desenvolvedores e, ainda, ao menos um usuário ou um usuário representante (como um especialista do domínio), Constantine define as seguintes atividades:

1. Construa de um inventário de regras dos usuários (*user roles*) através de *brainstorm* diretamente para cartões indexados (*index cards*).
2. Revise e refine o inventário de cartões, então o descreva brevemente, salientando aspectos de cada regra em seu cartão.
3. Ordene os cartões com as regras dos usuários a fim de estabelecer um *ranking* de acordo com sua prioridade para o sucesso do projeto.
4. Construa um inventário de casos de uso (os mais essenciais) para dar suporte às regras do usuário identificadas pelo *brainstorming* diretamente nos cartões, começando com as regras de maior prioridade.
5. Ordene os cartões para priorizá-los, primeiro pela freqüência antecipada ou pela sua comunalidade e, por último, por importância total para o sucesso do projeto.
6. Classifique os cartões em três categorias: necessário (faça na primeira versão), desejado (faça se tiver tempo durante essa versão), descartável. Marque os cartões com as suas categorias.
7. Para os cartões do tipo necessário, juntamente com os desejados que sejam críticos, complexos, não-claros ou interessantes, escreva o corpo da narrativa no cartão. A narrativa deve abordar o “caso de sucesso” (fluxo normal dos eventos) de forma essencial (abstrata, simplificada e livre de tecnologias) usando o formato padrão de duas colunas: uma com as intenções do usuário e outra com as responsabilidades do sistema. Extensões, alternativas ou casos de "falhas" da narrativa podem ser adicionados no verso de cada cartão posteriormente.
8. Armazene os cartões (todos eles) em grupos de afinidade baseado em suas semelhanças e interdependência.
9. Trate cada grupo de cartões como um conjunto de tarefas a serem trabalhadas através por um contexto de interação na interface do usuário, e então faça um esboço inicial através da prototipação em papel para essa parte da interface, concentrando-se nos casos de tarefas necessárias, porém, considerando outras (todos os *task cases* são considerados para garantir que a arquitetura da interface do usuário esteja harmoniosa).

10. Inspecione o protótipo com os usuários e clientes usando cenários derivados dos *task cases*.
11. Revise e refine o protótipo em papel baseado nos resultados da inspeção.
12. Comece programando a interface do usuário ou a camada de apresentação baseado no protótipo em papel e *task cases* associados.

Como na maioria das metodologias ágeis, o consenso é que o desenvolvimento desenrola-se através de ciclos sucessivos de versões. Iterações sucessivas irão adicionar novas regras e *task cases* a fim de guiar o refinamento e expansão da interface do usuário.

Vasconcelos - XPU

O objetivo principal do XPU (eXtreme Programming + Usabilidade) é prover à equipe de desenvolvimento um modelo para a construção de sistemas de software centrado no usuário que valorize a usabilidade como característica fundamental de qualidade.

O XPU reflete um processo de construção de sistemas de software dividido em 7 (sete) fases específicas: (1) definição de papéis, (2) conversa com o cliente, (3) inicialização, (4) planejamento de *releases*, (5) planejamento da iteração, (6) implementação e (7) verificação de testes. Na Figura 6, algumas atividades e artefatos do modelo são revelados.

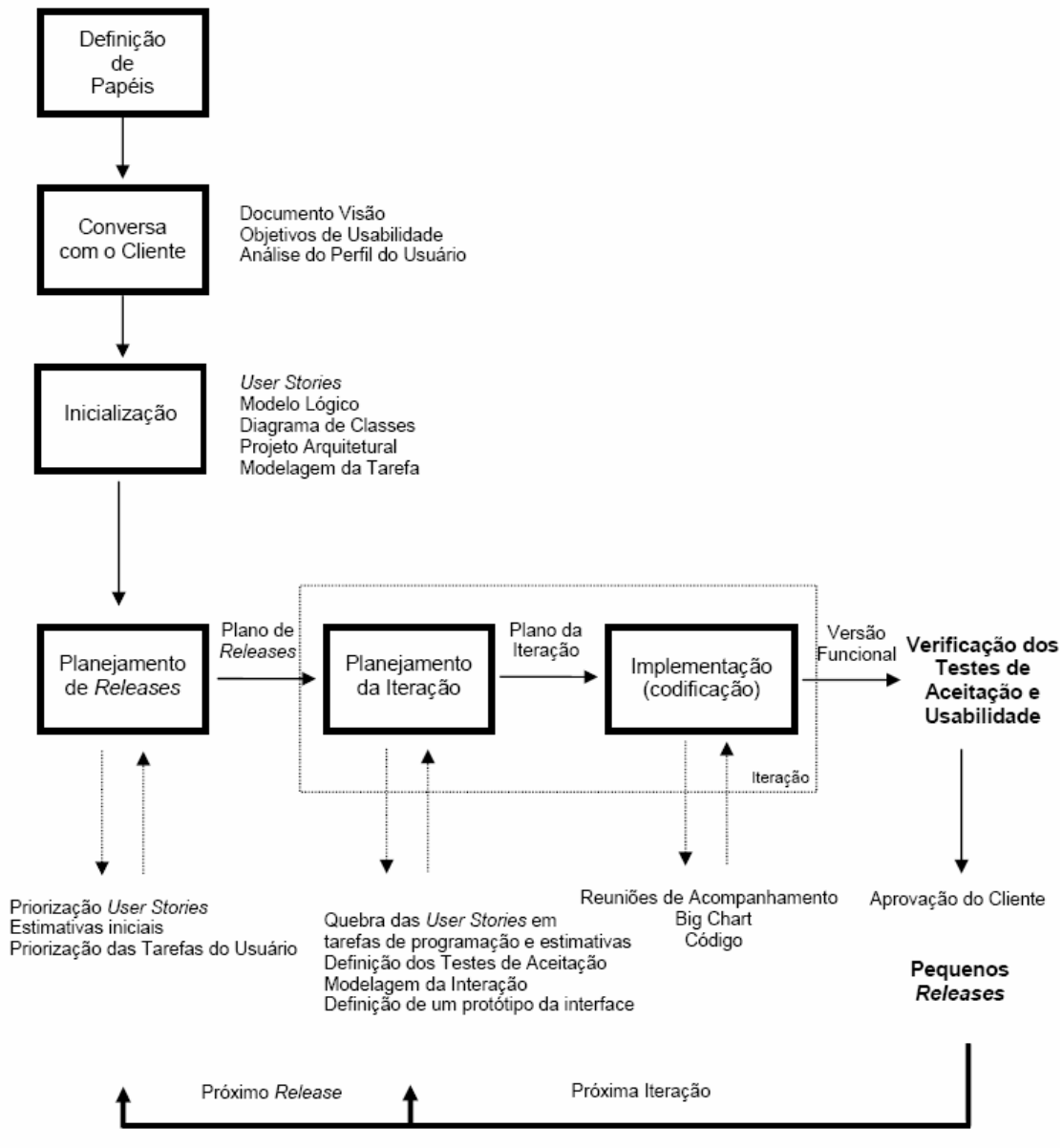

Figura 6. *Estrutura do modelo XPU.*

4 A Metodologia AgileUse

4.1 Introdução

A metodologia AgileUse surgiu da necessidade de viabilizar, de forma simples e eficaz, a inserção do design interativo, e conseqüente valorização da usabilidade, em empresas de pequeno e médio porte. Ela baseia-se na estrutura básica de processos de desenvolvimento unida a especificações das normas ISO 13407 e ISO TR 18529, bem como em práticas e valores das Metodologias Ágeis de desenvolvimento de software.

No restante deste capítulo, a metodologia será descrita, inicialmente através de sua estrutura para, então, detalhar suas fases e características.

4.2 Estrutura da Metodologia AgileUse

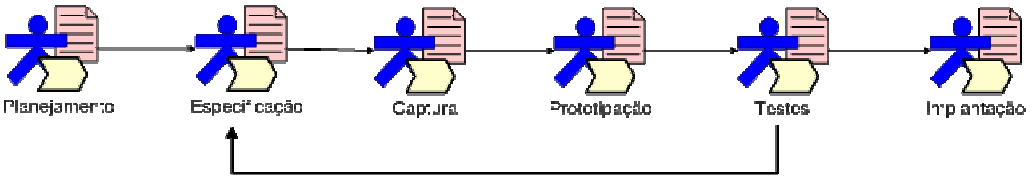

Figura 7. *Estrutura da metodologia AgileUse.*

A AgileUse divide-se em 6 fases básicas: Planejamento, Especificação, Captura, Prototipação, Testes e Implantação. Essas fases estão baseadas em conceitos comuns de desenvolvimento de software e na ISO TR 18529 (e, conseqüentemente, na ISO 13407). As fases de Planejamento, Especificação (Análise), Testes e Implantação são comuns em ambos. A fase de Captura foi trazida à metodologia pela necessidade de capturar dados sobre os usuários, tarefas e seu contexto; enquanto que a Prototipação propõe-se a concretizar a idéia do design interativo por meio de elaboração de pequenas versões funcionais do software para serem, posteriormente, testadas por usuários.

4.3 Detalhamento das Fases

4.3.1 Fase de Planejamento

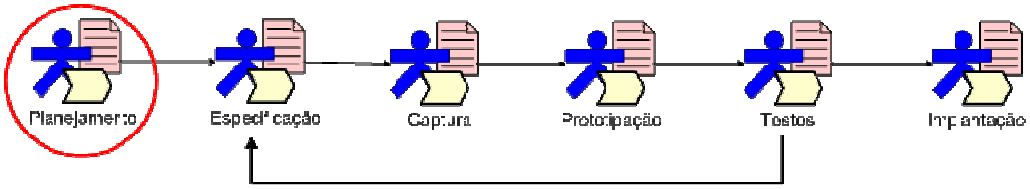

Figura 8. *Estrutura da metodologia AgileUse – Fase de Planejamento.*

Objetivo

Especificar como as atividades de design interativo se encaixam no projeto (ciclo de vida, recursos etc.) e formalizá-las através de um plano de projeto.

Atividades e Atores

- **Planejamento com o Cliente** – O Designer Interativo deverá participar da reunião de planejamento de projeto com o Cliente a fim entender, através de questionamentos padrões e específicos ao contexto do produto, onde o design interativo será de valor no desenvolvimento do produto.
    - Atores: *Designer Interativo e Cliente*
- **Identificar e Planejar o Envolvimento do Usuário** – Através dos dados obtidos na reunião anterior com o Cliente, o Designer Interativo identificará, baseado em sua experiência e projetos anteriores da empresa, em quais fases do desenvolvimento será mais intensa a participação do usuário.
    - Atores: *Designer Interativo*
- **Selecionar Métodos e Técnicas** – A partir de informações como recursos disponíveis para testes, tempo do projeto e usuários disponíveis para participar do projeto da interface, o Designer Interativo definirá os métodos e técnicas mais adequadas às expectativas sobre o papel do design interativo no projeto.
    - Atores: *Designer Interativo*
- **Planejar as Atividades** – Serão formalizadas em um documento de plano de projeto do design interativo (específico de acordo com a necessidade para o entendimento pelo Cliente e dentro da equipe do projeto) informações como

cronograma, responsáveis pelas atividades, recursos necessários, técnicas e métodos a serem utilizados etc.

- Atores: *Designer Interativo*

Artefatos Finais

- **Plano de Projeto do Design Interativo** – Documento contendo informações como as técnicas adotadas, cronograma e responsáveis pelas atividades, ambiente necessário, etc.

Técnicas Sugeridas

- **Entrevista com o Cliente** – Conversa face-a-face com o Cliente, tendo como guia um questionário semi-estruturado, ou seja, com perguntas definidas antes da entrevista, mas podendo haver a adição de novos questionamentos de acordo com o andamento desta.

4.3.2 Fase de Especificação

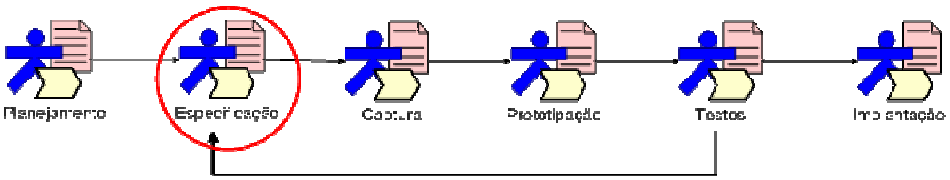

Figura 9. *Estrutura da metodologia AgileUse – Fase de Especificação.*

Objetivo

Estabelecer os requisitos dos *stakeholders*, levando em conta as tarefas e contexto de uso do software.

Atividades e Atores

- **Definir *Stakeholders*** – A partir dos perfis de usuários descritos na reunião com o Cliente durante o planejamento, o Designer Interativo deverá pesquisar e definir, através do próprio contexto de uso do software ou por outras fontes, todos os envolvidos no projeto, diretamente ou indiretamente, ou que tenha interesse no resultado deste.
  - Atores: *Designer Interativo*
- **Definir o Sistema (tarefas, usuários e contexto de uso)** – Tendo em mãos os perfis dos *stakeholders*, o Designer Interativo deverá entender o sistema junto a

um Especialista do Domínio do software a ser produzido. Deverão ser construídos, se possível, modelos das tarefas (passos, pessoas envolvidas, artefatos de entrada e saída etc.).

  - Atores: *Designer Interativo e Especialista do Domínio*

- **Gerar os Requisitos** – Após a compreensão do sistema, o Designer Interativo reunir-se-á junto ao Analista de Sistemas do projeto e apresentará a visão do Especialista do Domínio sobre o sistema. Juntos, eles deverão gerar requisitos básicos para a operação das tarefas identificadas na atividade anterior.
  - Atores: *Designer Interativo e Analista de Sistemas*

Artefatos Finais

- **Rascunho do Documento de Requisitos do Usuário** – Documento com requisitos básicos para a perfeita operação das tarefas identificadas pelo Especialista do Domínio e estruturadas pelo Designer Interativo e Analista de Sistemas do projeto.
- **Plano de Verificação dos Requisitos junto ao Usuário** – Com os requisitos básicos definidos, os usuários serão solicitados para a sugerir outros requisitos que influenciem na melhor utilização do software. Para isso, urge definir um plano para servir de arcabouço nessa pesquisa, incluindo número e perfis dos usuários a serem buscados, cronograma etc.

Técnicas Sugeridas

- **Entrevista com Especialista do Domínio** – O Especialista será solicitado a descrever as tarefas e informações envolvidas em seu domínio, estando o Designer Interativo atento no decorrer da discussão para que não se fuja do escopo do software a ser produzido.
- **Brainstorm com Analista(s) de Sistemas** – O Brainstorm [Jones, 1980] é uma técnica que consiste em reunir um grupo de pessoas com o objetivo de produzir o maior número de idéias e depois selecioná-las para achar a solução do problema. Nesta fase, o objetivo é captar objetivos necessários à correta execução das tarefas identificadas anteriormente junto ao Especialista do Domínio.

4.3.3 Fase de Captura

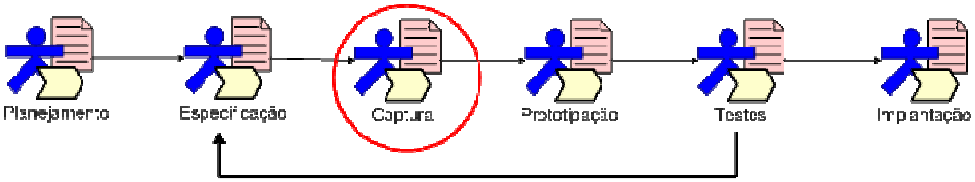

Figura 10. *Estrutura da metodologia AgileUse – Fase de Captura.*

Objetivo

Esclarecer as características identificadas na fase anterior (usuários, tarefas e contexto de uso), validando e identificando novos requisitos do usuário.

Atividades e Atores

- **Validar e Identificar Novos Requisitos do Usuário** – Por observação dos usuários ou mesmo interação com estes, o Designer Interativo deverá validar os requisitos definidos na fase anterior e complementar a lista de requisitos do usuário, caso necessário.
    - Atores: *Designer Interativo*

Artefatos Finais

- **Documento de Requisitos do Usuário** – Descrição das funções oferecidas pelo sistema aos usuários, descrevendo operações disponíveis dependendo do perfil do usuário.

Técnicas Sugeridas

- **Etnografia Rápida** [Millen, 2000] - Estudos rápidos realizados para fornecer informações gerentes sobre o ambiente onde será implantado o sistema. O Designer Interativo vai ao local onde o sistema será posto em execução para captar aspectos relevantes sobre o contexto de uso, os usuários e suas tarefas.
- **Entrevistas com Usuários** – O Designer Interativo consulta uma amostra de usuários de perfis distintos para validar os requisitos e, caso seja relevante ao projeto, adicionar novos requisitos.

4.3.4 Fase de Prototipação

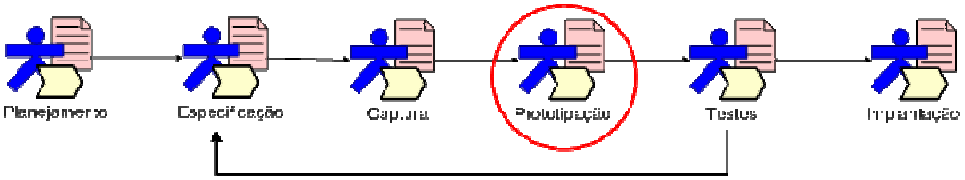

Figura 11. *Estrutura da metodologia AgileUse – Fase de Prototipação.*

Objetivo

Criar potenciais soluções de projeto da interface.

Atividades e Atores

- **Especificar o Sistema** – O Designer Interativo deverá reunir-se com a equipe de design gráfico para entrar em consenso sobre o conteúdo da interface a fim de permitir a realização das tarefas especificadas no documento de requisitos de forma eficaz, eficiente e satisfatória.
    - Atores: *Designer Interativo e Designer Gráfico*
- **Desenvolver os Protótipos** – A partir dos elementos básicos da interface, determinados na atividade anterior, a equipe de design gráfico produzirá protótipos da interface do usuário, tendo a supervisão do Designer Interativo.
    - Atores: *Designer Interativo e Designer Gráfico*

Artefatos Finais

- **Protótipo da Interface do Usuário** – Protótipo visual da interface que deverá permitir uma avaliação da interação com os usuários do sistema.

Técnicas Sugeridas

- **Prototipação em Papel** – Prototipação em papel [Rettig, 1994] é uma variação do teste de usabilidade onde usuários representativos realizam tarefas reais através da interação com uma versão em papel da interface que é manipulada por uma pessoa (funcionando como o computador), explicando como a interface é pressuposta a funcionar. Nesta fase, apenas é produzido o protótipo para, na próxima fase, ocorrer a fase de testes com os usuários finais do software.
- **Prototipação em Flash ou HTML** – Mesmo funcionamento básico da prototipação em papel. Entretanto, não há a necessidade de haver uma pessoa

explicando a interface, pois o próprio protótipo dá o retorno sobre o impacto das ações do usuário.

4.3.5 Fase de Testes

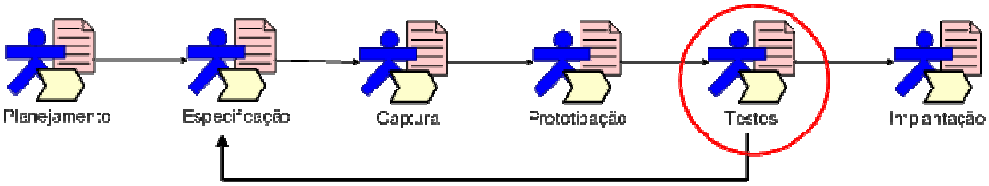

Figura 12. *Estrutura da metodologia AgileUse – Fase de Testes.*

Objetivo

Avaliar a interação do usuário e fontes representativas (ex: Especialista em Usabilidade e Especialista no Domínio) com o sistema ou protótipo.

Atividades e Atores

- **Validar o Protótipo/Sistema** – O protótipo, ou mesmo o sistema final, produzido na fase anterior será validado por seus usuários finais, tendo como coordenador da avaliação o Designer Interativo. Tanto o Designer Interativo quanto o Especialista do Domínio podem, também, exercer o papel de avaliadores da interface em questão.
    - Atores: *Designer Interativo e Usuários*

Artefatos Finais

- **Relatório de Avaliação** – Documento com problemas detectados e a serem considerados na próxima versão do software ou protótipo. Podendo conter seu nível de gravidade (impacto) e descrição de como resolvê-los.

Técnicas Sugeridas

- **Avaliação Heurística** – A Análise Heurística [Nielsen, 1992] é um método de avaliação de usabilidade em que profissionais examinam cada elemento de uma interface e os julgam segundo princípios básicos de usabilidade, os princípios heurísticos.
- **Grupo Focado** [Caplan, 1990] – Evento onde pesquisadores reunem-se com usuários potenciais de um produto na tentativa de planejar como melhorá-lo.

- **Percurso Cognitivo** [Wharton et al., 1994] - Método em que profissionais criam tarefas e cenários a partir de especificações de protótipos iniciais e então atuam como usuários reais trabalhando na interface.
- **Teste de Usabilidade** [Nielsen, 1993] - Método em que um número representativo de usuários finais do produto é observado por profissionais, enquanto executam tarefas pré-estabelecidas na interface.

4.3.6 Fase de Implantação

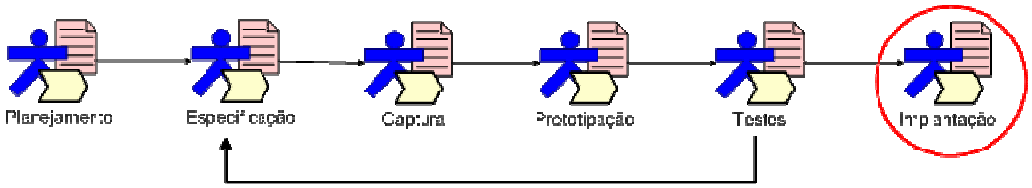

Figura 13. *Estrutura da metodologia AgileUse – Fase de Implantação.*

Objetivo

Obter *feedback* dos *stakeholders* após a implantação do software.

Atividades e Atores

- **Determinar o Impacto do Sistema no Contexto de Uso** – Após a implantação do sistema, o Designer Interativo fica responsável por entrar em contato com usuários de perfis distintos para obter *feedback* sobre a usabilidade e aspectos gerais do sistema. Os problemas detectados deverão ser documentados para posterior consulta caso haja o desenvolvimento de uma nova versão do software, evitando incorrer nos mesmos erros.
    - Atores: *Designer Interativo*
- **Suporte ao Usuário** – A equipe de Suporte da empresa fica encarregada de apoiar o usuário em possíveis dificuldades na execução do software, não esquecendo de documentar problemas e reportá-los à equipe de desenvolvimento e Design Interativo.
    - Atores: *Equipe de Suporte*

Artefatos Finais

- **Relatório de Impacto do Sistema Pós-Implantação** – Documento periódico com problemas e comentários relevantes para a melhoria do software.

Técnicas Sugeridas

- **Questionário de Satisfação do Usuário** – Questionário com perguntas gerais sobre a interação com o ambiente e específicas sobre o sistema, obtendo feedback sobre a eficácia, eficiência e satisfação após utilizar o software.
- **Entrevista com os Usuários** - Consulta a uma amostra de usuários de perfis distintos no ambiente real de uso.

5 Conclusões

Neste trabalho, apresentamos a união de valores e práticas de metodologias ágeis ao design interativo, criando assim a metodologia AgileUse. Apesar de ainda poder ser mais especificada, a estrutura estabelecida serve como passo inicial para o estabelecimento de uma cultura de usabilidade aliada a processos de desenvolvimento e qualidade de software.

Inicialmente, apresentamos as principais características e conceitos do design interativo, abordando tanto sua relação com a Engenharia de Software (ES) como enfatizando seu papel como fator de qualidade. Observamos a crescente utilização das metodologias ágeis, bem como suas principais metodologias e relação com o design interativo. Logo após, introduzimos os principais elementos da metodologia AgileUse.

Através de uma pesquisa com empresas de desenvolvimento software locais (Apêndice C), pudemos observar que a preocupação com os usuários do sistema ainda é precária, juntamente com a cultura de usabilidade. Nessas condições, uma das contribuições deste trabalho é fornecer o arcabouço para a definição de abordagens próprias oriundas da metodologia AgileUse, condicionada ao seu contexto de uso.

5.1 Trabalhos Futuros

Ao concluir este trabalho, podemos identificar alguns encaminhamentos futuros no sentido de aperfeiçoar a especificação da metodologia para o desenvolvimento de software com o usuário como um dos focos. Além disso, podemos apresentar algumas recomendações de trabalhos futuros:

- É necessária a implantação da metodologia proposta em projetos reais de pequenas e médias empresas. Através dessa ação, detectar incompatibilidades e aspectos a serem focados durante cada fase do desenvolvimento.
- Determinar média de tempo e capital investidos em projetos que utilizem a AgileUse, verificando as variáveis que influenciam esse resultado.
- Detalhar as atividades através de fluxos, passo-a-passo, entradas e saídas de dados, bem como os responsáveis por cada passo e artefato.
- Produzir modelos dos artefatos a serem utilizados durante o uso da metodologia e descrever roteiros (passo-a-passo) para a utilização das técnicas sugeridas.

- Especificar ou mesmo desenvolver ferramentas que possam ser utilizadas como apoio em cada fase da metodologia.
- Estender a iniciativa de inserir o design interativo, de forma viável, no desenvolvimento de software a outras metodologias e processos, tais como RUP e XP.

5.2 Considerações Finais

É importante lembrar que um dos objetivos dessa integração do design interativo com a ES foi o de estimular o crescimento de uma cultura mais focada nos requisitos dos usuários finais. Com esse trabalho, foi possível perceber que a usabilidade é uma forma de agregar valor ao software. Entretanto, para que as empresas passem desse entendimento para a real aplicação de técnicas de design interativo ainda há um obstáculo: mostrar, através de projetos, que a adequação a metodologias como a AgileUse trazem um bom retorno de investimento.

6 Referências Bibliográficas

[Antunes et al., 2001] Antunes, H.; Seffah, A. and Djouab, R. *Comparing and Reconciling Usability-Centered and Use-Case Driven Requirements Engineering Processes*. In IEEE: 2nd Australian User Interfaces Conference (AUIC'01), January 29 – February, (2001), Gold Coast, Queensland, Australia.

[Bankston, 2002] Bankston, A. *Usability and User Interface Design in XP*, White paper, July 2002, Disponível em: http://www.ccpace.com/Resources/UsabilityinXP.doc.

[Beck, 2000] Beck, K. *Extreme Programming Explained: Embrace Change*. Addison-Wesley, 2000.

[Bevan & Bogomolni, 2000] Bevan, N.; Bogomolni, I. *Incorporating User Quality Requirements in the Software Development Process*. Proceedings of 4th International Software Quality Week Europe, Brussels, 2000, pages 1192-1204.

[Boehm & Barry, 1981] Boehm, Barry W., *Software Engineering Economics*, Prentice Hall, 1981.

[Boehm, 2002] Boehm, B., *Get Ready for Agile Methods, with Care*. IEEE Computer, 2002. 35(1): p.64-69.

[Caplan, 1990] Caplan, S. *Using focus group methodology for ergonomic design*. Ergonomics, 33, 527-533, 1990.

[Cockburn & Highsmith, 2001] Cockburn, A., Highsmith, J. *Agile Software Development: The People Factor*, Computer, Nov. 2001, pp. 131-133.

[Cockburn & Highsmith, 2001b] Cockburn, A. and Highsmith, J., *Agile Software Development: The Business of Innovation*, IEEE Computer, pp. 120-122, Sept. 2001.

[Constantine, 1999] Constantine, L. L. 1999. *Process Agility and Software Usability: Toward Lightweight Usage-Centered Design*, Constantine & Lockwood, Ltd.

[Constantine, 2001] Constantine, L. & Lockwood, L. *Process Agility and Software Usability Toward Lightweight Usage-Centered Design*, The Management Forum, Software Development, Vol. 9, No. 6, June (2001).

[Constantine, 2001b] Constantine, L. *Methodological Agility*, Software Development, June 2001, pp. 67-69.

[Constantine, 2001c] Constantine, L. *Agile Usage-Centered Design*, forUse Newsletter, No. 12, April 2001, Disponível em: http://foruse.com/newsletter/foruse12.htm.

[Cybis et al., 1998] Cybis, W.; Pimenta, M.; Silveira, M.; Gamez, L. *Uma Abordagem Ergonômica para o Desenvolvimento de Sistemas Interativos*. Atas IHC98. Disponível em: http://www.unicamp.br/~ihc99/Ihc99/AtasIHC99/AtasIHC98.html (30 de janeiro de 2004).

[Deursen, 2001] Deursen, A., *Customer Involvement in Extreme Programming: XP2001 Workshop Report*, ACM Software Eng. Notes, Nov. 2001, pp. 70-73.

[Endler et al., 2004] Endler, A.; Pimenta, M. *Inserindo IHC em Empresas Brasileiras de Informática: Relato, Discussão e Lições Aprendidas*. Anais do VI Simpósio sobre Fatores Humanos em Sistemas Computacionais (IHC2004). Curitiba: SBC, 2004. v. 1, p. 101-109.

[Ferré, 2001] Ferré, X. *Incorporating Usability into a Object Oriented Development Process*. In INTERACT 2001. 8th IFIP TC.13 Workshop on Human-Computer Interaction.Waseda University Conference Centre, Shinjuku Tokyo, Japan, July 9-13, (2001).

[Fleming, 1998] Fleming, J. *Web Navigation: Designing the User Experience*. O'Reilly, Beijing, 1998.

[Gamma et al., 1994] Gamma, E.; Helm, R.; Johnson, R.; Vlissides, J. *Design Patters*, Reading, MA: Addison-Wesley, 1994.

[Highsmith & Cockburn, 2000] Highsmith, J. Orr, K. Cockburn, A. *Extreme Programming*, E-Business Application Delivery, Feb., (2000), pp. 4- 17.

[Hix & Hartson, 1993] Hix, D. & Hartson, H. *Developing User Interfaces: Ensuring Usability Through Product & Process*. John Wisley & Sons, 1993.

[IEEE, 1990] IEEE. IEEE Std. 610.12-1990. *IEEE Standard Glossary or Software Engineering Terminology*. IEEE, New York NY, 1990.

[ISO 13407, 1999] ISO 13407. *Human-centred design processes for interactive systems*, 1999.

[ISO 14598-1, 1998] ISO/IEC 14598-1. *Information Technology – Evaluation of Software Products – Part 1 General Guide*, 1998.

[ISO 15504, 1998] ISO/IEC 15504: 1998 – *Information Technology – Software Process Assesmen*.

[ISO 9241-11, 1998] ISO/IEC 9241-11. *Ergonomic Requirements for Office Work with Visual Displays Terminals*, ISO, Geneva, 1998.

[ISO TR 18529] ISO TR 18529. *Ergonomics of human system interaction - human-centred lifecycle process descriptions*, 2000.

[Jones, 1980] Jones, J. *Design Methods: Seeds of Human Futures*. Chichester: Wiley, 1980.

[Kane, 2003] Kane, D. *Finding a Place for Discount Usability Engineering in Agile Development: Throwing Down the Gauntlet*, Agile Development Conference 2003, pp. 40-46.

[Karat, 1997] Karat, J. *Evolving the scope of user-centred design*. Communications of the ACM, 40, 7, 1997, 33-38.

[Manifesto, 2001] *Manifesto for Agile Software Development*, 2001. Disponível em: http://www.agilemanifesto.org/ (07 de fevereiro de 2005).

[Marcus, 2002] Marcus, A. *Return on Investment for Usable User-Interface Design: Examples and Statistics*. Aaron Marcus and Associates, Inc . (AM+A), 2002.

[Millen, 2000] Millen, D. *Rapid Ethnography: Time Deepening Strategies for HCI Field Research*. Proceedings of the ACM 2000 conference for Designing interactive systems: processes, practices, methods, and techniques. New York, NY: ACM Press, 2000, pp. 280-286.

[Nielsen, 1992] Nielsen, J. *Finding usability problems through heuristic evaluation*. Em P. Bauersfels, J. Bennett & G. Lynch, Eds. Human Factors in Computing Systems - CHI'92, pp. 373-380. New York: ACM & Addison-Wesley, 1992.

[Nielsen, 1993] Nielsen, J. Usability Engineering. Academic Press, Cambridge, MA, 1993.

[Nikula et al., 2002] Nikula, U., Sajaniemi, J. *BaSyRE: A Lightweight Combination of Proven RE Techniques*, Proceedings of the International Workshop on Time Constrained Requirements Engineering, September 9-13, 2002.

[Patton, 2002] Patton, J. *Hitting the target: adding interaction design to agile software development*. Conference on Object Oriented Programming Systems Languages and Applications, OOPSLA 2002 Practitioners Reports, Seattle, Washington, SESSION: Session table of contents, 2002.

[Perzel & Kane, 1999] Perzel, K.; Kane, D. *Usability Patterns for Applications on the World Wide Web*, Pattern Languages of Programming 1999, August 1999.

[Pressman, 1995] Pressman, R. *Engenharia de Software*. Makron Books, 1995.

[Rettig, 1994] Rettig, M. *Prototyping for tiny lingers*. Communications of the ACM, 37, 21-27, 1994.

[Rubin, 1994] Rubin, J. *Handbook of Usability Testing: how to plan, design and conduct effective tests*. New York. J. Wiley, 1994.

[Schwaber & Beedle, 2002] Schwaber, K. e Beedle, M. *Agile Software Development with SCRUM*, Prentice-Hall, (2002).

[Schwaber, 1995] Schwaber, K. *Scrum Development Process*, OOPSLA'95 Workshop on Business Object Design and Implementation. Springer-Verlag. (1995).

[Seffah, 2002] Seffah, A. *Human-Centered Software Engineering: Designing for and with Humans*. Canadian Undergraduate Software Engineering Conference, March 7-9, 2002, Montreal, Canadá. Disponível em: http://www.cusec.ca/archives/cusec2002/keynoteSeffah.ppt (13 de dezembro de 2004).

[Shedroff, 1999] Shedroff, N. *Interaction Design Course Syllabus*, 1999. Disponível em: http://www.nathan.com/thoughts/course.html.

[Vasconcelos et al., 2003] Vasconcelos, C.; Garcia, F.; Turnell, M. *Integrando Usabilidade e Engenharia de Software: um modelo para o desenvolvimento de sistemas centrado no usuário*. WIHC-ES 2003: Integrating Human-Computer Interaction and Software Engineering Models and Processes, Rio de Janeiro – RJ, Brazil, August 2003.

[Vasconcelos et al., 2004] Vasconcelos, C.; Garcia, F.; Turnell, M. 2004. *XPU – Um Modelo para o Desenvolvimento de Sistemas Centrado no Usuário*, Dissertação de Mestrado, Campina Grande – PB, Brasil.

[Wharton et al., 1994] Wharton, C., Rieman, J., Lewis, C., Polson, P. *The Cognitive Walkthrough: A Practitioner´s Guide*. Em J. Nielsen (ed.) Usability Inspection Methods. John Wiley, New York, 1994.

Apêndice A: ISO 13407

Esse apêndice contém um resumo da ISO 13407:1999 *Processos de projeto centrado no humano para sistemas interativos* e apresenta seu sumário.

Título	ISO 13407 Processos de projeto centrado no humano para sistemas interativos
Data	Julho de 1999
Escopo	Guia para atividades centradas no humano através do ciclo de vida de sistemas de computadores iterativos.
Conteúdo	Rasão para o processo centrado no usuário. Uma descrição dos quatro principios essenciais de um projeto centrado no humano. Planejamento do projeto do processo centrado no usuário. Descrição das quatro atividades que devem ser realizadas durante o processo de deselvolvimento de um sistema. Uma lista de processos atuais e padrões de produto para projeto centrado no usuário.
Propósito	ISO 13407 objetiva ajudar aqueles responsáveis por gerenciar os processos de projeto de hardware e software a fim de identificar e planejar atividades de projeto centradas no usuários efecientes e precisas.
Audiência	Aqueles que gerenciam o processo de projeto. Todas as partes envolvidas do desenvolvimento de sistemas baseados no humano, incluindo os usuários finais dos sistemas, devem achar o padrão que seja relevante.
Requisitos	Qualquer processo de desenvolvimento que reivindique ter cumprido as recomendação na ISO 13407 deve especificar os procedimentos utilizados, informações coletadas e o uso feito dos resultados.

Tabela A. Visão geral da ISO 13407.

As cláusulas do padrão são as seguintes:

Introdução

1. Escopo
2. Termos e definições
3. Estrutura desse Padrão Internacional
4. Razão para adotar esse processo de projeto centrado no humano
5. Princípios de projeto centrado no humano

   5.1. Geral

Apêndice B: ISO TR 18529

Esse apêndice apresenta um resumo da ISO TR 18529:2000 *Descrição de processos de ciclo de vida de projeto centrado no humano* e descrições parciais de processos de projeto centrado no humano (HCD, *human-centered design*) contidas nesse padrão.

Processos HCD consideram os usuários finais e outros stakeholders na especificação, desenvolvimento e operação de um sistema. Os processos sempre remetem ao sistema funcional sendo desenvolvido, não apenas aos detalhes de hardware e software. Os processos levam em conta as atividades centradas no humano durante toda a vida do sistema. Cada descrição de processo é apresentada na forma: número do processo, título, propósito, lista de resultados, lista de títulos de práticas. Práticas são passadas através do uso de métodos, técnicas e ferramentas. Métodos, técnicas e ferramentas particulares centrados no humano não são descritos na ISO TR 18529. No entanto, algumas notas explicativas para as práticas ilustram os requisitos de métodos, técnicas e ferramentas. Os próximos parágrafos resumem cada processo de projeto centrado no humano, seus resultados e práticas.

HCD 1: INSERIR NA ESTRATÉGIA DO SISTEMA OS CONCEITOS DO HCD

O propósito do processo HCD 1 é estabelecer e manter um foco nas questões dos *stakeholders* e do usuário nas partes da organização que lidam com marketing, conceitos, desenvolvimento e suporte de sistemas. Como resultado de uma bem sucedida implementação desse processo teremos que:

- O marketing levará em conta questões de usabilidade, ergonomia e socio-técnicas.
- O sistema estará focado em satisfazer as necessidades e expectativas do usuário.
- Os planejadores irão considerar os *stakeholders* e os requisitos da organização quando forem definir a estratégia do sistema.
- O sistema responderá mais facilmente a mudanças em seus usuários (suas necessidades, tarefas, contexto etc).
- O empreendimento responderá mais facilmente a mudanças em seus usuários.
- Será menos provável que o sistema seja rejeitado pelo mercado.

Esse processo compreende as seguintes práticas:

- Representar o usuário final.
- Coletar a inteligência do marketing.
- Definir e planejar uma estratégia de sistema.
- Coletar o *feedback* do marketing.
- Analisar as tendências do usuário.

HCD 2: PLANEJAR E GERENCIAR O PROCESSO HCD

O propósito do processo HCD 2 é especificar com as atividades centradas no humano se encaixam no empreendimento e em todo o processo de ciclo de vida do sistema. Como resultado de uma bem sucedida implementação desse processo teremos que:

- O plano de projeto permitirá a interação e incorporação do feedback do usuário.
- Recursos serão alocados para comunicação eficiente entre os participantes do time de projeto.
- Potenciais conflitos e *trade-offs* entre questões centradas no humano e outras serão reconciliados.
- Processos centrados no humano serão incorporados em sistemas de qualidade, procedimentos e padrões.

Esse processo compreende as seguintes práticas:

- Consultar *stakeholders*.
- Identificar e planejar o envolvimento do usuário.
- Selecionar métodos e técnicas centrados no usuário.
- Assegurar que o time de projeto segue uma abordagem centrada no humano.
- Planejar as atividades centradas no humano.
- Gerenciar as atividades centradas no humano.
- Premiar a abordagem centrada no humano.
- Prover suporte ao projeto centrado no humano.

HCD 3: ESPECIFICAR OS STAKEHOLDERS E OS REQUISITOS ORGANIZACIONAIS

O propósito do processo HCD 3 é estabelecer os requisitos da organização e outros grupos de interesse do sistema. Esse processo leva em conta completamente as necessidades, competências e o ambiente de trabalhos de cada stakeholder relevante do sistema. Como resultado de uma bem sucedida implementação desse processo teremos:

- A necessidade de performance do novo sistema mediante objetivos operacionais e funcionais.
- Requisitos legais ou legislativos relevantes.
- Cooperação e comunicação entre usuários e outros grupos relevantes.
- Os trabalhos do usuário (incluindo alocação de tarefas, conforto do usuário, segurança, saúde e motivação).
- Tarefa sobre a performance do usuário ao utilizar sistema.
- Projeto de trabalho e práticas e estruturas organizacionais.
- Viabilidade de operação e manutenção.
- Objetivos para a operação e/ou uso dos componentes de software e hardware do sistema.

Esse processo compreende as seguintes práticas:

- Esclarecer e documente as metas do sistema.
- Definir stakeholders.
- Impor riscos aos stakeholders.
- Definir o sistema.
- Gerar os requisitos dos stakeholders e da organização.
- Definir a qualidade do uso.

HCD 4: ENTENDER E ESPECIFICAR O CONTEXTO DE USO

O propósito do processo HCD 4 é identificar esclarecer e armazenar as características dos stakeholders, suas tarefas e o ambiente físico organizacional no qual o sistema irá operar. Como resultado de uma bem sucedida implementação desse processo teremos:

- As características do usuário final.
- As tarefas que os usuários irão realizar.
- A organização e o ambiente no qual o sistema é usado.

Esse processo compreende as seguintes práticas:

- Identificar e documentar as tarefas do usuário.
- Identificar e documentar atributos significantes do usuário.
- Identificar e documentar o ambiente organizacional.
- Identificar e documentar o ambiente técnico.
- Identificar e documentar o ambiente físico.

HCD 5: PRODUZIR SOLUÇÕES DE PROJETO

O propósito do processo HCD 5 é criar potenciais soluções de projeto utilizando como fonte de suporte práticas já estabelecidas do estado da arte, a experiências e conhecimento dos participantes e os resultados do contexto da análise do uso. Como resultado de uma bem sucedida implementação desse processo teremos que:

- Todo o sistema social-técnico no qual qualquer componente técnico opere será considerado no projeto.
- Características e necessidades do usuário serão levadas em consideração na compra de componentes do sistema.
- Características e necessidades do usuário serão levadas em consideração no projeto do sistema.
- Conhecimento existente sobre as melhores práticas da engenharia de sistemas social-técnica, ergonomia, psicologia, ciência cognitiva e outras disciplinas relevantes serão integrados aos sistema.
- A comunicação entre os stakeholders do sistema será melhorada porque as decisões de projeto serão mais explícitas.

- O tipo de desenvolvimento será capaz de explorar vários conceitos de projeto antes de se decidir por um.
- Feedback do stakeholder e do usuário final serão incorporados no projeto mais cedo no processo de desenvolvimento.
- Será possível validar várias iterações de um projeto e de projetos alternativos.
- A interface entre o usuário e os componentes de software, hardware e organizacionais do sistema será projetada.
- O treinamento e o suporte dos usuários serão desenvolvidos.

Esse processo compreende as seguintes práticas:

- Alocar funções.
- Produzir um modelo de tarefas composto.
- Explorar o projeto do sistema
- Use conhecimento existente para desenvolver soluções de projeto.
- Especificar o sistema.
- Desenvolver protótipos.
- Desenvolver o treinamento do usuário.
- Desenvolver o suporte ao usuário.

HCD 6: VALIDAR O PROJETO MEDIANTE OS REQUISITOS

O propósito do processo HCD 6 é coletar o feedback do projeto em desenvolvimento. Esse feedback será coletado de usuários finais e outras fontes representativas. Como resultado de uma bem sucedida implementação desse processo teremos que:

- O feedback será provido para melhorar o projeto.
- Será avaliado se os objetivos dos stakeholders e organizacionais foram atingidos.
- Uso prolongado do sistema será monitorado.

No caso de validações para identificar melhoramentos no sistema (validação formativa), uma bem sucedida implementação do processo refletira:

- Problemas e escopos potenciais para o melhoramento: na tecnologia, no suporte de material, no ambiente físicos ou organizacional e no treinamento.
- Que opção de projeto melhor se encaixa nos requisitos funcionais e do usuário.

- Feedback e requisitos posteriores do usuário.

No caso de validações para avaliar se os objetivos foram atingidos (validação aditiva), uma bem sucedida implementação do processo demonstrará:

- Quão bem o sistema atinge suas metas organizacionais
- Que um projeto particular satisfaz requisitos centrados no humano.
- Conformidade com requisitos internacionais, nacionais e/ou legais.

Esse processo compreende as seguintes práticas:

- Especificar e validar o contexto de validação.
- Validar protótipos recentes a fim de definir requisitos para o sistema.
- Validar protótipos a fim de melhorar o projeto.
- Validar o sistema a fim de checar se os requisitos do sistema foram satisfeitos.
- Validar o sistema a fim de checar se as práticas necessárias foram seguidas.
- Validar o sistema em uso a fim de assegurar que ele continuar a satisfazer necessidades organizacionais e do usuário.

HCD 7: INTRODUZIR E OPERAR O SISTEMA

O propósito do processo HCD 7 é estabelecer os aspectos humanos do suporte a da implementação do sistema. Como resultado de uma bem sucedida implementação desse processo teremos que:

- As necessidades dos stakeholders do sistema serão comunicadas ao projeto.
- O controle de mudanças, incluindo as responsabilidades dos usuários e desenvolvedores, será especificado.
- Os requisitos de suporte a usuários finais, mantedores e outros stakeholders serão endereçados.
- Haverá obediência a procedimentos de saúde e segurança.
- Haverá suporte a customização local do sistema.
- As reações do usuário serão coletadas e as mudanças resultantes do sistema serão reportadas aos stakeholders.

Esse processo compreende as seguintes práticas:

- Controle de mudanças

- Determinar o impacto na organização e nos stakeholders.
- Projeto local e customizado.
- Entregar o treinamento do usuário.
- Suporte aos usuários nas atividades planejadas.
- Garantir conformidade com a legislação sobre a ergonomia do ambiente de trabalho.

Apêndice C: Pesquisa sobre a Cultura de Usabilidade Local

Introdução

Definição do Problema

Em três anos de atuação, o Porto Digital conseguiu reunir 68 empresas. Juntas, elas geram 1600 empregos diretos, 90% dos quais para profissionais graduados. Apesar da presença de multinacionais como IBM, Motorola e Microsoft, a maioria das empresas que compõem tal cenário é de pequeno e médio porte, muitas das quais em processo de incubação.

Sendo assim, por se encontrarem no estágio inicial de sobrevivência em seu ciclo de vida, essas empresas ainda enfrentam algumas dificuldades para se firmarem no mercado. Uma solução largamente utilizada é maximizar a produção de software, mesmo com orçamentos reduzidos e cronogramas apertados.

É evidente que a adoção de tal método tende a comprometer a qualidade dos produtos gerados, pois as empresas não estão dispostas a investir recursos e tempo demais em seus projetos. Isto é ainda mais grave no que se refere ao quesito de qualidade de software usabilidade, dada a incipiente cultura nesse campo de estudo por parte das empresas do Porto Digital. Muitos gerentes acreditam que a aparente demora pelos benefícios e real eficácia da utilização de técnicas de usabilidade entravam o investimento de mais recursos na área em seus processos de desenvolvimento.

Exposto o problema, a pesquisa buscou colher e analisar dados mais precisos sobre a cultura de usabilidade dentro do Porto Digital. O objetivo de tal estudo foi obter uma base de conhecimento real para a fundamentação de propostas sobre como adequar e inserir aspectos da interação homem-computador na realidade de empresas locais.

Projeto de Pesquisa

Identificação das Questões de Pesquisa

Definido o escopo do problema da pesquisa, pode-se, então, ser dividi-lo nos seguintes questionamentos básicos:

- Qual a interação das empresas com seus usuários?
    - Há uma área ou pessoa específica?
    - Em que fase do projeto ocorre?
    - Qual a finalidade?
    - O que ocorre? Quais as atividades?
    - Que artefatos são produzidos?
    - Há a utilização de ferramentas?
- Qual o interesse da empresa na usabilidade de seus produtos?
    - Que tempo e custo seriam viáveis?
    - Qual a melhor forma da empresa trabalhar com usabilidade? Através de especialista, treinamento geral ou contrato de terceiros?
    - De que forma se sugere a utilização de usabilidade? De forma iterativa ou apenas no final do projeto?

Definição da Amostra

Dada a quantidade de empresas (um total de 68) que fazem parte do Porto Digital, o ambiente de estudo considerou um espaço amostral com 9 empresas das mais representativas como suficiente para garantir a validade e corretude dos resultados da pesquisa. Através do auxílio de contatos dentro dessas empresas (identificados e contatados), o projeto avaliou as seguintes empresas:

- Capital Login
- Casullo
- CESAR
- Facilit
- Fundição
- Jynx
- Meantime
- Motorola
- Qualiti

Técnicas de Coleta de Dados

Para a coleta dos dados, foi utilizada uma entrevista semi-estruturada. Trata-se de um roteiro preliminar de perguntas que se molda à situação concreta da entrevista, já que o entrevistador tem liberdade de acrescentar novas perguntas a esse roteiro. A vantagem desse tipo de técnica é a possibilidade de aprofundar e clarificar os pontos que se considere relevantes aos objetivos do estudo.

Definição da Técnica de Análise de Dados

Os dados qualitativos obtidos a partir das entrevistas semi-estruturadas e de anotações dos observadores apresentaram-se na forma de descrições narrativas, sendo, posteriormente, analisados. Buscaram-se similaridades entre as informações fornecidas por cada gerente, o que pôde revelar aspectos em comum sobre a cultura e a atenção aos aspectos humanos nos processos de desenvolvimento das empresas.

Resultados das Entrevistas

Capital Login

Empresa

- Nome: *Capital Login*
- Área de Atuação: *consultoria em TI, suporte técnico (físico e soluções Microsoft) e fábrica de software (web, .NET).*
- Tempo de Atuação: *10 anos*
- Tamanho: *média*
- Equipe: *~300 pessoas (Brasil), sendo >100 pessoas em Recife.*
- Duração Média dos Projetos: *3 meses*

Entrevistado

- Nome: *André Salgado Barhum*
- Idade: *27 anos*
- Função: *Design/Programador da Interface*
- Tempo na Empresa: *2 anos e 10 meses*
- Carga Horária: *8hs*

Empresa e Usuários

A área de fábrica de software trabalha, atualmente, sob demanda de Clientes.

Durante a reunião para definição de requisitos, o entrevistado captura também requisitos da interface e informações sobre os usuários e suas tarefas. Entretanto, durante a fase de testes, ocorrem apenas testes com pessoas internas à equipe de desenvolvimento (funcionais e de usabilidade), ocorrendo apenas um caso (cliente Claro), onde houve a interação com os usuários.

Depois de implantado, às vezes, é passado um questionário de satisfação (uso geral e de usabilidade).

Usabilidade

Como cita o entrevistado, usabilidade é dar a solução mais simples possível, fazendo com que o usuário não pense no software, mas na tarefa.

Há o interesse de utilizar ferramentas para captura das ações dos usuários no software.

A Usabilidade é considerada como um conceito abstrato, uma forma de tornar o software intuitivo e melhorar sua interação com o usuário, aumentando sua produtividade e, conseqüentente, deixando o cliente satisfeito (o que dá abertura à empresa produtora ao mercado).

A forma de terceirização sugerida é através da interação com os desenvolvedores através de uma pessoa da empresa de consultoria no local e apoio pelo restante desta para avaliações, havendo também a conscientização/treinamento da equipe da empresa de desenvolvimento.

- *"É uma coisa que tem o seu preço, mas também tem seu lucro".*
- *"Usabilidade é um diferencial".*

Casullo

Empresa

- Nome: *Casullo Comuncação e Design*
- Área de Atuação: *Comunicação digital, E-mail Marketing, Usabilidade, Mídias Educativas e Marketing Político-Digital*

- Tempo de Atuação: *02 anos*
- Tamanho: *micro-empresa*
- Equipe: *05 colaboradores e 4 sócios – total de 09 pessoas*
- Duração Média dos Projetos: *entre 2 e 12 meses.*

Entrevistado

- Nome: *Klaus Hachenburg*
- Idade: *32 anos*
- Função: *Diretor de Negócios e Marketing*
- Tempo na Empresa: *02 anos*
- Carga Horária: *full time*

Empresa e Usuários

- Existe relação empresa-usuário? *Sim.*
- Há uma área ou pessoa específica? *Havia uma pessoa com conhecimento em usabilidade.*
- Em que fase do projeto ocorre? *Na fase de arquitetura da informação, antes da prototipação e na fase de desenvolvimento da proposta gráfica*
- Qual a finalidade? *Entender o perfil e hábitos de trabalho, pesquisa, compra, navegação, desejos de consumo e forma de entendimento do computador em suas atividades.*
- O que ocorre? Quais as atividades? *Levantamento de requisitos, avaliação heurística.*
- Artefatos são produzidos? *Desenvolvimento de guidelines, projetos gráficos e navegação.*
- Utilização de ferramentas? *Métodos e práticas de design.*

Usabilidade

- Tempo: *A definir*
- Custo: *A orçar*

- Qual a melhor forma de sua empresa trabalhar com usabilidade? Através de especialista, treinamento geral ou contrato de terceiros? *Através de especialistas nos ajudando na implantação do serviço para outras empresas.*
- De que forma você sugere a utilização de usabilidade? De forma iterativa ou apenas no final do projeto? *Pode ocorrer em diversas fases do projeto, desde a concepção até a entrega final ao cliente/usuário.*

CESAR

Empresa

- Nome: *Centro de Estudos Avançados do Recife (CESAR)*
- Área de Atuação: *TI, sendo incubadora e desenvolvedora de sistemas.*
- Tempo de Atuação: *9 anos*
- Tamanho: *média~grande, sendo uma ONG*
- Equipe: *~450 pessoas (Recife e São Paulo)*
- Duração Média dos Projetos: *~ 1 ano, não sendo menos de 6 meses*
- Projetos Atuais: *~30 projetos*

Entrevistado

- Nome: *José Carlos Porto Arcoverde Jr. (Mabuse)*
  Ubirajara Lucena Pereira da Silva Jr. (Bira)
- Idade: *33 anos*
  27 anos
- Função: *Gerente de Design*
  Pesquisador de Design, Bolsista
- Tempo na Empresa: *desde 1999 (6 anos)*
  desde 2004 (~1 ano)
- Carga Horária: *8 horas*
  6 horas

Empresa e Usuários

De acordo com os entrevistados, a empresa não possui uma interação com o usuário especificada no processo de desenvolvimento, ocorrendo casos isolados e raros onde o Cliente solicita a avaliação da interface junto ao usuário (ex: Intelbrás).

Mesmo o levantamento de requisitos, que conta com a definição tanto de requisitos de programação como de design, baseia-se apenas na opinião do Cliente. Levando em conta, ainda, que dados como público-alvo e objetivos do Cliente são obtidos pela equipe de marketing e não há uma preocupação de interar tanto a equipe de design como a de desenvolvimento em aspectos relevantes aos usuários.

Desde 1999, a equipe de Design trabalha na criação de padrões baseados em componentes. Isso significou mudar a visão que as equipes de desenvolvimento e engenharia de software possuíam do trabalho de um designer. Atualmente, a equipe de desenvolvimento possui o conhecimento desses padrões e ocorrem avaliações periódicas (mensais) de interface feitas pelos designers da empresa. Entretanto, ainda fica a questão de como funciona a interação do design com os analistas e engenheiros de software.

Atualmente, ocorrem prototipações do sistema (devido ao processo iterativo adotado na empresa) através da linguagem HTML, havendo a idéia de passar desse método para o de prototipação em papel.

Dois casos onde ocorreu ou ocorre a participação de usuários foram citados: (a) Bompreço, onde os usuários estavam próximos e acessíveis, interados ao desenvolvimento do produto por apoio do Cliente e (b) Motorola, onde o usuário final é o comprador direto e sua satisfação influencia bastante em sua escolha de compra e o próprio processo de desenvolvimento envolve a participação desse.

Usabilidade

Um dos problemas apontados sobre a utilização de usabilidade relaciona-se ao fato de que a equipe de design sugere e atesta a importância da participação de usuários na elaboração e teste do sistema, entretanto, o Cliente não se dispõe a investir financeiramente nesse aspecto.

O caminho apontado para a valorização da usabilidade e interação com usuários foi o de Qualidade de software, inserindo aspectos de usabilidade no processo interno da empresa. No início, passando por terceirização para, depois, desenvolver esse conhecimento dentro da empresa. Esse conhecimento deverá ser partilhado entre toda a equipe envolvida no desenvolvimento e mais concentrada em uma equipe específica, a fim de tirar dúvidas e tomar a frente das atividades relacionadas à usabilidade. É de interesse geral justificar mudanças feitas no processo devido à inserção de usabilidade, mesmo que pequenas, para que toda a equipe compreenda sua importância no produto final e o papel de cada indivíduo nessa conquista.

Facilit

Empresa

- Nome: *Facilit*
- Área de Atuação: *produtos Borland (treinamento, venda de produtos de consultoria) e Fábrica de Software.*
- Tempo de Atuação: *10 anos.*
- Tamanho: *pequena*
- Equipe: *15 pessoas, sendo 9 de desenvolvimento (Fábrica).*
- Duração Média dos Projetos: *8 meses.*
- Projetos Atuais: *Datalegis e Justiça.*

Entrevistado

- Nome: *Roni*
- Idade: *27 anos*
- Função: *Gerente de Tecnologia (Fábrica).*

- Tempo na Empresa: *3 anos.*
- Carga horária: *9~10 horas diárias.*

Empresa e Usuários

Geralmente, os projetos surgem apenas para implementação, mas em sua maioria, cabe à empresa o papel de desenvolver tudo desde o escopo.

A interação com o usuário ocorre em dois momentos: durante a definição do escopo (através de entrevistas) e durante a definição dos requisitos (também por entrevistas). Esse processo pode ser resumido:

1. *Conversa com Cliente*
2. *Entrevista com Usuários para definição do escopo do projeto*
3. *Aceitação do Cliente*
4. *Entrevista com Usuários para definição dos requisitos do software*
5. *Aceitação do Cliente, havendo a decisão entre requisitos conflitantes (usuários com opiniões distintas sobre algum aspecto).*
6. *Definição dos Casos de Uso.*

A interação com o usuário durante o desenvolvimento e após finalização do produto só ocorre por intervenção do Cliente, ficando a cargo deste a decisão de fazê-lo.

A hipótese que pode ser levantada é a de que, em empresas onde os produtos surgem de necessidades específicas do Cliente (solicitação do Cliente), a usabilidade só é considerada caso esteja dentre as exigências do Cliente. Ou seja, o incentivo à busca de usabilidade no produto deve vir do Cliente.

O último projeto de produto de software advindo de iniciativa da própria empresa, o Communis, teve sua 2a versão no mesmo momento da entrada do Entrevistado na empresa. Essa versão possuía as mesmas funcionalidades da versão anterior, melhorando a forma de implementação. O Entrevistado não soube dar informações detalhadas de como as funcionalidades, e o projeto em si, foram determinados dentro da empresa.

As interfaces dos softwares produzidos na empresa são elaboradas através de terceiros (empresa de design), sendo responsabilidade da Facilit fornecer a arquitetura da informação.

- *"Usuário não sabe o que quer e muda de opinião constantemente".*
- *"Usabilidade deve melhorar a satisfação do Cliente. Podendo ser através do retorno dos usuários".*

Fundição

Empresa

- Nome: *Fundição*
- Área de atuação: *usabilidade, design de interfaces e tipografia. Sendo interfaces de software e websites.*
- Tempo de atuação: *2 anos.*
- Equipe: *5 pessoas, sendo todos da área de design.*
- Surgimento da Empresa: *diretoria vinda da antiga empresa Mobile, sendo o entrevistado responsável pelas interfaces de palms.*
- Tamanho da Empresa: *micro.*
- Duração Média dos Projetos: *cerca de 3 meses.*

Entrevistado

- Nome: *Buggy*
- Idade: *28 anos*
- Função: *Prospect de Clientes.*
- Tempo na Empresa: *2 anos.*
- Carga Horária: *12 horas diárias.*

Empresa e Usuários

A empresa possui, atualmente, uma pessoa responsável pela usabilidade, Eduardo Cavalcante; sendo esta apoiada pelo restante da equipe (em especial,

Bosco e Paulo). Antes dessa especialização na área, esse serviço era fornecido através de terceiros (professores, especialistas etc.). Hoje, toda a equipe da empresa possui conhecimento geral sobre usabilidade, podendo colaborar em avaliações.

Exemplos de métodos bastante utilizados pela empresa são:

- *Heurísticas - considera não só o problema (Nielsen), mas o usuário e suas tarefas.*
- *Percurso cognitivo*
- *Análise da tarefa*
- *Teste de usabilidade - em laboratórios de marketing (pagos), havendo filmagem, captura de tela, parecer de avaliadores que observam incógnitos e apoio de um moderador (acompanha de perto o usuário).*

Atualmente a empresa conta com 4 'pacotes' compostos pela otimização de 7 métodos de avaliação de usabilidade utilizados por essa.

Os documentos utilizados e produzidos são, principalmente, formulários (questionários) e Relatório de Avaliação. Sendo capturados dados tanto quantitativos quanto qualitativos. A justificativa, tanto da otimização dos métodos quanto dos tipos de dados se dá pela constatação pela empresa de que o mercado se distingue bastante da universidade, havendo restrições de tempo e custo e a necessidade de dados mais numéricos e objetivos para o Cliente.

Uma das ferramentas utilizadas para prototipação é a Freehand.

Os custos do serviço de usabilidade são contados por hora de trabalho, sendo o mesmo valor do custo de hora por desenvolvimento.

Inicialmente a demanda foi somativa (softwares prontos), mas houve o surgimento da demanda formativa (antes do desenvolvimento, através do documento de requisitos), fazendo-se o uso de teste de protótipos com usuários.

Três formas foram apontadas para a obtenção de Clientes nessa área pela empresa: clientes de projetos gráficos anteriores, indicação ou busca da Fundição pelo cliente.

- *"Usabilidade enfrenta problema de falta de apoio institucional".*
- *"É um problema inserir usabilidade tentando estabelecer uma nova cultura de produção. Já que uma mudança de processo é custosa. A terceirização, principalmente no primeiro momento, é uma opção mais barata."*

Jynx

Empresa

- Nome: *Jynx*
- Área de Atuação: *jogos (mais plataforma) e videogames (futuro).*
- Tempo de Atuação: *5 anos.*
- Tamanho: *?*
- Equipe: *20 pessoas*
- Duração Média dos Projetos: *5 anos (jogo Futsim) a 1 mês ou 15 dias ('jogos sérios' - publicitários, treinamento; Flash ou C++).*

Entrevistado

- Nome: *Reginaldo Valadares*
- Idade: *30 anos*
- Função: *sócio, gerente de desenvolvimento e gerente de projetos (3 atualmente).*
- Tempo na Empresa: *5 anos*
- Carga Horária: *10hs.*

Empresa e Usuários

Em projetos maiores, há a validação junto aos usuários antes da entrega. O mesmo não ocorre em projetos menores, confiando-se na capacidade da equipe de encontrar problemas na interface.

Quando ocorrem testes de usabilidade, é definido, mesmo que superficialmente, um projeto de teste, definindo cronograma, pessoas responsáveis etc. Isso ocorreu em jogos como o Futsim (2 ou 3 vezes, capturando também o movimento do mouse e gravando vozes dos usuários) e o Jogo do Empreendedor (utilização por 300 pessoas que participavam de uma feira de TI, capturando dados por questionário).

Não há ninguém específico responsável nessa atividade. Esses testes geralmente são acompanhados pelo Game Designer e Gerente de Projeto, tendo, o segundo, o papel de organizar. Os testes com usuários reais são presenciais, ocorrendo também testes internos com game designers e desenvolvedores de outros projetos da empresa.

O entrevistado citou a diferença entre jogos e softwares comuns, pois no jogo não é o Cliente que define os requisitos, sendo o Game Designer o responsável por tomar as decisões de projeto. Nos jogos também há uma necessidade maior por facilidade em uso, interface agradável, user friendly etc.

Usabilidade

A usabilidade foi descrita como facilidade de uso do software, levando em conta algumas características como: eficiência, eficácia, satisfação e o 'ser intuitivo'.

Meantime

Empresa

- Nome: *Meantime*
- Área de Atuação: *informação e entretenimento para celular (mais jogos). Sendo projetos por demanda (Cliente), bem como de iniciativa própria - o primeiro mais freqüente.*
- Tempo de Atuação: *2 anos, tendo sido incubada a cerca de 1 ano. Antes possuía o papel de testar celulares da Motorola, utilizando para isso, jogos.*
- Tamanho: *(?) incubada*
- Equipe: *30~40 pessoas*
- Duração Média dos Projetos: *3 meses, sendo 3 semanas para testes.*
- Projetos Atuais: *BBBrasil, Zaak etc.*

Entrevistado

- Nome: *Alexandre Damasceno*
- Idade: *27 anos*
- Função: *atualmente, Game Design Chefe; tendo sido engenheiro e arquiteto de software.*
- Tempo na Empresa: *~2anos.*
- Carga Horária: *8~9hs*

Empresa e Usuários

Os usuários interagem com a empresa na fase de testes, quando há uma versão mais estável, permitindo Beta testers e testes pela equipe interna da empresa. Após os testes funcionais pela empresa, os beta-testers são solicitados a utilizar o sistema e responder a um questionário (satisfação e outros aspectos considerados relevantes ao tipo de jogo).

As pessoas que trabalham nessa fase, junto ao usuário, são os game designers e pessoas de qualidade. Esses testes são acompanhados pessoalmente a fim de capturar a reação do usuário.

Usabilidade

O problema apontado pelo entrevistado é o de coletar pessoas e seus respectivos dados.

Motorola

Empresa

- Nome: *Motorola Recife*
- Área de Atuação: *software para celular, middleware e motocoder (web-site para desenvolvedores). Sendo criada para criação de jogos para celular junto à empresa Meantime (jogo 3D e 2D).*
- Tempo de Atuação: *?*
- Tamanho: *grande*
- Equipe: *~100 pessoas em Recife*
- Duração Média dos Projetos: *contratos renovados anualmente, sendo jogos demorando cerca de 3 meses.*

Entrevistado

- Nome: *Saulo Dourado*
- Idade: *28 anos*
- Função: *líder da equipe de design.*
- Tempo na Empresa: *~1ano.*
- Carga Horária: *8hs.*

Empresa e Usuários

Apenas testes com desenvolvedores internos e feedback do sponsor (Cliente). Não há conhecimento se a opinião do Cliente é baseada em alguma pesquisa ou envolvimento com usuários.

Como o entrevistado citou, a pouca interação com usuário deve-se a uma falta de cultura tanto da empresa como do Cliente.

Qualiti

Empresa

- Nome: *Qualiti*
- Área de Atuação: *consultoria (mais processos), treinamento, venda de ferramentas (Rational) e desenvolvimento de ferramentas (Coder, Metrics e Audit).*
- Tempo de Atuação: *~ 4 anos*
- Tamanho: *pequena*
- Equipe: *22 pessoas*
- Duração Média dos Projetos: *2 anos*
- Projetos Atuais: *Coder, Metrics e Audit. Sendo Coder uma ferramenta de geração de código, o Metrics de coleta de métricas de código e o Audit, para auditoria de código.*

Entrevistado

- Nome: *Livar Correia*
- Idade: *21 anos*
- Função: *estagiário, passando por desenvolvedor e atualmente trabalhando com cursos de treinamento.*
- Tempo na Empresa: *1 ano*
- Carga Horária: *4hs*

Empresa e Usuários

De acordo com o entrevistado, apenas em um caso ocorreu a real utilização do sistema por usuários. Esta ocorreu após um treinamento com a empresa CSI, que levou em consideração a sugestão da Qualiti de utilizar o software Audit para avaliar a qualidade do código da equipe em treinamento. Fora esse caso, todos os testes são feitos por pessoas internas à empresa.

Atualmente, o esboço da interface é produzido antes da codificação, através dos casos de uso especificados.

Usabilidade

O interesse em usabilidade foi relacionado à obtenção de feedback do usuário para elaboração de novas versões e captura de novos requisitos.

A ausência de um trabalho maior junto ao usuário já foi discutido em reuniões da empresa, mas deixado para segundo plano devido ao surgimento de outras urgências.

A usabilidade é vista como uma forma de trazer o produto para mais perto da realidade do mercado.

Atividades relevantes nesse intuito seriam as de:

- *analisar as tarefas do usuário (como faz, melhorias, causar menor impacto na tarefa, aumentar eficiência e ser de fácil entendimento);*
- *definir interfaces no meio do processo (protótipos);*
- *avaliar a interface.*

Considerações Finais

Através dessa pesquisa, pôde-se observar que são poucas as empresas que trabalham junto aos usuários finais de seus produtos. Muitas vezes, a produção por demanda limita essas empresas a fazer apenas o que foi especificado, levando a problemática de cultura de usabilidade ao nível do Cliente. E, pelo processo natural do mercado, parte desses clientes só focará sua atenção para essa característica de qualidade quando seus usuários a exigirem, fazendo com que sua ausência signifique perda de lucros.

Em empresas de grande porte, a solução de treinamento da equipe de desenvolvimento em valores básicos de usabilidade e formação de um núcleo de pessoas especialistas em trabalhar sobre a interação com os usuários mostra-se viável. Já em empresa menores, isso se torna particularmente inviável caso essa área não seja o foco da empresa, levando em conta os custos. Nesse caso, a opção por terceirizar atividades como o acompanhamento do processo e avaliação dos

protótipos e da interface final junto aos usuários melhora o custo-benefício nessas empresas.