

**XPU: UMA EXTENSÃO DO MÉTODO XP
VISANDO À USABILIDADE**

THIAGO ROCHA SILVA

**XPU: UMA EXTENSÃO DO MÉTODO XP
VISANDO À USABILIDADE**

Dissertação apresentada ao Programa de Pós-Graduação em Ciências da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciências da Computação.

ORIENTADOR: CLARINDO ISAÍAS PEREIRA DA SILVA E PÁDUA

Belo Horizonte

Julho de 2009

© 2009, Thiago Rocha Silva.
Todos os direitos reservados.

Silva, Thiago Rocha
S586x XPU: uma extensão do método XP visando à
usabilidade / Thiago Rocha Silva. — Belo Horizonte,
2009
xxii, 110 f. : il. ; 29cm

Dissertação (mestrado) — Universidade Federal de
Minas Gerais
Orientador: Clarindo Isaías Pereira da Silva e Pádua

1. Engenharia de Software. 2. Engenharia de
Usabilidade. 3. Métodos Ágeis. I. Título.

CDU 519.6*32



UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

XPu: uma extensão do método XP visando à usabilidade

THIAGO ROCHA SILVA

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:

PROF. CLARINDO ISAÍAS PEREIRA DA SILVA E PÁDUA - Orientador
Departamento de Ciência da Computação - UFMG

PROFA. ANA LIDDY CENNI DE CASTRO MAGALHÃES
Universidade FUMEC

PROF. RODOLFO SÉRGIO FERREIRA DE RESENDE
Departamento de Ciência da Computação - UFMG

Belo Horizonte, 03 de julho de 2009.

À Maria de Fátima Rocha com toda a minha admiração!

Agradecimentos

Esta seção é mais difícil de escrever do que muitos capítulos. São muitas pessoas a agradecer, algumas diretamente ligadas ao desenvolvimento deste trabalho e outras que, mesmo indiretamente, foram indispensáveis na construção do conhecimento necessário para chegar até aqui.

Agradeço imensamente ao professor Clarindo I. P. S. Pádua por aceitar orientar este trabalho, mesmo depois de tantas indefinições iniciais e das minhas diversas ausências nas reuniões semanais do nosso grupo de pesquisa. Agradeço também aos professores Rodolfo S. F. Resende e Ana Liddy C. C. Magalhães que participaram da minha banca examinadora e contribuíram de maneira muito positiva para a melhoria dos resultados apresentados.

Aos meus pais pelo suporte financeiro, emocional e por sempre acreditarem nos meus ideais e nos meus objetivos, me dando o carinho e o amor necessários para seguir adiante.

Ao Felipe pelo apoio nos primeiros meses do mestrado e aos colegas Hércules Batista e Amanda Ramos por aceitarem participar de maneira tão gentil e prestativa do estudo de caso realizado, sempre solícitos e disponíveis durante suas férias em horários nem sempre agradáveis.

Aos demais amigos e colegas do grupo de pesquisa em Engenharia de Software do DCC/UFMG pelas discussões, críticas e sugestões de melhoria deste trabalho. Aos funcionários do departamento pelo atendimento sempre tão ágil e gentil todas as vezes em que precisei.

Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e à Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG) pelo apoio financeiro ao longo do mestrado.

Por fim, agradeço de maneira muito especial ao criador, ao meu anjo da guarda e aos meus guias espirituais pela energia, pela proteção, pela força e pela oportunidade de viver e de cada vez mais conquistar todos os meus objetivos.

A todos vocês, o meu mais sincero muito obrigado!

*“If everything’s under control,
you’re going too slow.”*
(Mario Andretti)

Resumo

Os métodos ágeis têm recebido grande atenção da comunidade de desenvolvimento de software nos últimos anos. Esses métodos propõem uma alternativa ao desenvolvimento de software dirigido por processos em prol de um desenvolvimento com foco nas necessidades dos clientes e das pessoas envolvidas, na entrega constante de software funcional, no desenvolvimento iterativo baseado em ciclos curtos de entrega e na aceitação da constante mudança dos requisitos ao longo do desenvolvimento. Por outro lado, os processos de usabilidade especificam práticas ao longo do ciclo de vida de desenvolvimento que contribuem para a qualidade da interface final no que se refere à usabilidade do produto. No entanto, os métodos ágeis não tratam das práticas relacionadas à usabilidade. Diante desse cenário, este trabalho propõe um método denominado XPu. O XPu foi proposto como uma extensão do método XP e tem como principal objetivo tratar da usabilidade de produtos de software através de uma disciplina ágil. Resultados de uma inspeção e de um estudo de caso comparando o desenvolvimento de um produto utilizando o método XP e o método XPu sugerem um considerável ganho de qualidade na interface final a um custo de desenvolvimento relativamente baixo quando o XPu é utilizado.

Palavras-chave: Engenharia de Software, Engenharia de Usabilidade, Métodos Ágeis.

Abstract

Agile methods have been received a great attention from the software development community in the last years. These methods propose an alternative for process driven software development on behalf of development with focus on the customer and the others stakeholders needs, on the constant delivery of functional software, on the iterative development based on delivery short cycles and on the acceptance of constant change of requirements through the development. On the other hand, usability processes specify practices through the development cycle that contribute for the end interface quality regarding to product usability. However, the agile methods do not treat practices related to usability. Because of that, this work proposes a method called XPu. The XPu was proposed as a XP extension and it has as main goal to deal with software products usability through an agile discipline. Results of an inspection and a case study comparing the development of a product using the XP and the XPu methods suggest a considerable quality improvement in the end interface with a relatively low development cost when XPu is used.

Keywords: Software Engineering, Usability Engineering, Agile Methods.

Lista de Figuras

2.1	Ciclo de vida em estrela (adaptado de [Hix & Hartson, 1993])	20
2.2	Ciclo de vida do modelo ISO/IEC 13407.	21
2.3	Disciplina de Usabilidade do Praxis-u.	23
2.4	Atividades de UED e iterações ágeis.	28
4.1	Visão de Projeto do XPu e sua integração com o XP - Parte 1.	43
4.2	Visão de Projeto do XPu e sua integração com o XP - Parte 2.	44
4.3	Exemplo de História de Usabilidade.	48
4.4	Visão de Iteração do XPu e sua integração com o XP - Parte 1.	53
4.5	Visão de Iteração do XPu e sua integração com o XP - Parte 2.	54
4.6	Visão de Desenvolvimento do XPu e sua integração com o XP - Parte 1.	57
4.7	Visão de Desenvolvimento do XPu e sua integração com o XP - Parte 2.	57
5.1	História de Usuário 1 - XP.	73
5.2	História de Usuário 2 - XP.	73
5.3	Tela de Busca gerada utilizando o método XP.	74
5.4	Tela de Login gerada utilizando o método XP.	75
5.5	Tela de Cadastro de Cliente gerada utilizando o método XP.	75
5.6	Tela de Comprovante de Locação gerada utilizando o método XP.	76
5.7	História de Usuário 1 - XPu.	77
5.8	História de Usuário 2 - XPu.	77
5.9	<i>Template</i> gerado utilizando o método XPu.	79
5.10	Protótipo gerado utilizando o método XPu.	80
5.11	Tela de Busca gerada utilizando o método XPu.	80

Lista de Tabelas

5.1	Tempo consumido pelas atividades relativas ao método XPu	81
5.2	Avaliação da tela de busca gerada pelo método XP	82
5.3	Avaliação da tela de busca gerada pelo método XPu	83
5.4	Comparativo dos resultados para os métodos XP e XPu	83
6.1	Mapeamento das atividades do Praxis-u para o XPu.	89
6.2	Atividades Técnicas do XPu - Insumos e Artefatos.	89
6.3	Atividades Gerenciais do XPu - Insumos e Artefatos.	89
B.1	Legenda de Atividades Técnicas do XPu.	103
B.2	Legenda de Atividades Gerenciais do XPu.	103
B.3	Legenda de Insumos e Artefatos do XPu.	104
B.4	Legenda de Atividades do XP.	104

Sumário

Agradecimentos	ix
Resumo	xiii
Abstract	xv
Lista de Figuras	xvii
Lista de Tabelas	xix
1 Introdução	1
1.1 Visão Geral do Problema e Motivação	4
1.2 Objetivos	6
1.2.1 Objetivo Geral	6
1.2.2 Objetivos Específicos	6
1.3 Organização do Trabalho	7
2 Referencial Teórico	9
2.1 Métodos Ágeis	9
2.1.1 O Manifesto Ágil	11
2.1.2 <i>eXtreme Programming</i> (XP)	12
2.2 Processos de Usabilidade	19
2.2.1 Praxis-u	21
2.3 Trabalhos Relacionados	25
2.3.1 Considerações	33
3 Metodologia de Pesquisa	35
3.1 Tipo de Pesquisa	35
3.2 Procedimentos Metodológicos	37

4	O método XPu	41
4.1	Visão de Projeto	42
4.1.1	Atividades Técnicas	42
4.1.2	Atividades Gerenciais	51
4.2	Visão de Iteração	53
4.2.1	Atividades Técnicas	53
4.3	Visão de Desenvolvimento	56
4.3.1	Atividades Técnicas	56
5	Avaliação	63
5.1	Caracterização de Agilidade	63
5.2	Estudo de Caso	67
5.2.1	Definição dos Objetivos	68
5.2.2	Planejamento	69
5.2.3	Execução	71
5.2.4	Análise e Interpretação dos Resultados	81
6	Conclusão	87
6.1	Considerações Finais	87
6.2	Trabalhos Futuros	90
6.3	Publicações Resultantes	90
	Referências Bibliográficas	91
	Apêndice A Descrição de Heurísticas	101
	Apêndice B Legenda para o XP/XPu	103
	Apêndice C Glossário	105
	Anexo A Relatório de Avaliação Heurística	107

Capítulo 1

Introdução

O desenvolvimento de software vem passando por um processo constante de evolução desde a chamada “crise do software” nos anos 70. O rápido crescimento pela demanda de software, a complexidade dos problemas a serem resolvidos e a inexistência de técnicas estabelecidas para o desenvolvimento de sistemas que funcionassem adequadamente ou que pudessem ser validadas foram os principais fatores que determinaram a crise descrita por Sommerville [2003].

Desde o final da década de 60, no entanto, o termo “Engenharia de Software” passou a ser utilizado para designar um conjunto de técnicas, práticas e paradigmas que a indústria incorporou rapidamente na tentativa de melhorar os seus processos de produção de software. Com isso, processos e técnicas experimentadas pela engenharia tradicional foram adaptadas ao contexto de desenvolvimento de software com o objetivo de torná-lo mais próximo do sistema de produção de outras engenharias já consolidadas.

Modelos de desenvolvimento surgiram para criar uma disciplina que pudesse ser seguida em diferentes projetos. O modelo em cascata [Royce, 1970] foi o primeiro a propor o desenvolvimento dividido em fases sequenciais, iniciando por atividades de levantamento e mapeamento de requisitos e terminando com atividades de teste. Em seguida, vieram modelos como o cascata com realimentação, a espiral de Boehm [Boehm, 1986] e os modelos baseados em protipagem e entrega evolutiva [Kotonya & Sommerville, 1998].

Assim como na indústria tradicional, esses modelos foram construídos segundo o pressuposto de que era possível sistematizar o processo de desenvolvimento de software através de um ciclo de vida que poderia ser seguido em diferentes projetos, de características e níveis de complexidade distintos.

No entanto, alguns autores mostraram que o processo de desenvolvimento de software é muito mais complexo do que o desenvolvimento de produtos em outras

indústrias. Cockburn [2006] sugere que o desenvolvimento de software depende muito mais das pessoas e da comunicação entre elas do que de fatores técnicos e operacionais. Outros autores como Larman [2003] e Poppendieck & Poppendieck [2003] sugerem outras importantes diferenças entre o desenvolvimento de software e os processos industriais utilizados por outras engenharias.

Ao longo da década de 90, foram surgindo os métodos ágeis de desenvolvimento de software como uma alternativa ao desenvolvimento de software tradicional, até então referência no mercado. Em 2001, um grupo de importantes desenvolvedores de software se reuniu em prol de um Manifesto Ágil [Beck et al., 2001], formalizando a filosofia de desenvolvimento até então utilizada por eles em seus projetos, e que serviu de referência para a criação e o aprimoramento dos métodos que já eram utilizados e que viriam posteriormente.

A filosofia de desenvolvimento ágil propõe um conjunto de valores com foco no cliente e nas pessoas envolvidas, na entrega constante de software funcional, no desenvolvimento iterativo baseado em ciclos curtos de entrega e na aceitação da constante mudança dos requisitos ao longo do desenvolvimento; boa parte das características citadas como principais diferenças entre a indústria de software e a indústria tradicional.

Beck [1999] propôs um dos mais importantes conjuntos de práticas e valores relacionados às idéias do movimento de desenvolvimento ágil denominado XP (*eXtreme Programming* ou Programação eXtrema). Atualmente, o XP está na sua segunda edição [Beck & Andres, 2004] e aprimorou, além de valores e práticas, um conjunto de princípios ágeis. Esses valores, princípios e práticas estão relacionados à forma como o método percebe o desenvolvimento de software, sempre alinhados à filosofia defendida pelo Manifesto Ágil.

Diversos trabalhos na literatura atual descrevem importantes resultados relacionados à efetividade de adoção desses métodos em diferentes contextos [Bernardo & Kon, 2007; da Silva et al., 2005; Lui & Chan, 2004; Mann & Maurer, 2005; Muller, 2004; Sato et al., 2006].

Um segundo aspecto importante a observar no desenvolvimento de software é o tratamento da usabilidade do produto final. Segundo a norma ISO/IEC-9241 [1997], a usabilidade é definida como a capacidade de um produto ser usado por usuários específicos para atingir determinados objetivos com eficácia, eficiência e satisfação, em um contexto específico de uso.

Além disso, a norma ISO/IEC-9126 [2001] define a usabilidade como um dos atributos da qualidade de software. Logo, tratar da usabilidade ao longo dos processos de desenvolvimento de software é uma importante atividade que deve ser desempenhada

durante toda a execução do processo, visando à qualidade do produto final.

A usabilidade, portanto, é uma característica que deve ser cuidadosamente observada na construção de um produto de software. As atividades relacionadas à usabilidade podem ser tratadas como um processo (ou subprocesso) ao longo de um ciclo de vida de desenvolvimento. Alguns dos processos de desenvolvimento de software como o RUP [RUP, 2000] e o Praxis [Paula-Filho, 2003] tratam da usabilidade de seus produtos somente através de um fluxo, uma disciplina ou um subprocesso integrado e extensível ao processo padrão. Outros autores como Mayhew [1999] e Hix & Hartson [1993] também apresentam processos de usabilidade, porém independentes de um processo de desenvolvimento de software específico. Naturalmente, esses processos podem ser adaptados e integrados ao ciclo de vida de desenvolvimento através de suas personalizações.

As práticas de usabilidade presentes nesses processos retiram o foco exclusivamente das questões técnicas internas ligadas ao produto e colocam o foco nas necessidades e características dos usuários finais relacionadas à utilização do sistema que será desenvolvido. Informações importantes como o ambiente de realização das tarefas, o perfil técnico, social e cultural dos usuários da aplicação e a expectativa em relação ao produto são alguns dos fatores levados em conta e cuidadosamente observados durante as atividades de usabilidade.

O foco nos usuários finais durante o projeto do sistema é descrito na literatura como UCD (*User-Centered Design*) [Constantine & Lockwood, 1999, 2002]. Essa abordagem específica que as atividades desenvolvidas devem ser direcionadas de forma a prover qualidade de uso ao usuário final da aplicação. Esse objetivo é compartilhado pelos processos de usabilidade.

Já os métodos ágeis, apesar da sua crescente aceitação na indústria atual de desenvolvimento de software, não tratam das práticas relacionadas à usabilidade [Constantine, 2001]. Em relação ao XP e aos demais métodos, ainda não está claro como incorporar as práticas de um processo de usabilidade sem sacrificar os benefícios providos pelo desenvolvimento ágil de software [Lee, 2006].

Diante desse cenário, torna-se necessário especificar um conjunto de práticas de usabilidade que possam se integrar aos métodos ágeis, sem prejuízo das características que permitem classificá-los como tal.

Para tanto, este trabalho propõe um método de usabilidade denominado XPu. Esse método foi definido através de uma adaptação do processo de usabilidade denominado Praxis-u, proposto por Pádua et al. [2006]. O XPu é resultado de uma pesquisa direcionada a encontrar um método capaz de incorporar práticas de usabilidade ao XP, sem que para isso seja necessário descaracterizar sua agilidade.

É fundamental que o método ora proposto também possa ser adaptado sem grandes dificuldades por equipes que já utilizem o XP em projetos de software.

Diversos trabalhos na literatura atual têm se dedicado a tentativas nesse sentido, porém quase sempre sem uma análise do impacto das atividades de usabilidade na “agilidade” do processo padrão. Essa análise é fundamental, visto que o grande desafio encontrado na adoção dessas práticas é exatamente encontrar o limite entre “agilidade” e burocracia.

É importante observar, por fim, que em termos técnicos, este trabalho assume o entendimento de que os métodos ágeis descrevem um conjunto de atividades, práticas, valores e princípios que devem ser incorporados a um processo de desenvolvimento de software. Por essa razão, o XPu foi classificado como método, já que descreve um conjunto de atividades de usabilidade integradas ao método XP e que devem ser associadas a um processo que possa ser instanciado e executado.

1.1 Visão Geral do Problema e Motivação

A integração entre processos de usabilidade e métodos ágeis é esperada e possível, visto que tanto os métodos ágeis como os processos de usabilidade têm em comum características que colocam o foco do desenvolvimento nas necessidades e anseios dos usuários finais, na interação entre os *stakeholders* envolvidos e na qualidade final do produto a ser desenvolvido.

As atividades de UCD [Constantine & Lockwood, 1999, 2002], que são compartilhadas pelo processo Praxis-u, implicam em uma abordagem multidisciplinar ao *design*; baseada no envolvimento ativo dos usuários para um entendimento claro do papel destes, dos requisitos das tarefas e das iterações de *design* (como projeto e processo), além das avaliações. São consideradas como elemento-chave para utilidade e usabilidade.

A abordagem de UCD vista como processo leva em consideração o usuário, o contexto de uso e o ambiente no qual o produto será utilizado. UCD requer um maior investimento nos estágios iniciais do desenvolvimento, mas tem sido eficiente na redução de custos de desenvolvimento e também no custo final de produção. Além disso, a abordagem de UCD ajuda a reduzir o risco de modificações de requisitos que inevitavelmente levariam ao retrabalho.

De acordo com a norma ISO/IEC-13407 [1999], os princípios de UCD são:

- envolvimento ativo dos usuários;
- alocação apropriada de funções para o sistema e para o usuário;

- iteração de soluções de *design*;
- multidisciplinariedade.

Já as atividades de UCD envolvem:

- entender e especificar o contexto de uso;
- especificar as necessidades dos usuários e da organização;
- produzir mais do que uma possível solução de *design*;
- avaliar as propostas de acordo com as necessidades identificadas.

É possível perceber que os objetivos, princípios e atividades da abordagem de UCD são bastante similares aos da abordagem utilizada pelos métodos ágeis. Ao colocar as necessidades do usuário como foco central do processo de desenvolvimento, as duas abordagens convergem para objetivos comuns e esta é uma das principais motivações deste trabalho. O ciclo de vida iterativo e incremental também é percebido nas duas abordagens, com destaque para o Design Iterativo em UCD e dos métodos ágeis baseados no ciclo de vida em espiral como o XP. O método XPu aqui proposto apresenta uma alternativa factível para a integração das duas abordagens, que compartilham, além de seus objetivos, uma série de princípios e práticas.

Para tanto, nossa hipótese fundamental de pesquisa é a de que a integração proposta pelo XPu acrescente qualidade à interface final do produto, sem que para isso seja necessária a burocratização do processo, isto é, sem que o método XP seja descaracterizado em sua “agilidade”. Os métodos ágeis como o XP propõe uma alternativa ao desenvolvimento de software centrado em documentação e pouca iteração com o cliente em prol de benefícios como entrega constante e *feedback* contínuo e adequado do cliente. Por compartilharem benefícios em comum, nossa hipótese secundária é a de que métodos ágeis e abordagens de UCD possam ser utilizadas em harmonia dentro de um mesmo processo de desenvolvimento de software.

O desafio deste trabalho, portanto, é avaliar o método proposto de maneira que seja possível caracterizar o método XPu em sua agilidade, ou seja, demonstrar que mesmo após a integração com práticas de UCD, a filosofia e as atividades do método resultante continuam caracterizando o processo como ágil. Como já foi ressaltado anteriormente, essa validação tem sido negligenciada por grande parte dos trabalhos que compõem o estado da arte atual.

Um segundo desafio de pesquisa é demonstrar que a utilização do método XPu instanciado em um processo de desenvolvimento de software é factível e não representa

expressiva sobrecarga de trabalho a uma equipe ágil. Para isso, é necessário sua utilização em um estudo de caso representativo do ambiente real de desenvolvimento ágil, utilizando uma equipe com as características originais de desenvolvimento propostas pelo método XP.

1.2 Objetivos

Os objetivos deste trabalho são apresentados abaixo em uma visão do objetivo geral e um conjunto de objetivos específicos.

1.2.1 Objetivo Geral

O objetivo geral do trabalho aqui apresentado é propor uma solução ao cenário de integração entre métodos ágeis e práticas de usabilidade. Para uma solução ideal e factível a esse cenário, é necessário também demonstrar que a integração proposta não descaracteriza a agilidade do processo e que este permanece possível de ser instanciado em uma equipe ágil que atua em um ambiente de desenvolvimento XP.

1.2.2 Objetivos Específicos

- Definir como as práticas do processo de usabilidade Praxis-u podem se adaptar ao contexto ágil;
- Apresentar um método integrando o processo Praxis-u ao método XP;
- Discutir pontos de contenção e limites de adaptação do Praxis-u ao XP;
- Verificar até que ponto o método proposto não descaracteriza a agilidade do método XP;
- Avaliar o processo proposto em um estudo de caso com profissionais de desenvolvimento de software;
- Verificar as dificuldades encontradas na instanciação e execução de um “processo ágil” a partir do método proposto.

1.3 Organização do Trabalho

Os capítulos seguintes dessa dissertação estão organizados de modo a prover uma visão teórica dos processos, métodos e técnicas aqui utilizados e uma visão da aplicação prática resultante da validação do método XPu.

O **capítulo 2** apresenta o referencial teórico do trabalho e descreve os conceitos envolvidos com os métodos ágeis, com o Manifesto Ágil e com o método XP, em particular. Além disso, o capítulo discute o conceito de processos de usabilidade e apresenta o Praxis-u em detalhes. Em seguida, são discutidos os principais e mais importantes trabalhos relacionados, apontando suas principais contribuições e suas limitações em relação ao método XPu.

O **capítulo 3** apresenta a metodologia de pesquisa utilizada por este trabalho. Deve ser possível com esse capítulo que outros pesquisadores da área possam identificar de maneira clara qual a classificação da pesquisa utilizada e quais os procedimentos metodológicos adotados por ela.

O **capítulo 4** apresenta o método XPu em suas três visões, discutindo em detalhes cada uma das técnicas utilizadas e suas sugestões de uso e incorporação em um processo instanciado baseado no XP.

O **capítulo 5** apresenta as avaliações realizadas com o método XPu. No início do capítulo, é apresentada a discussão sobre a caracterização de agilidade do método e em seguida, é apresentado o estudo de caso realizado para avaliar o método proposto. O principal objetivo do estudo foi verificar a execução das atividades do método na prática e o impacto dessas no esforço de desenvolvimento de uma equipe ágil.

Finalmente, o **capítulo 6** apresenta as conclusões e considerações finais relativas aos resultados obtidos pelo XPu e aponta os trabalhos futuros que podem ser desenvolvidos a partir das contribuições advindas do método.

Capítulo 2

Referencial Teórico

Este capítulo descreve de forma sucinta as principais bases teóricas que permeiam o desenvolvimento deste trabalho, bem como suas relações com o método de usabilidade que nos propusemos a definir.

A seção 2.1 apresenta o referencial teórico sobre métodos ágeis. São discutidos em detalhes os conceitos relacionados ao Manifesto Ágil e ao método XP. Em relação ao método XP são apresentados suas práticas, valores e princípios, segundo Beck [1999] e Beck & Andres [2004]. A seção 2.2 apresenta o referencial sobre processos de usabilidade com foco no Praxis-u, o processo que foi adaptado para a definição do XPu. Por fim, a seção 2.3 apresenta os trabalhos relacionados ao XPu e publicados na literatura técnica. O conjunto desses trabalhos descreve o estado da arte da área em questão.

2.1 Métodos Ágeis

Os métodos ágeis surgiram a partir da década de 90 e ganharam bastante atenção da academia e da indústria de software nos últimos anos. Esses métodos propõem uma alternativa ao desenvolvimento de software tradicional, reunindo práticas já consagradas mas que até então não haviam sido utilizadas em conjunto com o objetivo de entregar software com qualidade a curto prazo.

Autores como Beck [1999], Schwaber & Beedle [2001] e Cockburn [2004] são alguns dos principais responsáveis pelos métodos ágeis de maior sucesso na indústria de software atual. O objetivo em comum a esses métodos é focar o desenvolvimento no cliente, valorizar e respeitar as pessoas envolvidas, desenvolver em iterações e ciclos curtos e entregar software funcional constantemente e com qualidade.

Atualmente, os principais métodos ágeis são:

eXtreme Programming (XP): proposto por Beck [1999] e aperfeiçoado por Beck & Andres [2004], propõe um conjunto de valores, princípios e práticas para o desenvolvimento ágil. O XP enfatiza principalmente a comunicação, a coragem para se adaptar a mudanças e o respeito e confiança mútua entre clientes e desenvolvedores.

Scrum: proposto por Schwaber & Beedle [2001], apresenta um método mais centrado nos aspectos gerenciais do desenvolvimento de software. Por essa razão, muitas vezes é utilizado em conjunto com outro método como o XP, que aborda questões mais técnicas e operacionais.

Lean Software Development: proposto por Poppendieck & Poppendieck [2003], foi desenvolvido com base no sistema de produção da Toyota. O método é centrado em “eliminar desperdícios” e teve bastante respaldo na indústria de produtos e gerenciamento da cadeia de suprimentos.

Família *Crystal*: proposto por Cockburn [2004], propõe uma família de métodos por acreditar que diferentes abordagens são necessárias para diferentes equipes e níveis de complexidade no desenvolvimento. Apesar de apresentar uma família de métodos, todos eles compartilham dos valores definidos para os métodos ágeis.

Feature Driven Development (FDD): publicado por Palmer & Felsing [2002], define duas fases compostas por cinco processos bem definidos e integrados: a fase de concepção e planejamento e a fase iterativa de construção. O FDD utiliza a linguagem UML estereotipada para gerar seus modelos.

Adaptive Software Development: proposto por Highsmith [1997], esse método explora a natureza adaptativa e a incerteza no desenvolvimento de software. É dividido em três fases distintas: exploração, colaboração e aprendizado. Possui suas idéias fundamentadas nos Sistemas Adaptativos Complexos, fruto da Teoria do Caos.

Dynamic System Development Method (DSDM): proposto por Stapleton [1997] através de um consórcio de empresas britânicas. O método prescreve duas fases iniciais: um estudo de viabilidade e um estudo de negócio. Após essa prospecção inicial, o método prossegue de maneira iterativa e incremental em três etapas: uma para o modelo funcional, uma para o *design* e construção do sistema e uma última de implementação para entrega e implantação do produto.

O método XP, foco de análise deste trabalho, será abordado em detalhes na seção 2.1.2.

2.1.1 O Manifesto Ágil

O manifesto para o desenvolvimento ágil de software (*Agile Manifesto* ou Manifesto Ágil) [Beck et al., 2001] foi publicado inicialmente em fevereiro de 2001 por um grupo de 17 importantes desenvolvedores de software em Utah. Os resultados do manifesto representavam a visão desse grupo sobre o que deveria ser valorizado e quais os direcionamentos que a indústria de software deveria adotar para entregar software de qualidade de maneira mais eficiente.

Baseado nessa filosofia, o manifesto propõe os seguintes valores:

- **Indivíduos e interações** são mais importantes que processos e ferramentas;
- **Software funcionando** é mais importante que documentação completa e detalhada;
- **Colaboração do cliente** é mais importante que negociação de contratos e
- **Adaptação às mudanças** é mais importante que seguir um plano.

Apesar do manifesto reconhecer a importância dos itens à direita, são mais valorizados os itens em destaque à esquerda. Além desses quatro valores, o manifesto ainda define 12 princípios considerados fundamentais para o desenvolvimento ágil de software. São eles:

- A maior prioridade é a satisfação do cliente por meio de entregas rápidas e contínuas de software que agrega valor;
- Mudanças nos requisitos são aceitas, mesmo em estágios mais avançados no desenvolvimento. Métodos ágeis aceitam mudanças que trarão vantagem competitiva para o cliente;
- Software que funciona é entregue frequentemente, em períodos que variam de algumas poucas semanas a poucos meses, com preferência para intervalos mais curtos;
- Pessoas de negócio e desenvolvedores devem trabalhar juntos diariamente no projeto;
- Construção de projetos através de indivíduos motivados, fornecendo o ambiente e o suporte necessário, confiando que o trabalho será realizado;
- A maneira mais eficiente e eficaz de transmitir informações dentro e fora da equipe de desenvolvimento é a comunicação face a face;

- Software funcionando é a medida principal de progresso;
- Métodos ágeis promovem o desenvolvimento em um ritmo sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante;
- Atenção contínua à excelência técnica e ao bom *design* ajuda a melhorar a agilidade;
- Simplicidade - a arte de minimizar a quantidade de trabalho necessário - é essencial;
- Os melhores requisitos, arquiteturas e *design* surgem de equipes auto-gerenciadas e
- Em intervalos regulares, a equipe deve refletir sobre como se tornar mais eficiente, refinando e ajustando seu comportamento apropriadamente.

Baseado nesse manifesto, os métodos ágeis compartilham e implementam valores e princípios de maneira a tornar o processo de desenvolvimento de software mais ágil, eficiente e agradável.

2.1.2 *eXtreme Programming* (XP)

O método XP (*eXtreme Programming* ou Programação eXtrema) proposto por Kent Beck em 1999 [Beck, 1999] tem como objetivo a excelência no desenvolvimento de software, visando baixo custo, poucos defeitos, alta produtividade e alto retorno de investimento.

As idéias de XP começaram a ser definidas a partir das experiências do autor no desenvolvimento de software em Smaltalk. Em 1996, o conjunto inicial das práticas de XP foi implementado no projeto C3 (*Chrysler Comprehensive Compensation System*) do qual Kent Beck era o principal responsável. Boa parte dessas práticas já era conhecida pela indústria de desenvolvimento de software. A idéia de Kent Beck era reuni-las e utilizá-las ao eXtremo, já que eram comprovadamente eficientes.

Na segunda edição do XP, Beck & Andres [2004] aprimoram a definição apresentada por Beck na primeira edição, enumerando suas principais características:

- **XP é um método leve:** o time só deve fazer o necessário para trazer valor para o cliente;

- **XP é um método que enfatiza o desenvolvimento de software:** apesar de ter implicações em áreas como marketing, vendas ou operações, o XP não tenta resolver os problemas diretamente ligados a elas;
- **XP funciona para times de qualquer tamanho:** apesar das práticas de XP funcionarem melhor em times pequenos, seus valores e princípios podem ser aplicados em qualquer escala e;
- **XP se adapta a requisitos vagos ou em constante mudança.**

Ainda segundo os autores, a Programação eXtrema inclui:

- uma filosofia para o desenvolvimento de software baseada nos valores de **Comunicação, Feedback, Simplicidade, Coragem e Respeito;**
- um conjunto de práticas comprovadamente úteis para melhorar o desenvolvimento de software. As práticas expressam os valores de XP;
- um conjunto complementar de princípios, técnicas intelectuais que auxiliam a tradução dos valores em práticas, úteis quando as práticas existentes não resolvem seu problema particular e;
- uma comunidade que compartilha os mesmos valores e muitas das mesmas práticas.

Conforme apresentado, o XP inclui um conjunto de valores, práticas e princípios que foram enfatizados na segunda edição e é útil, pois possibilita ter uma visão mais ampla e integrada do método.

2.1.2.1 Valores

Valores são critérios amplos utilizados para julgar determinada situação. Beck & Andres [2004] estabelecem cinco valores defendidos pelo método XP. São eles:

- **Comunicação:** o XP pressupõe que a maioria dos problemas no desenvolvimento de software ocorre por dificuldades de comunicação. Essa dificuldade geralmente ocorre na comunicação entre todos os *stakeholders* envolvidos e acarreta sérios problemas no entendimento dos requisitos reais da aplicação.

- **Feedback:** é fundamental para uma equipe XP obter respostas rápidas do cliente para se adaptar às mudanças. O *feedback* deve estar presente diariamente na interação entre cliente e desenvolvedores e na aprovação por parte do cliente de cada uma das iterações.
- **Simplicidade:** os membros de uma equipe XP estão frequentemente buscando a solução mais simples possível para resolver seus problemas atuais. Como as equipes trabalham com requisitos que mudam frequentemente, é importante obter sempre o *design* mais simples possível para suportar futuras modificações.
- **Coragem:** coragem para aceitar as mudanças, experimentar novas tecnologias, prosseguir com o projeto sem um escopo bem definido e dispensar a documentação. A equipe XP convive todo o tempo com essas variáveis e o cliente concorda em trabalhar dessa maneira.
- **Respeito:** se os membros da equipe não se importam uns com os outros e com o resultado de suas ações sobre o sistema, então XP não vai funcionar. É importante reconhecer que a excelência no desenvolvimento de software depende das pessoas, e elas devem se respeitar para conseguir extrair o máximo do seu potencial.

Os valores do XP trabalham de maneira integrada e não podem ser vistos separadamente. Esses valores formam a base para a execução das práticas.

2.1.2.2 Princípios

Os princípios são definidos como mecanismos, do ponto de vista filosófico, que possibilitam a implementação das práticas e dos valores. Os princípios de XP são:

- **Humanidade:** as pessoas têm papel fundamental no desenvolvimento de sistemas. Suas necessidades básicas e anseios devem ser respeitados e levados em conta ao longo de todo o desenvolvimento.
- **Economia:** é importante que toda a equipe XP esteja sempre preocupada em agregar valor ao negócio associado ao sistema que estão desenvolvendo.
- **Benefício Mútuo:** os benefícios advindos do desenvolvido do sistema devem se estender a todos os envolvidos.
- **Auto-Semelhança:** permite que uma solução seja aplicada a diferentes contextos e até mesmo em projetos de diferentes escalas.

- **Melhoria:** o time está sempre em busca de maneiras de melhorar suas atividades diárias relativas ao software produzido e ao próprio processo.
- **Diversidade:** as várias habilidades em uma equipe são valorizadas. O time deve ser suficientemente heterogêneo para suprir todas as necessidades do desenvolvimento.
- **Reflexão:** é necessário que a equipe reflita constantemente acerca do trabalho realizado, buscando atingir a melhoria contínua.
- **Fluxo:** software funcionando deve ser entregue em fluxo contínuo. Os *releases* e iterações devem ser suficientemente pequenos para permitir entregas constantes.
- **Oportunidade:** é importante observar e aproveitar todas as oportunidades de boas mudanças ao longo do desenvolvimento.
- **Redundância:** sugere que para os problemas críticos e de difícil compreensão devem ser testadas várias soluções diferentes, buscando a solução ideal.
- **Falha:** indica que toda falha é uma oportunidade de aprendizado. Caso haja mais de uma solução possível para o problema, implemente-as de modo que as falhas sejam evitadas.
- **Qualidade:** característica essencial do desenvolvimento XP. A qualidade não é uma variável de controle e é inaceitável para o time sacrificá-la em prol de interesses diversos.
- **Passos Pequenos:** fazer o mínimo possível para que aquela porção de software funcione e melhorar o artefato produzido em sucessivos refinamentos.
- **Aceitação de Responsabilidades:** a responsabilidade não deve ser imposta e sim aceita pelo time.

2.1.2.3 Práticas

As práticas são atividades experimentadas e que visam melhorar - do ponto de vista ágil - o trabalho do desenvolvimento de software. Beck & Andres [2004] definem 24 práticas para o XP, divididas em práticas primárias e práticas corolárias.

Práticas primárias são aquelas que trazem melhorias imediatas para o time e podem ser aplicadas separadamente, apesar de haver uma forte sinergia entre elas. As práticas primárias definidas pelos autores são:

- **Sentar Junto:** o ambiente de desenvolvimento deve ser integrado com todo o time em um ambiente aberto e informativo, contribuindo para a colaboração.
- **Time Completo:** as equipes devem ser multidisciplinares e possuir todas as habilidades necessárias para a execução do projeto.
- **Área de Trabalho Informativa:** deve ser possível acompanhar o progresso e o andamento do projeto a partir de qualquer ponto do ambiente de desenvolvimento.
- **Trabalho Energizado:** a equipe deve se manter motivada e trabalhar com prazos factíveis para execução de suas atividades diárias.
- **Programação Pareada:** todo o código desenvolvido pela equipe deve ser escrito em pares que se revezam constantemente.
- **Histórias:** o levantamento de requisitos e o planejamento em XP são feitos através de histórias escritas pelo cliente.
- **Ciclo Semanal:** o planejamento de uma iteração em XP deve ser feito para uma semana.
- **Ciclo Trimestral:** os *releases* - formados por um conjunto de iterações - são planejados para um trimestre.
- **Folga:** o planejamento deve ser feito com contingência e as estimativas não podem ser vistas como compromissos.
- **Build em 10 minutos:** toda a bateria de testes do sistema e o *build* automático devem rodar em, no máximo, 10 minutos.
- **Integração Contínua:** o código pertence a um repositório único e deve estar sempre integrado ao que já foi produzido e testado pelos outros pares do time.
- **Desenvolvimento Dirigido por Testes:** um dos principais focos do XP. Os testes são sempre automatizados e escritos antes do código fonte.
- **Design Incremental:** desenvolver apenas o que foi especificado para aquela iteração. Não prever funcionalidades futuras.

Práticas corolárias são aquelas mais difíceis de serem implementadas por times que estão começando a utilizar XP, pois demandam maior experiência da equipe com o método. As práticas corolárias definidas pelos autores são:

- **Envolvimento Real com o Cliente:** é fundamental para a equipe XP que o pessoal de negócios faça parte do time. O cliente precisa participar ativamente como membro da equipe.
- **Implantação Incremental:** substituir sistemas existentes gradualmente, de maneira que a implantação gere pouco ou nenhum impacto para o cliente e os usuários.
- **Continuidade da Equipe:** manter equipes eficientes trabalhando juntas é um benefício que aumenta a produtividade do time.
- **Diminuição da Equipe:** reduzir a carga de trabalho de membros específicos da equipe para que eles possam formar novas equipes.
- **Análise de Causa Inicial:** investigar a origem do problema e das falhas introduzidas no sistema, com o objetivo de evitar que os problemas ocorram novamente.
- **Código Compartilhado:** o código-fonte produzido pertence a todo o time e qualquer membro da equipe pode alterá-lo, desde que execute os testes.
- **Código e Testes:** são os únicos artefatos a serem mantidos por uma equipe XP. Se necessário, documentações podem ser geradas a partir dos códigos e testes.
- **Repositório de Código Unificado:** todo código desenvolvido alimenta um mesmo repositório único para todo o time. Somente códigos que passaram pela execução de todos os testes podem alimentar o repositório.
- **Implantação Diária:** novas versões do sistema devem ser colocadas em produção diariamente.
- **Contrato de Escopo Negociável:** o cliente deve fixar prazo e custo, enquanto que o escopo é negociável e acordado com o time.
- **Pague-Pelo-Uso:** determina que o cliente pode optar por pagar pelo sistema à medida que o utiliza. Nesse caso, o cliente só paga pelo *release* que foi entregue e está em funcionamento.

2.1.2.4 Papéis

Para a execução do método, o XP também estabelece um conjunto de papéis para as pessoas envolvidas com o time. São eles:

- **Programadores:** são os responsáveis pelo desenvolvimento dos códigos-fonte da aplicação, dos testes unitários automatizados do sistema e por estimar histórias e tarefas. O *Coach* e o *Tracker* são tipos especiais de programadores. O primeiro é geralmente o membro mais experiente em XP na equipe e contribui para a disseminação de conhecimento no time. Já o segundo é responsável por coletar constantemente métricas relacionadas ao desenvolvimento com o objetivo de monitorar a execução do projeto como planejado.
- **Arquitetos:** são os responsáveis por refatorar todo o sistema em larga escala, melhorando a arquitetura do software e sugerindo alternativas de projeto.
- **Analistas de Teste:** trabalham diretamente com o cliente e com os Analistas de Negócio auxiliando na escrita dos testes de aceitação do sistema.
- **Analistas de Negócio:** trabalham diretamente com o cliente para definir as histórias do sistema.
- **Projetista de Interação:** avalia o modo como o sistema está sendo utilizado pelos usuários finais, sugerindo, quando necessário, melhorias na interface gráfica.
- **Gerentes de Projeto:** facilitam a comunicação dentro do time, servindo de referência para o relacionamento com pessoas externas à equipe.
- **Gerentes de Produto:** escrevem e priorizam histórias para o Ciclo Semanal e definem os temas para o Ciclo Trimestral. Trabalham juntos com os Usuários nessas atividades.
- **Executivos:** representam confiança e responsabilidade para o time. Avaliam constantemente se o projeto está alinhado aos objetivos de negócio da organização. Esse papel é representado pelo cliente.
- **Redatores Técnicos:** observam o sistema sob o ponto de vista do usuário final. Contribuem para o *feedback* constante e adequado.
- **Usuários:** ajudam a escrever e priorizar histórias, já que são os responsáveis pela utilização do sistema a ser desenvolvido.
- **Recursos Humanos:** cuidam de problemas burocráticos e alheios ao desenvolvimento.

Os papéis descritos acima correspondem à segunda edição do método XP. Alguns deles foram modificados e outros adicionados em relação aos papéis da primeira edição.

Observa-se que um dos papéis adicionados e de grande interesse deste trabalho foi o papel de Projetista de Interação. A adição desse papel na segunda edição demonstra um princípio de interesse do XP em relação à usabilidade do produto. No entanto, o método não apresenta práticas que contribuam para a execução de atividades de usabilidade associadas a esse papel.

O fato do método indicar o papel de Projetista de Interação, mas não especificar atividades para ele talvez se justifique pelo escopo da abordagem XP. Os autores deixam claro que o objetivo do método é o de especificar atividades relativas apenas ao desenvolvimento do produto de software, ou seja, é possível imaginar nesse caso que os autores não consideram atividades de usabilidade como parte das atividades de desenvolvimento. Esse indício reforça a necessidade de tratar as atividades de usabilidade como atividades extensíveis ao método XP, como propõe este trabalho.

2.2 Processos de Usabilidade

O propósito dos processos de usabilidade, segundo a norma ISO/IEC-12207 [2002], é garantir que sejam considerados os interesses e necessidades dos envolvidos (*stakeholders*), de forma a proporcionar otimização do suporte e do treinamento, aumento da produtividade e da qualidade do trabalho, melhoria das condições para o trabalho humano e redução das chances de rejeição do sistema por parte do usuário.

Os processos de usabilidade, do ponto de vista da Engenharia de Software, são vistos como processos de apoio ao ciclo de vida do software [ISO/IEC-12207, 2002], isto é, fornecem atividades complementares às fases e fluxos de desenvolvimento do produto. Por essa razão, esses processos devem ser definidos e executados por um processo padrão da Engenharia de Software, contribuindo dessa maneira para o sucesso e a qualidade do produto de software a ser desenvolvido do ponto de vista da usabilidade.

Assim como os demais processos, os processos de usabilidade definem atividades ordenadas em um fluxo de trabalho que devem ser seguidas de maneira disciplinada para que a execução do processo ocorra de maneira satisfatória, atendendo aos propósitos para os quais eles foram definidos.

Nesse contexto, autores como Mayhew [1999] e Hix & Hartson [1993] propuseram modelos de processo baseados em ciclos de vida bem definidos para as atividades de usabilidade. O ciclo de vida em estrela apresentado na Figura 2.1 e adaptado de [Hix & Hartson, 1993] é atualmente uma importante referência de modelo para a área de Engenharia de Usabilidade.

As atividades do ciclo de vida em estrela não foram definidas de maneira

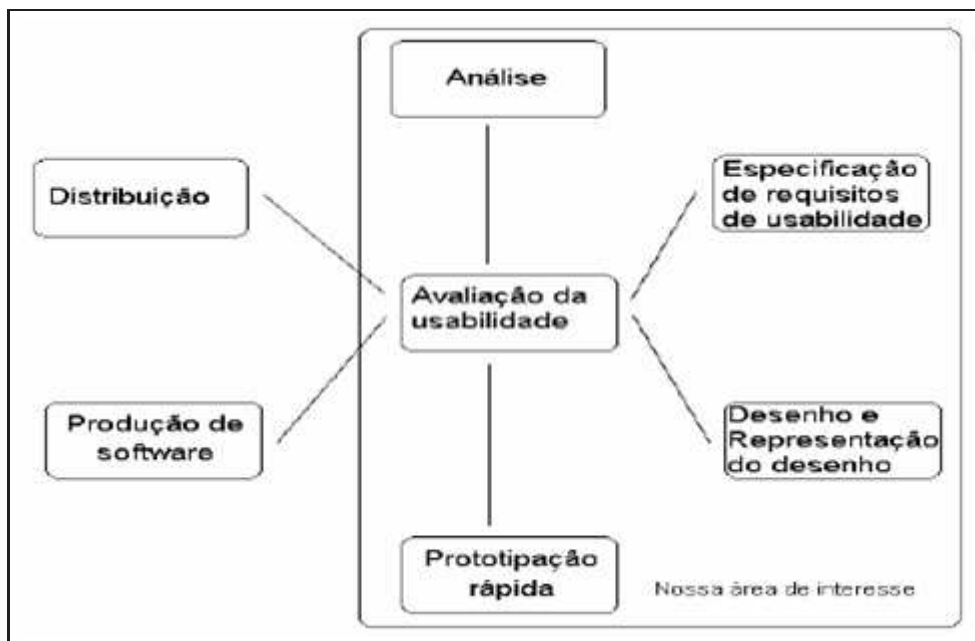


Figura 2.1. Ciclo de vida em estrela (adaptado de [Hix & Hartson, 1993])

sequencial e portanto podem ser realizadas diversas e repetidas vezes ao longo de um mesmo ciclo de vida do produto. O modelo prescreve a Avaliação da Usabilidade como atividade central. À medida que o projeto de interface evolui, são realizadas sucessivas avaliações com o objetivo de identificar problemas de usabilidade o mais cedo possível no ciclo de desenvolvimento. O modelo em estrela também destaca a importância da Prototipação Rápida ao longo do desenvolvimento do produto. Completam o modelo as atividades de Análise, que prevêem abordagens relacionadas à Análise de Usuários, de Tarefas e de Funcionalidades; a Especificação de Requisitos de Usabilidade que determina os parâmetros de usabilidade relacionados às expectativas e necessidades dos usuários finais e, por fim, o Desenho e Representação do Desenho que pode ser gerado a partir das experiências adquiridas e avaliadas através da Prototipação Rápida.

Com exceção das atividades relacionadas à Distribuição e Produção de Software, como já destacado na Figura 2.1, todas as demais atividades propostas pelo ciclo de vida em estrela são de interesse deste trabalho.

Um outro modelo muito utilizado para descrever processos de usabilidade é a norma ISO/IEC-13407 [1999]. Como já citada no capítulo 1, essa norma descreve atividades e princípios em um ciclo de vida centrado na abordagem de UCD. Esse ciclo é apresentado na Figura 2.2.

A estratégia consiste em, a cada ciclo de projeto e testes, identificar e refinar continuamente o conhecimento sobre o contexto de uso do sistema e os requisitos em termos de usabilidade da interface. Nos testes, os representantes dos usuários simulam

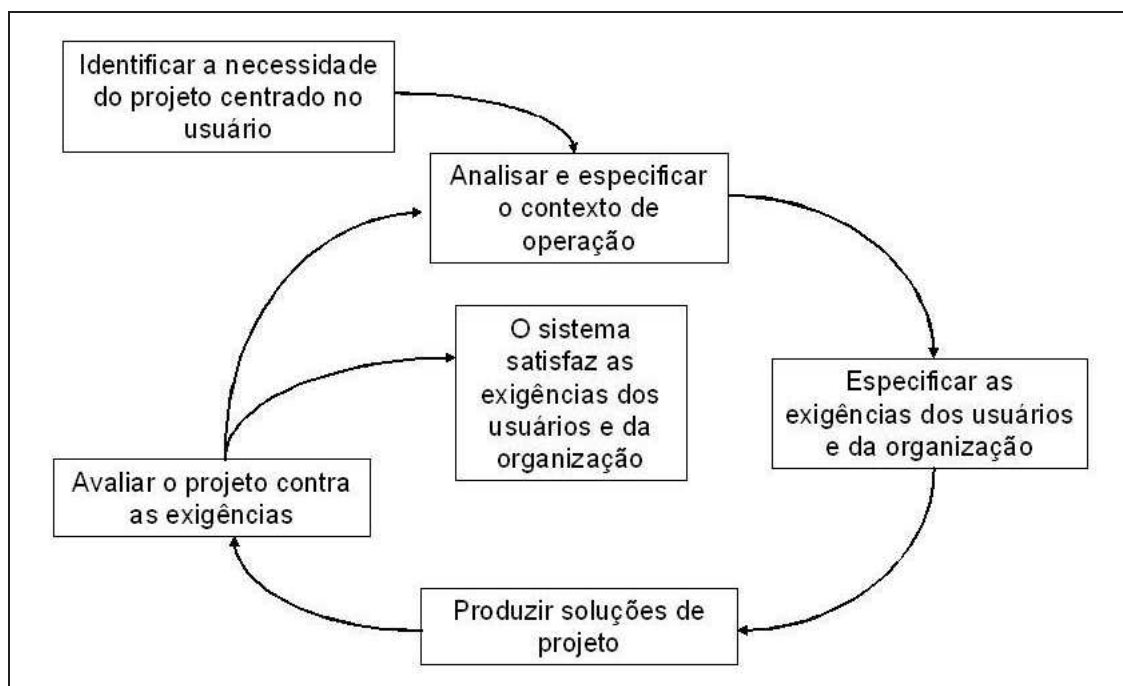


Figura 2.2. Ciclo de vida do modelo ISO/IEC 13407.

a realização de suas tarefas. Inicialmente eles participarão de simulações utilizando protótipos de baixa fidelidade como maquetes ou rascunhos em papel, no entanto com o avanço do desenvolvimento, eles recorrerão a “protótipos funcionais” e versões acabadas do sistema, em simulações cada vez mais fidedignas.

Se implementada desde cedo no desenvolvimento, tal estratégia pode reduzir o risco de falhas conceituais do projeto, garantindo que, a cada ciclo, o sistema responda cada vez melhor às expectativas e necessidades dos usuários em suas tarefas.

O processo de usabilidade Praxis-u, tratado em detalhes na subseção seguinte, utiliza o ciclo de vida proposto pela norma ISO/IEC-13407 [1999] para definir suas atividades.

2.2.1 Praxis-u

O processo de usabilidade Praxis-u, proposto por Pádua et al. [2006], foi desenvolvido para se integrar ao Praxis [Paula-Filho, 2003] como uma disciplina ao longo do ciclo de desenvolvimento de software. Atuando como uma disciplina do Praxis, o Praxis-u propõe a execução de uma série de atividades que possibilitam tratar da usabilidade através das várias fases e fluxos de desenvolvimento. Apesar de integrado ao Praxis, o Praxis-u foi desenhado como um (sub)processo independente, com seus próprios recursos e atividades, podendo ser utilizado associado a outros processos de

desenvolvimento de software.

A escolha do Praxis-u para utilização neste trabalho se deve ao fato de que o processo já se encontra consolidado no ensino e na pesquisa acadêmica e foi definido a partir de trabalhos consagrados como os de Constantine & Lockwood [1999], Hix & Hartson [1993], Hackos & Redish [1998] e Mayhew [1999], além de estar baseado no ciclo de vida proposto pela norma ISO/IEC-13407 [1999].

O Praxis-u apresenta um conjunto de atividade técnicas e gerenciais. A Figura 2.3 apresenta um diagrama de atividades da UML ilustrando a disciplina de usabilidade. As atividades do fluxo de usabilidade estão organizadas em duas raias, correspondentes a essa divisão do conjunto.

As seções a seguir descrevem separadamente o conjunto de atividades técnicas e gerencias descrito para o processo.

2.2.1.1 Atividades Técnicas

As atividades técnicas são de responsabilidade da equipe de desenvolvimento e estão focadas em aspectos que visam à usabilidade. As atividades técnicas do Praxis-u são:

- **Análise de Contexto de Uso:** tem como objetivo principal a caracterização dos usuários, das tarefas que eles realizam, de produtos semelhantes ou concorrentes e do ambiente onde será utilizado o produto em desenvolvimento. Essa análise é composta pela Análise de Usuários, pela Análise de Tarefas, pela Definição de Modelos Mentais utilizados pelas pessoas envolvidas e pela Análise de Concorrência. A Análise de Tarefas trata da modelagem das tarefas realizadas pelos usuários, incluindo a Análise de Necessidades e a Análise de Ambiente de realização das atividades.
- **Definição das Funções do Produto:** essa atividade fica na fronteira das áreas de engenharia de software e usabilidade. As metodologias de engenharia de usabilidade propõem a descrição das tarefas a serem contempladas no produto de uma forma mais ampla (observando as características mapeadas durante a Análise de Usuários e Tarefas) do que normalmente utilizado na área de engenharia de software. Esta atividade visa definir as tarefas ou partes de tarefas que serão automatizadas com o apoio do produto em desenvolvimento.
- **Prototipação de Requisitos de Interface:** o objetivo desta atividade é a criação do Protótipo de Requisitos de Interface (PRI). O PRI é um modelo usado para validação dos requisitos com os usuários. Compreende os aspectos de conteúdo e de navegação da interface, entendidos como requisitos de interação.

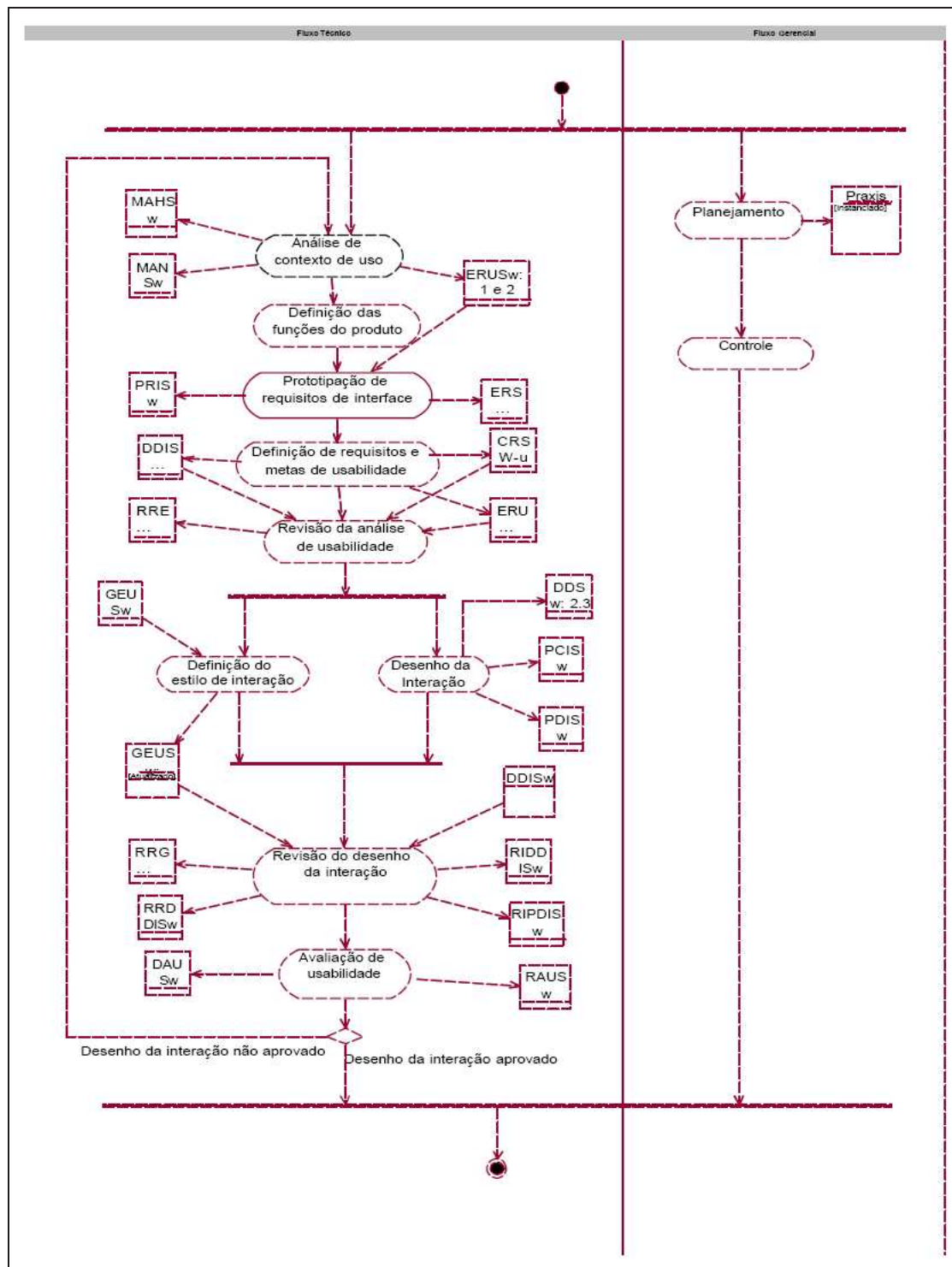


Figura 2.3. Disciplina de Usabilidade do Praxis-u.

- **Definição de Requisitos e Metas de Usabilidade:** essa atividade visa à definição de níveis de qualidade almejados para os atributos de usabilidade considerados importantes para o produto em desenvolvimento. Tarefas de referência (*benchmark*) são utilizadas nas medições envolvendo dados objetivos e questionários podem ser utilizados para a quantificação de dados subjetivos.
- **Revisão da Análise de Usabilidade:** atividade que visa avaliar a qualidade e aperfeiçoar os artefatos produzidos nas atividades de Análise de Contexto de Uso e de Especificação de Requisitos. Como parte do trabalho de revisão, deve ser realizada uma validação do PRI com a participação dos usuários.
- **Definição do Estilo de Interação:** um estilo de interação abrange padrões, diretrizes ou até métodos para o desenvolvimento da interação visando garantir a consistência entre famílias de produtos ou mesmo dentro de um produto. A atividade de Definição de Estilo da Interação visa o desenvolvimento de um estilo de interação ou a atualização e melhoria de estilos criados anteriormente.
- **Desenho da Interação:** a atividade de desenho da interação parte do PRI e dos resultados das atividades de análise com o objetivo de se desenvolver a especificação da interface do usuário pronta para ser desenhada (desenho interno) e implementada em uma plataforma específica. Durante o Desenho da Interação, a prototipagem é um importante recurso para gerar desenhos que atendam às necessidades dos usuários finais.
- **Revisão do Desenho da Interação:** atividade que visa avaliar a qualidade e aperfeiçoar os artefatos produzidos nas atividades de Definição do Estilo de Interação e Desenho da Interação.
- **Avaliação de Usabilidade:** a avaliação de usabilidade visa à avaliação da qualidade da interface como instrumento da interação usuário-computador. Diferentes tipos de avaliação, com objetivos e características próprias podem ser utilizados.

2.2.1.2 Atividades Gerenciais

As atividades gerenciais são de responsabilidade do gerente de projetos e estão focadas no planejamento e no controle da execução das atividades de usabilidade. As atividades gerenciais do Praxis-u são:

- **Planejamento:** o planejamento compreende a personalização do processo de desenvolvimento com relação aos aspectos de usabilidade e uma estimativa de

recursos necessários ao cumprimento do escopo do produto a ser desenvolvido. A estimativa de recursos visa a determinação do esforço (número de horas) e outros recursos necessários ao projeto, no que se relaciona às atividades que visam à usabilidade. A necessidade de recursos deve ser mapeada a marcos que indicam o progresso na evolução do escopo durante a execução do projeto. O mapeamento em marcos de progresso permitirá o acompanhamento (controle) do projeto durante sua execução, possibilitando a confrontação dos cronogramas de metas atingidas de esforço, escopo, prazo e custo.

- **Controle:** o controle compreende o acompanhamento do progresso do projeto, durante sua realização, por meio da confrontação de metas de esforço, escopo, prazo e custo, comparando o previsto no planejamento com o realizado até um determinado momento.

Tipicamente, durante o desenvolvimento do produto de software, podem ser realizados vários ciclos utilizando a disciplina de usabilidade. Personalizações do processo com esse objetivo são possíveis através da atividade gerencial de Planejamento.

2.3 Trabalhos Relacionados

Pesquisas referentes à definição de processos de usabilidade para métodos ágeis são relativamente recentes na comunidade científica. No entanto, o crescente aumento no número de publicações em periódicos e conferências especializadas da área demonstra o claro interesse da comunidade por iniciativas no sentido de entender o processo de integração dos métodos ágeis com as atividades de usabilidade. O tema tem sido pesquisado tanto por profissionais da área de usabilidade, quanto por profissionais da área de processos de software, particularmente, da área de métodos ágeis.

O trabalho de Constantine [2001], publicado em 2001 e reeditado em 2002, foi um dos primeiros a iniciar a discussão sobre o tema. Seu trabalho propõe uma variante simplificada de UCD (*Usage-Centered Design*) que esteja integrada com os métodos “leves”, ditos ágeis. Constantine afirma que XP e os demais métodos ágeis compartilham das mesmas fraquezas apresentadas por processos tradicionais como o Processo Unificado (*Unified Process* - UP), no que se refere à usabilidade e ao desenho de interface de usuário.

A proposta de Constantine é mostrar que UCD está prontamente integrado aos métodos ditos “leves”. Analisando o método XP, percebe-se que as histórias escritas pelos usuários ajudam bastante no processo de modelagem e priorização, visando

o *design* e implementação em sucessivas iterações. Essa característica é bastante reforçada pelos princípios de usabilidade. A existência de cartões escritos com as histórias de usuário facilita bastante o processo de mapeamento de usuários e tarefas em atividades de UCD.

Patton [2002] utiliza o método de *design* de interação baseado no modelo de UCD proposto em Constantine & Lockwood [2003]. O conjunto de recomendações proposto no artigo se baseia em como as práticas ágeis poderiam ser melhoradas utilizando UCD incorporado em pequenos *designs* de interação, de forma que essa atividade seja pensada dia-a-dia durante o desenvolvimento do produto e durante as tomadas de decisão relativas ao seu planejamento. O conjunto de recomendações foi utilizado pelo autor em processos de desenvolvimento iterativos em uma empresa americana. De acordo com Patton [2002], a adoção das práticas ajudou a equipe a entregar software de alta qualidade, enquanto que o sentimento em relação ao resultado do software foi muito mais próximo das expectativas dos usuários finais.

Em 2003, Vasconcelos et al. [2003] descreveram um processo ágil centrado em usabilidade chamado XPU¹. Segundo os autores, o objetivo principal do XPU era prover à equipe de desenvolvimento um modelo para a construção de sistemas de software centrado no usuário que valorizasse a usabilidade como característica fundamental da qualidade. Portanto, o XPU deve guiar o time de desenvolvimento, passo a passo em todo o projeto, segundo técnicas e artefatos simplificados, a fim de se obter não apenas sistemas portáteis, fracamente acoplados ou reutilizáveis, mas, sobretudo, interfaces que reflitam os objetivos, as características e as necessidades do usuário. Ainda segundo os autores, 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. As fases descritas pelo processo estão fundamentadas nos métodos *eXtreme Programming* (XP) e no Método para Concepção de Interfaces (MCI).

Um dos principais pontos questionáveis do trabalho de Vasconcelos et al. [2003] é centrá-lo em documentação através de artefatos descritivos em papel e sustentar um processo de desenvolvimento baseado em fases bem definidas, o que contraria os valores definidos pelo Manifesto Ágil [Beck et al., 2001]. Uma outra limitação importante apresentada por este processo é avaliá-lo apenas utilizando um conjunto de questionários estruturados aplicados a alunos em nível de graduação. Essa abordagem

¹O processo XPU descrito por Vasconcelos et al. [2003] foi descoberto depois que o método apresentado por esta dissertação já havia sido publicado com o nome XPu. Apesar das grafias semelhantes, os dois trabalhos não possuem qualquer relação entre eles.

difícilmente traz resultados relevantes quanto à viabilidade de aplicação prática do processo em um contexto real de desenvolvimento de software, tampouco avalia o impacto na manutenção das características ágeis de um processo de desenvolvimento.

No mesmo ano, Kane [2003] afirmou que algumas técnicas ágeis não resolvem certos problemas de usabilidade, por exemplo, não consideram diferentes necessidades entre usuários novíços e proficientes. O autor afirmou ainda que a engenharia de usabilidade deveria ser de particular interesse da comunidade de desenvolvimento ágil, já que as duas disciplinas compartilham diversas práticas essenciais. O artigo de Kane [2003] discute um conjunto de práticas baseadas no conceito de *Discount Usability Engineering*, apresentado em Nielsen [1993]. Essas práticas incluem o uso de Roteiros, Avaliações Heurísticas, Cartões Ordenados e uma simplificação do método *Think Aloud* [Ericsson & Simon, 1993]. Essas práticas são confrontadas com as práticas ágeis e lacunas entre as duas abordagens são apresentadas e discutidas.

Armitage [2004] apresenta um relatório de experiências pessoais com XP e suas implicações em *design*. O trabalho mensurou o nível de esforço despendido pela equipe de desenvolvimento ao utilizar atividades de *design* ao longo da implementação de seis histórias de usuários. Para isso, o autor utilizou uma abordagem híbrida que demandou um trabalho de *design* em três níveis, realizados em paralelo. O menor e primeiro nível de esforço (obtido nas atividades típicas do XP) suportou iterações com equipes pequenas, provendo desenhos detalhados de componentes para guiar a construção. Um segundo nível de esforço apresentou re-*design* de baixa fidelidade em códigos já implementados, já que era necessário utilizar de re-trabalho em relação às interfaces já desenvolvidas. O objetivo era estimar o esforço com *Refactoring* [Fowler et al., 1999]. A divisão proposta limitou a habilidade dos *designers* para otimizações em apenas uma tentativa, no entanto pelo fato de todo o desenho ter sido construído de forma flexível, não foi difícil estender e aplicar melhorias com o progresso do projeto. Com o posterior detalhamento das atividades trabalhadas nos dois primeiros níveis, o projeto do terceiro nível foi de fácil adaptação para prover uma visão completa do produto. Ao final do trabalho, o autor apresenta um conjunto de diretrizes a serem seguidas pelos *designers* quando trabalharem com projetos em ambientes ágeis.

McInerney & Maurer [2005] apresentaram as dificuldades encontradas em um estudo de caso onde três experientes profissionais de UCD trabalhavam em seus primeiros projetos ágeis. O artigo foi estruturado de forma a apresentar as principais diferenças nas interpretações dadas pelas duas abordagens para atividades importantes do desenvolvimento como análise de usuários, desenho da interface e avaliação de usabilidade.

As conclusões apontam que à primeira vista, os métodos ágeis não são bem

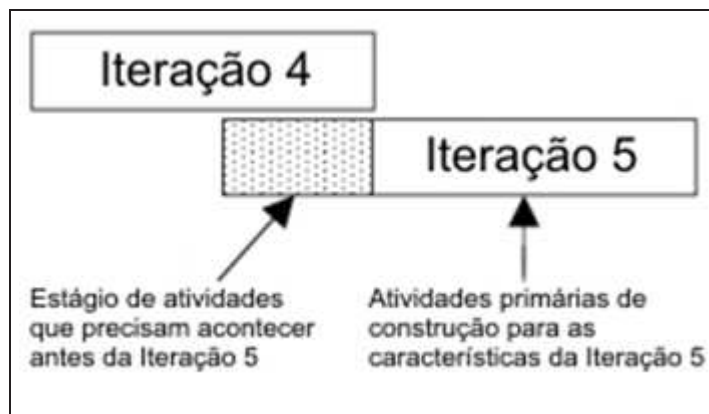


Figura 2.4. Atividades de UED e iterações ágeis.

recebidos pelas equipes de UCD. No entanto, as experiências mostraram que a aderência das duas abordagens é possível e positiva, além de se proporem aos mesmos objetivos. Os autores ressaltam que os métodos ágeis ainda estão em processo de consolidação e que é necessário observar seu comportamento de forma mais detalhada em futuros trabalhos.

Hodgetts [2005] afirma que os métodos ágeis são quase sempre apresentados sob a visão dos programadores, deixando de lado outras importantes disciplinas como a de UED (*User Experience Design*), apresentada pelo artigo. A disciplina envolve práticas como análise de usuário, desenho de interface, desenho visual e testes de usabilidade.

O autor discute suas experiências integrando práticas de UED em iniciativas de processos ágeis em diversas organizações, principalmente utilizando XP e Scrum. São registrados os esforços da equipe com a abordagem centrada em programação dos métodos ágeis, e posteriormente, com os progressos alcançados na integração das melhores práticas de UED com o mundo incremental e colaborativo dos métodos ágeis.

Ao adaptar as práticas de UED às iterações ágeis, Hodgetts observou algumas práticas paralelas entre as atividades de produção de código e as atividades de UED que auxiliaram as equipes a formular abordagens para integrar as atividades de UED com as iterações ágeis. Como é possível observar na Figura 2.4, a integração das atividades se iniciam ao final da iteração 4, antes que se iniciem as atividades primárias de construção da iteração 5. Tipicamente, as atividades que precisam acontecer antes que se inicie uma iteração estão relacionadas à adoção de práticas de UED como análise de usuários e suas tarefas. Na proposta de Hodgetts [2005], essa análise é realizada antes de cada iteração.

Holzinger et al. [2005] apresentam um método chamado *eXtreme Usability* (XU) que combina práticas de XP e de UCD. As idéias dos autores foram utilizadas no ensino

de Engenharia de Software, e segundo eles, essa é a maior contribuição do método. Os autores planejam aplicar o método em cenários reais, com o objetivo de coletar experiências e descobrir se o método se tornaria compreensivo diante da combinação entre teoria e prática.

O XU foi desenvolvido baseado nas observações dos autores que perceberam uma grande semelhança entre os métodos utilizados pelos processos ágeis e pelos processos de usabilidade. Apesar de possuírem objetivos diferentes, ambos os métodos são baseados em iterações e participação constante do usuário. Essa perspectiva resultou na adoção do modelo em espiral para o XU.

Sousa et al. [2005] apresentam o processo UPI, um processo de desenvolvimento de software envolvendo usabilidade, produtividade e integração. O processo absorve práticas da Engenharia de Software e da Interação Humano-Computador (*Human-Computer Interaction* - HCI) com três principais objetivos.

O primeiro deles é ajudar profissionais das duas áreas no desenvolvimento de sistemas interativos com qualidade e que contemplem aspectos de usabilidade. O segundo é fazer de HCI parte essencial dos processos de engenharia de software, facilitando a comunicação entre os profissionais dessas duas áreas e trazendo produtividade ao novo ambiente de trabalho. E o terceiro e último objetivo é descrever as bases para geração de UIs através da integração dos conceitos de HCI no processo, ao invés de depender apenas da experiência dos *designers* de UI.

Sousa et al. [2005] não tratam diretamente da aplicação de HCI em métodos ágeis, no entanto suas contribuições são bastante positivas para este trabalho ao planejar o desenvolvimento de um processo baseado em produtividade e integração, princípios que também norteiam o desenvolvimento ágil.

Meszaros & Aston [2006] apresentam um trabalho que descreve como partiram das práticas aceitas e já experimentadas de uma empresa para o desenvolvimento de software; inicialmente utilizando métodos ágeis e em um segundo momento, introduzindo testes de usabilidade nos métodos já experimentados. Ao longo do trabalho, os autores foram obtendo respostas sobre o que de fato funcionava ao utilizarem as duas abordagens.

Dentre as soluções encontradas, Meszaros & Aston [2006] utilizaram testes de usabilidade baseados em protótipos de papel e versões preliminares do software foram adicionadas ao processo de desenvolvimento ágil para um segundo *release*, resultando em uma significativa redução de re-trabalho relacionado à usabilidade. O protótipo de papel tornou-se uma representação tangível da visão do projeto que foi usada de várias formas que contribuíram para o seu sucesso.

Lievesley & Yee [2006] discutem o papel do *designer* de interação durante o

processo de desenvolvimento ágil de software. As experiências relatadas no artigo mostram a integração de uma equipe de *designers* de interação em um projeto que utilizaria Scrum como método ágil. O artigo apresenta conclusões parciais do trabalho, no entanto já ressalta que a flexibilidade da orientação a objetos e do desenvolvimento ágil está mais bem alinhada com as práticas de *design* do que os processos tradicionais da engenharia de software. Utilizando tais abordagens, os *designers* podem produzir mais cedo interfaces que serão corretamente interpretadas pela equipe de desenvolvimento, provendo uma visão clara e compartilhada, considerada um elemento essencial pelos métodos ágeis.

Lee & McCrickard [2007] descrevem os resultados do trabalho inicialmente proposto em Lee [2006]. O trabalho apresenta um processo de desenvolvimento que une as práticas de um método ágil, o XP, e as práticas de um processo de engenharia de usabilidade, o SBD (*Scenario-Based Design*). Dentre outras contribuições, o SBD apresenta uma solução interessante para a parte de avaliação e testes de usabilidade. Como o XP trabalha com desenvolvimento orientado a testes, os desenvolvedores escrevem os casos de teste para validar as funcionalidades de novas classes e métodos antes de qualquer outra implementação. Esse conjunto de testes é usado para validar a funcionalidade do código.

Ainda de acordo com os autores Lee & McCrickard [2007], a partir das visões de arquitetura definidas pelo SBD, um processo similar poderia ser utilizado para adicionar avaliações de usabilidade ao processo de teste. Esse processo manteria consistência entre a arquitetura já definida e os códigos gerados pelos desenvolvedores. As avaliações analíticas e empíricas envolvidas no conjunto de testes validariam os requisitos presentes no SBD e identificariam pontos de melhoria que deveriam ser implementados para a interface.

O trabalho de Ferreira et al. [2007] afirma que os projetos ágeis necessitam projetar a interação do usuário (*User Interaction - UI*). No entanto, a integração de UI no desenvolvimento ágil ainda não está bem entendida. Isso acontece porque tanto o desenvolvimento ágil quanto o *design* UI são iterativos, porém enquanto a iteração nos métodos ágeis é focada no código, as iterações em UI se dão tipicamente apenas na interface de usuário, usando protótipos de baixa tecnologia.

Apesar das diferenças, métodos ágeis e UI enfatizam testes. Os métodos ágeis envolvem testes automatizados do código, enquanto testes em UI devem ser feitos por profissionais especializados ou idealmente por usuários potenciais. Tendo essas variáveis em vista, o trabalho de Ferreira et al. [2007] apresenta um estudo teórico qualitativo de projetos ágeis reais envolvendo significativa participação de *design* UI. Os resultados são que iterações ágeis facilitam os testes de usabilidade, permitem desenvolvedores

de software incorporarem resultados de seus testes em iterações subseqüentes e o mais importante, podem melhorar significativamente a qualidade do relacionamento entre profissionais de UI e desenvolvedores de software.

Detweiler [2007] considera a experiência do usuário (*User eXperience* - UX) envolvida no desenvolvimento de projetos de software que seguem a metodologia ágil. O autor apresenta idéias sobre como antecipar potenciais mudanças que as abordagens ágeis devem colocar para a equipe de usabilidade. A proposta envolve a divisão do trabalho de UCD em três fases iterativas. A primeira busca entender o usuário e suas necessidades, a segunda define as iterações e a última trabalha com o desenho da interface. Essas três fases permeiam o desenvolvimento baseado em UCD e acontecem paralelamente em forma de pequenos *releases*. Um conjunto de diretrizes e mudanças necessárias para as adaptações também é apresentado.

Aureliano [2007] apresenta um processo de IHC denominado eXCeeD (*eXtreme Communication-Centered Design*). O processo propõe práticas de usabilidade mais simplificadas para utilização em ambientes de desenvolvimento ágeis, como o ambiente XP. O eXCeeD busca unir o apoio à reflexão oferecido pela teoria da Engenharia Semiótica [de Souza, 2005] com a característica de agilidade de técnicas de prototipação de interfaces, incorporando os valores e princípios dos métodos ágeis, mais especificamente do método XP.

Mais recentemente, foram publicados trabalhos como os de Singh [2008], que propõe uma variante do método Scrum centrada em usabilidade e denominada U-SCRUM. O artigo afirma que o método Scrum não define atividades de usabilidade, bem como os demais métodos ágeis, conforme já destacamos em nosso trabalho. O U-SCRUM agrega o papel de *Usability Product Owner*, responsável pelas atividades propostas pelo método. Como resultado de validação, o trabalho aplicou o método a projetos orientados a negócios. A autora conclui que a abordagem utilizada pelo U-SCRUM melhora consideravelmente o nível de usabilidade dos produtos desenvolvidos, benefício apontado, inclusive, pelos usuários finais e pelos demais *stakeholders*.

Wolkerstorfer et al. [2008] descreve adaptações do método XP para as abordagens de HCI (*Human Computer Interaction*). Em especial, são propostas atividades como: análise de usuários, uma variação da técnica Personas [Cooper & Reimann, 2003] que os autores chamam de *Extreme Personas*, avaliações de usabilidade por especialistas, testes de usabilidade e avaliações automatizadas de usabilidade.

O objetivo do trabalho de Wolkerstorfer et al. [2008] foi avaliar a adaptabilidade dessas abordagens de HCI ao contexto do desenvolvimento de software com métodos ágeis. Os autores aplicaram a proposta em projetos de desenvolvimento de aplicações multimídia para dispositivos móveis. A experiência dos autores com o projeto durou

cerca de 3 anos e também apontou resultados finais bastante positivos, tanto na integração dos profissionais de HCI com a equipe de desenvolvimento ágil, quanto em relação à mudança cultural implantada na equipe a partir do modelo proposto. Essa mudança cultural é bem destacada pela filosofia ágil. Beck [1999] afirma que a mudança cultural nas equipes que desejam adotar XP é o primeiro passo para o sucesso do método. Essas mudanças envolvem repensar a maneira como o software é desenvolvido, o objetivo e os resultados esperados a curto, médio e longo prazo pelo cliente e a maneira com a qual o papel das pessoas ao longo do desenvolvimento é encarado. Essa mesma mudança cultural é destacada também pelos processos de usabilidade e suas abordagens de UCD.

Fox et al. [2008] discute uma pesquisa com participantes baseada em três abordagens combinando atividades de UCD e atividades ágeis. As abordagens se dividem em: Generalista, Especialista e Híbrida. Os autores conduziram em profundidade entrevistas semi-estruturadas com 10 participantes com diferentes perfis e experiências de atuação em equipes de desenvolvimento. Como resultado, os autores perceberam que grande parte das práticas utilizadas pelos participantes entrevistados eram comuns no cenário de integração das abordagens ligadas a UCD e métodos ágeis.

Ao final do mapeamento qualitativo das atividades descritas pelos participantes, foram definidas as três abordagens propostas pelo artigo e discussões foram levantadas acerca das técnicas e adaptações já realizadas em outros trabalhos aqui relacionados.

Ungar [2008] apresenta um relatório de experiência prática na indústria integrando atividades de UCD a uma equipe ágil. A experiência ocorreu através de um *workshop* em um projeto utilizando *Design studio*. *Design studio* é uma técnica composta de 4 componentes principais: Pesquisa, Projeto, *Studio* e Participantes.

A execução da técnica nesse trabalho permitiu a identificação de necessidades através de entrevistas com os usuários finais, a criação de protótipos de baixa fidelidade em quadro branco (prática estimulada pelos métodos ágeis) e o desenvolvimento de *sprints* de exploração antes do desenvolvimento do produto. Um *sprint* no método Scrum é equivalente a uma iteração no método XP, e representa uma porção de software funcional que agrega valor ao negócio do cliente. Ungar [2008] relata que a técnica estimula a criatividade e a liberdade de criação do pessoal envolvido e recomenda a experiência de utilização da técnica em projetos onde o contato inicial com as atividades de UCD são necessárias.

Najafi & Toyoshiba [2008] trabalha os conceitos de UED e desenvolvimento ágil em dois estudos de caso na VeriSign. O primeiro estudo de caso é o desenvolvimento de um *web site* de vendas a varejo para o consumidor final, utilizando profissionais de UED na equipe ágil. O segundo foi o *re-design* de um *web site* existente, porém sem

a participação da equipe de UED no planejamento dos *sprints*. Como era esperado, o projeto desenvolvido com a presença da equipe de UED apresentou nítidos resultados positivos na qualidade da interface do produto final. Já o projeto que trabalhou com uma equipe sem esses profissionais, apresentou problemas na conformidade dos requisitos e em consequência disso, não satisfaz as necessidades de negócio que haviam sido determinadas para a aplicação.

Já em Broschinsky & Baker [2008], os autores fazem uso especificamente da técnica de Personas [Cooper & Reimann, 2003] incorporada ao método XP em um projeto da LANDesk Software. As contribuições do trabalho desses autores são importantes para reafirmar o sucesso da integração de uma técnica consagrada para Análise de Usuários a um método ágil como o XP. O método XPu aqui proposto também faz uso de Personas como técnica de escolha para a Análise de Usuários e contribui com mais uma experiência de uso dessa atividade no contexto aplicado.

Finalmente, em Barbosa et al. [2008] é discutida uma abordagem para a institucionalização de práticas de usabilidade com o objetivo de apoiar as organizações de desenvolvimento de software na realização de uma melhoria de seus processos, baseando-se em uma integração dos conceitos de desenvolvimento e gestão ágil, maturidade em usabilidade e gestão de pessoas. Os autores descrevem um estudo de caso realizado com o intuito de aplicar a integração proposta na prática. Nele, incluíram-se novas atividades na rotina das equipes de desenvolvimento e gerentes de uma organização com foco de trabalho no desenvolvimento ágil. Os resultados atingidos com a experiência descrita foram validados e contribuíram para uma mudança na cultura de desenvolvimento da organização.

2.3.1 Considerações

De uma maneira geral, os trabalhos relacionados acima se dividem em 3 grandes grupos. O primeiro deles [Aureliano, 2007; Constantine & Lockwood, 2003; Hodgetts, 2005; Holzinger et al., 2005; Lee & McCrickard, 2007; Meszaros & Aston, 2006; Patton, 2002; Singh, 2008; Sousa et al., 2005; Vasconcelos et al., 2003; Wolkerstorfer et al., 2008] apresenta abordagens centradas na adaptação das práticas de usabilidade aos métodos ágeis. Alguns deles partem de modelos bem definidos da área de usabilidade, porém sem apresentar claramente a integração das atividades de usabilidade com as atividades de desenvolvimento. Outros não tratam das atividades gerenciais de um processo de usabilidade ou tratam apenas de aspectos específicos, como o trabalho de Meszaros & Aston [2006] que está focado na discussão de testes de usabilidade para métodos ágeis.

O método XPu aqui apresentado se insere nesse primeiro grupo de trabalhos ao propor uma adaptação do processo Praxis-u ao contexto de desenvolvimento do método XP. Essa característica faz com que este trabalho tenha pontos de convergência com todos os trabalhos deste grupo, no entanto se difere de todos eles ao apresentar uma análise crítica sobre a caracterização de agilidade da proposta, confrontando-a com uma experiência prática de desenvolvimento. Além disso, o XPu ao adaptar as atividades do Praxis-u manteve suas atividades gerenciais, característica negligenciada pelos demais trabalhos desse grupo.

Outros trabalhos [Armitage, 2004; Barbosa et al., 2008; Detweiler, 2007; Ferreira et al., 2007; Kane, 2003; Lievesley & Yee, 2006; McInerney & Maurer, 2005] formam um segundo grupo e se concentram em discussões mais genéricas sobre como adaptar as práticas e atividades de usabilidade em um contexto ágil, sem propor uma solução completa (método), adaptável a um processo de software. O trabalho de Lievesley & Yee [2006], inclusive, foca a sua discussão primordialmente no papel esperado do Projetista de Interação em um método ágil.

Por fim, um terceiro e último grupo de trabalhos [Broschinsky & Baker, 2008; Fox et al., 2008; Najafi & Toyoshiba, 2008; Ungar, 2008] discute estudos de caso na indústria e em projetos específicos sobre a adoção de práticas de usabilidade em equipes ágeis.

Capítulo 3

Metodologia de Pesquisa

Este capítulo apresenta detalhadamente a metodologia de pesquisa utilizada para a realização deste trabalho. A primeira seção classifica a pesquisa realizada baseando-se no seu tipo em cada uma das dimensões metodológicas. Já na segunda seção são abordados procedimentos metodológicos adotados desde a concepção e revisão bibliográfica sobre o tema até procedimentos relacionados à proposição, execução e validação das contribuições do método XPu aqui apresentadas.

Espera-se que através deste capítulo seja possível a outros pesquisadores da área compreender o método de pesquisa utilizado, bem como reproduzir em trabalhos futuros os procedimentos aqui adotados.

3.1 Tipo de Pesquisa

Jung [2004] e Marconi & Lakatos [2003] afirmam que a pesquisa científica pode ser classificada em 4 dimensões: quanto à sua natureza (básica ou fundamental, aplicada ou tecnológica); quanto aos seus objetivos (exploratória, descritiva ou explicativa); quanto aos procedimentos (experimental, operacional ou estudo de caso) e quanto ao local de realização da mesma (laboratório ou campo). Jung [2004] ainda aborda a importância de classificar a pesquisa quanto ao tempo de aplicação dos estudos (transversal ou longitudinal).

Na visão dos autores, quanto à natureza, a pesquisa básica objetiva entender ou descobrir novos fenômenos, com foco em conhecimentos básicos e fundamentais, enquanto a pesquisa tecnológica objetiva a aplicação dos conhecimentos básicos na geração de novos produtos, processos, patentes e serviços.

Quanto aos objetivos, a pesquisa exploratória visa a descoberta de teorias e práticas que modificarão as existentes - inovações tecnológicas; a pesquisa descritiva

tem a finalidade de observar, registrar e analisar os fenômenos ou sistemas técnicos - identificação, registro e análise de características, fatores e ou variáveis; e a pesquisa explicativa objetiva ampliar generalizações, definir leis mais amplas, modelos teóricos - síntese, teorização e reflexão.

Quanto aos procedimentos tem-se que a pesquisa experimental busca a descoberta de novos materiais, métodos, técnicas, protótipos de software – ensaios e estudos de laboratório, modelagem, simulação, sistemas e circuitos; a pesquisa operacional trata da busca do ótimo, uso de ferramentas estatísticas, métodos matemáticos da otimização, busca do melhor resultado - condição ótima; e o estudo de caso permite investigar um fenômeno dentro de um contexto local e real, estudar o fenômeno, dar limites, definir claramente - entender como e por que as coisas funcionam.

Quanto ao local de realização da mesma, a pesquisa em laboratório e/ou em campo é fundamentalmente a última etapa de todo o processo metodológico. Entende-se como pesquisa em laboratório aquela onde ocorre a possibilidade de se controlar as variáveis que possam intervir no experimento. Já como pesquisa em campo, aquela onde não há a possibilidade de controle das variáveis. Trata-se do local em condições reais onde ocorrem os fenômenos.

Finalmente, quanto ao tempo de aplicação do estudo, o estudo transversal é aquele que se realiza em determinado instante de tempo (t) naquele exato corte temporal e o estudo longitudinal é aquele no qual os dados são coletados ao longo do tempo, com obtenção sistemática e lenta de resultados.

A pesquisa realizada para a definição do método XPu é de natureza aplicada ou tecnológica, já que objetiva a aplicação das práticas existentes nos métodos ágeis e nos processos de usabilidade para a definição de um método integrado, visando adequação ao objetivo inicialmente delineado. Por estar voltada para a identificação, o registro e a análise de características, fatores e ou variáveis, seu objetivo é descritivo.

Com o objetivo de validar o método proposto, o XPu foi submetido a um estudo de caso e, portanto, com relação aos procedimentos, essa é a classificação da pesquisa. Considerando que os resultados desse estudo de caso foram coletados em um ambiente controlado, simulando o ambiente real de desenvolvimento de uma equipe ágil, essa pesquisa pode ser classificada como realizada em laboratório, quanto ao local de realização da mesma. Por fim, o tempo de aplicação desse estudo foi longitudinal, pois as coletas de dados para o estudo de caso ocorreram ao longo do desenvolvimento do produto de software proposto.

3.2 Procedimentos Metodológicos

As atividades relacionadas à pesquisa realizada por este trabalho iniciaram-se em 2007. Com o objetivo de compreender os problemas de pesquisa em aberto na área de métodos ágeis e usabilidade, foram estudadas as principais publicações em conferências e periódicos especializados na tentativa de estabelecer uma abordagem relevante ao tema.

No desenvolvimento deste trabalho, optou-se pela adaptação do processo de usabilidade Praxis-u, já que este vinha sendo utilizado há alguns anos nas pesquisas direcionadas à área de usabilidade no DCC/UFMG. Além disso, o Praxis-u é um processo consolidado e que foi definido baseado em trabalhos de grande aceitação na indústria atual. Em paralelo, também optamos por trabalhar com o método ágil XP, visto que este é o método que mais tem recebido atenção das comunidades de pesquisa e dos diversos setores da indústria de software atual.

O primeiro desafio estabelecido pelo trabalho, portanto, era o de viabilizar a utilização de práticas de usabilidade (oriundas de modelos tradicionais e centrados em documentação) de maneira adaptada a um ambiente de desenvolvimento ágil. Para isso, inicialmente as práticas de usabilidade do Praxis-u foram analisadas e sugestões foram propostas para a realização de suas atividades com foco em práticas consagradas de IHC, de forma que estas estivessem alinhadas à filosofia ágil de desenvolvimento.

Em seguida, essas práticas foram adaptadas ao contexto ágil do XP, de maneira que pudessem ser prontamente integradas ao ciclo de vida de um projeto. Com essas adaptações, foram observadas restrições e limitações relativas ao Praxis-u. Naturalmente, algumas atividades foram retiradas para que o XP não fosse descaracterizado em alguma instância. O método resultante dessa adaptação foi chamado de XPu.

Após essa etapa, o segundo desafio era garantir (pelo menos do ponto de vista conceitual) que as atividades adicionadas e sustentadas pelo método XPu não fariam com que o método XP modificado se aproximasse de um modelo de desenvolvimento tradicional e burocrático, ou seja, era necessário garantir que o método XPu não descaracterizaria o modelo de “agilidade” definido pelo XP e pelos demais métodos ágeis.

Foi necessário, portanto, encontrar definições na literatura atual que estabelecessem de maneira clara e objetiva o que caracterizava o conceito de agilidade em processos. Devido à prematuridade das pesquisas na área de métodos ágeis, ainda é difícil encontrar uma boa base de trabalhos que sustente esse conceito. Os principais trabalhos pesquisados que formam o estado da arte atual são os

trabalhos de Abrantes & Travassos [2007a] e seu artigo resultante publicado em 2007 [Abrantes & Travassos, 2007b], além dos trabalhos de Stamelos & Sfetsos [2007] e Dybå & Dings [2008].

A questão básica de pesquisa, segundo Abrantes & Travassos [2007b], é determinar o que caracteriza um método de desenvolvimento de software como sendo um método ágil. O objetivo do trabalho dos autores era chegar a um conjunto básico de características que são necessárias para que o método possa ser classificado com método ágil, investigando através de revisão sistemática de literatura (estudo secundário), quais são as características de agilidade no contexto dos métodos ágeis.

Abrantes & Travassos [2007b] adotaram uma abordagem que estrutura a questão de pesquisa em 4 elementos básicos: população, intervenção, comparação e resultado. De acordo com o protocolo estabelecido em Pai et al. [2004] e tendo em vista que o objetivo da pesquisa era o de realizar uma caracterização da área, não houve comparação e nem foi possível estabelecer uma meta-análise dos dados. Dessa forma, os autores do artigo classificaram o estudo secundário como uma *quasi-revisão* sistemática de literatura.

Não era objetivo do trabalho em questão investigar características de métodos ágeis específicos, mas de uma maneira geral, identificar quais são as propriedades ou características desse grupo de métodos para desenvolver software e assim obter um conjunto de características desejáveis para um método ser considerado ágil.

Através de uma string de busca estruturada com as principais palavras chaves que esperava-se para os artigos da área, Abrantes & Travassos [2007b] realizaram uma consulta nas principais bibliotecas digitais disponíveis no portal da CAPES e como resultado obtiveram mais de 1000 trabalhos relacionados. Após uma série de filtros, esses trabalhos foram organizados e selecionados baseado na relevância do seu conteúdo e foi formulada uma tabela com as principais características identificadas para os métodos ágeis em cada um deles.

As características identificadas foram classificadas baseado na frequência com que apareciam nos trabalhos, considerando a faixa de ano de publicação dos artigos nos quais elas estavam presentes. Por fim, os autores propuseram em uma sentença clara e objetiva uma proposta de caracterização dos métodos ágeis que, de acordo com eles, podem permitir algum direcionamento nos trabalhos e pesquisas relacionadas à obtenção de agilidade em métodos de desenvolvimento de software.

A partir dessa definição, portanto, o XPu foi inspecionado em relação às suas características ágeis através do trabalho de Abrantes & Travassos [2007b]. Os resultados dessa inspeção foram fundamentais para buscar assegurar que a condução dos trabalhos de pesquisa e definição do método XPu não descaracterizava o método

XP em sua agilidade.

Em seguida, o método foi avaliado em um estudo de caso realizado no início deste ano com dois profissionais recém-graduados na área de Sistemas de Informação. O XPu foi utilizado no desenvolvimento de um pequeno projeto de um sistema interativo de consulta ao acervo de uma loja especializada em locação de DVDs. O objetivo principal era observar a execução das atividades de usabilidade propostas em um ambiente ágil de desenvolvimento baseado no método XP.

Para coletar dados que indicassem medidas de eficiência na adoção do método XPu, o projeto foi desenvolvido pelos dois profissionais inicialmente utilizando o método XP padrão e, em seguida, utilizando o método XPu. Métricas relativas ao esforço de desenvolvimento utilizando o método XP padrão e utilizando o método XPu foram coletadas com o objetivo de analisar o impacto da adoção das práticas de usabilidade sugeridas e confrontá-las com a melhoria na qualidade da interface produzida em ambos os casos. Os resultados foram úteis para avaliar a eficiência do método e apontar as dificuldades encontradas durante a realização de suas atividades. O capítulo 5 dessa dissertação descreve em detalhes o estudo de caso realizado.

Capítulo 4

O método XPu

Este capítulo descreve o método de usabilidade proposto para o XP. O XPu (*eXtreme Programming with usability* ou Programação eXtrema com usabilidade) foi definido a partir do processo de usabilidade Praxis-u, proposto por Pádua et al. [2006], utilizado para ensino e pesquisa no Departamento de Ciência da Computação da Universidade Federal de Minas Gerais e integrado ao Praxis [Paula-Filho, 2003].

Diferente da abordagem utilizada por Pádua et al. [2006], a adaptação utilizada para o método XPu não define um fluxo para tratar as atividades de usabilidade dentro do método XP. Essa opção é facilmente justificada pelo fato do método XP não trabalhar com os conceitos de fases, fluxos e disciplinas, utilizados no desenvolvimento de software dirigido por processos. Para realizar tal adaptação, portanto, optamos pela integração paralela das atividades propostas para o XPu com as atividades já habitualmente desenvolvidas pelo XP.

Assim como no Praxis-u, as atividades propostas para o XPu estão divididas em atividades técnicas e gerenciais. As atividades técnicas compreendem todas as práticas necessárias à execução correta do processo de usabilidade, ao passo que as atividades gerenciais garantem o efetivo controle do andamento e da execução dessas práticas, intervindo sempre que necessário na sua execução.

O método está apresentado nas seções a seguir por um conjunto de visões que abordam aspectos como projeto, iteração e desenvolvimento. Essa divisão em visões foi estabelecida para o XP em Wells [2006] e mantida por este trabalho, adaptando apenas as visões que sofreram algum tipo de modificação ou adaptação devido ao método XPu.

Para cada visão, são apresentados diagramas de atividades da UML referentes às atividades do método XP tradicional e às adaptações realizadas pela integração das atividades do XPu. Ao abordar cada uma das atividades, são discutidas as técnicas de escolha, suas possíveis situações de uso, bem como os insumos consumidos e os artefatos

produzidos por cada uma delas. Assim como na representação padrão do diagrama de atividades da UML, os retângulos ovalados identificam as atividades dos métodos XP e XPu e as setas identificadas representam os fluxos de insumos consumidos ou artefatos produzidos ao longo dessas atividades. Para melhor visualização dos diagramas, em cada uma das visões, eles foram divididos em duas partes complementares uma à outra, ou seja, a Parte 2 de um diagrama é a continuação da sua Parte 1 e não sua especialização ou detalhamento. Todos os diagramas foram gerados pela ferramenta Open Source StarUML.

4.1 Visão de Projeto

A visão de projeto apresenta as atividades do método em mais alto nível de abstração. Esse nível de abstração corresponde à definição dos *releases* que irão compor o produto a ser desenvolvido. O fluxo dessas atividades e a interação entre elas são apresentados nas Figuras 4.1 e 4.2.

4.1.1 Atividades Técnicas

As Atividades Técnicas formam um conjunto de atividades necessárias à construção de um sistema interativo com qualidade no que se refere às suas características de usabilidade. As Atividades Técnicas para a Visão de Projeto do XPu são apresentadas em detalhes nas seções seguintes.

4.1.1.1 Análise de Usuários

As Atividades Técnicas do método XPu se iniciam com a **Análise de Usuários** que é feita a partir de **Personas** [Cooper & Reimann, 2003]. Essa técnica foi desenvolvida por Alan Cooper com o objetivo de caracterizar grupos de usuários finais através da observação direta destes em seus locais habituais de trabalho. Com a utilização de Personas é possível descrever protótipos típicos desses usuários finais em um determinado contexto que é de interesse da aplicação a ser desenvolvida.

Segundo Cooper & Reimann [2003], para construir Personas que sejam, de fato, representativas das várias classes de usuários é necessário seguir as seguintes etapas:

1. Identifique as variáveis comportamentais (geralmente atividades, atitudes, habilidades, motivações).
2. Mapeie os entrevistados com as variáveis comportamentais (classificar de acordo com os critérios levantados na etapa 1).

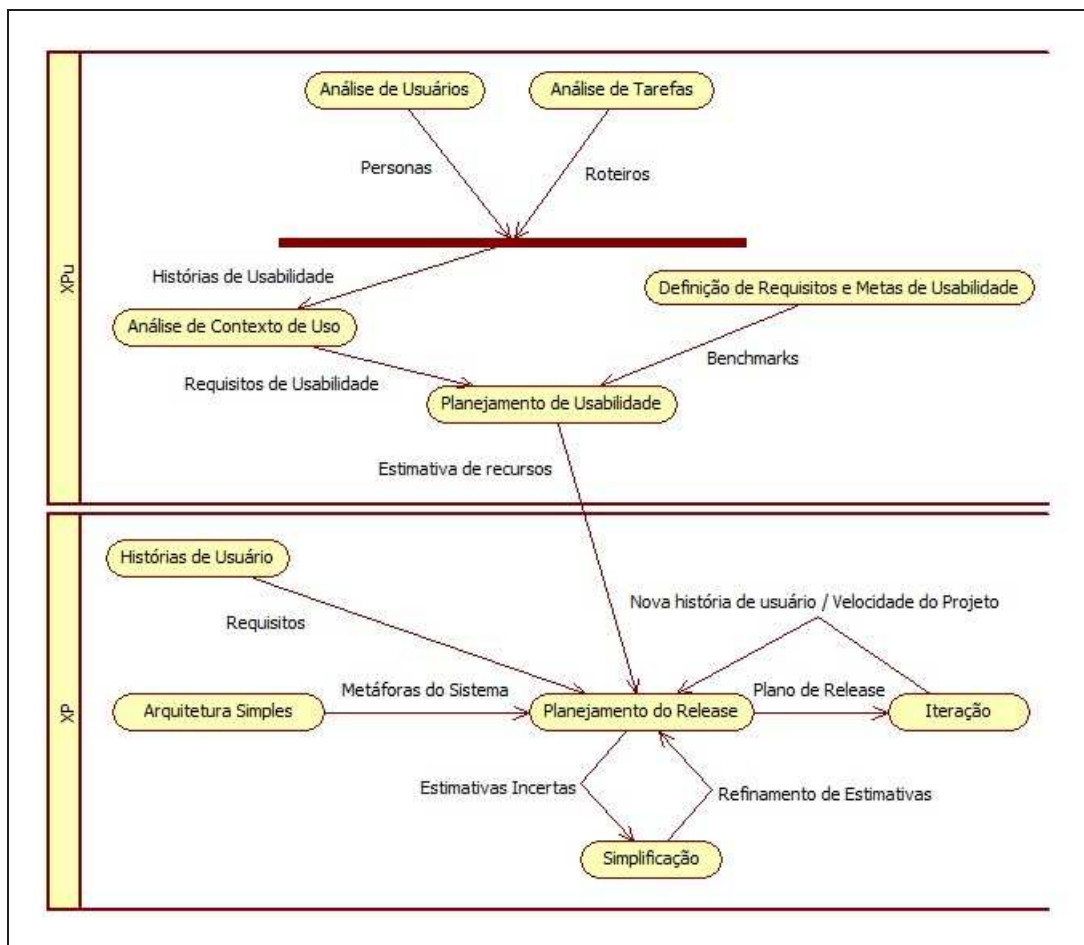


Figura 4.1. Visão de Projeto do XPU e sua integração com o XP - Parte 1.

3. Identifique padrões comportamentais significativos.
4. Sintetize características e objetivos relevantes (o autor enfatiza a característica de descobrir quais são os objetivos do usuário ao usar o produto).
5. Identifique as variáveis comportamentais (geralmente atividades, atitudes, habilidades, motivações).
6. Verifique completude e redundância (refinar as Personas).
7. Expanda descrições de atributos e comportamentos.
8. Defina tipos de Personas (principais, secundárias, suplementares).

Um exemplo de Personas para mapear um usuário ou um grupo de usuários típicos de um sistema é apresentado abaixo:

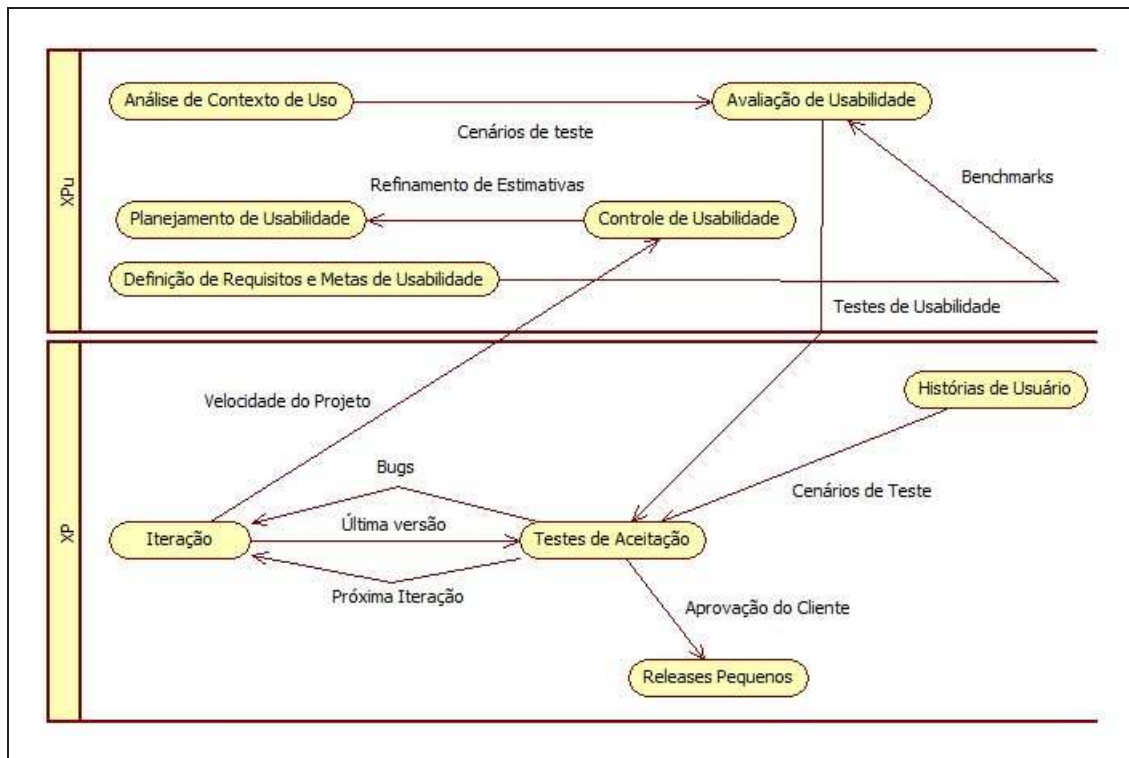


Figura 4.2. Visão de Projeto do XPu e sua integração com o XP - Parte 2.

Felipe Quintão é um estudante de 19 anos com segundo grau completo e atualmente vive em Belo Horizonte. Felipe mora com os pais e é um típico adolescente de classe média que passa boa parte do tempo utilizando computadores, acessando a internet e de olho nas últimas novidades eletrônicas do mercado. Felipe utiliza a internet na maior parte do tempo para acessar e-mails, ler blogs, participar de chats e de comunidades de relacionamento, além de usar sites de busca para saber mais sobre assuntos diversos.

Felipe acabou de passar no vestibular e está apreensivo com a nova fase da sua vida que está por vir. Como este é o primeiro contato de Felipe com o ambiente acadêmico, ele desconhece o processo de registro e controle de alunos da universidade. Felipe obteve informações com amigos sobre o que fazer, quando e onde para garantir sua vaga na UFMG. Ele precisa preencher corretamente todos os campos obrigatórios para complementar suas informações cadastrais, além de gerar um comprovante dessas informações e apresentá-lo no dia da matrícula.

Como é possível perceber, a Persona relatada acima descreve perfeitamente os hábitos, a proficiência e a expectativa de um usuário ou um grupo representativo de usuários que utilizarão o sistema a ser desenvolvido. Ou seja, a partir da Análise de Usuários realizada, podemos identificar as principais características que o sistema deve contemplar, de forma a atender os anseios e expectativas desses usuários em relação ao sistema.

O método XPu sugere que sejam adotadas diversas Personas seguindo o protocolo descrito em Cooper & Reimann [2003], de modo a mapear e identificar o maior número possível de grupos de usuários representativos que utilizarão o sistema. Essa atividade é realizada pelo Projetista de Interação, em paralelo com as atividades técnicas do XP relativas à descrição de Histórias de Usuário, realizadas pelos Usuários e Gerentes de Produto e que mapeiam os requisitos funcionais do sistema.

4.1.1.2 Análise de Tarefas

Paralelamente, a **Análise de Tarefas** é feita através do uso de **Roteiros** [Rosson & Carroll, 2002]. Roteiros fazem parte de uma técnica de IHC que propõe uma narrativa textual real, concreta e rica em detalhes contextuais para descrever a execução de uma tarefa por um perfil de usuário típico. Os Roteiros, portanto, representam situações de uso e são utilizadas para conhecimento das tarefas realizadas por usuários, bem como suas necessidades em relação ao produto.

Roteiros são bastante úteis para representar e gerar alternativas de *design* relacionadas a situações de uso do sistema, além de obter *feedback* do usuário sem o custo de se construir um protótipo funcional. Os Roteiros, de uma maneira geral:

- são flexíveis;
- constituem uma representação provisória;
- focam as consequências para usabilidade (qualidade de uso) de propostas de *design* específicas;
- mostram o uso do sistema em detalhes;
- são escritos em linguagem natural, compreensível por todos os *stakeholders*;
- e provocam discussões e perguntas do tipo “e se”.

Segundo Rosson & Carroll [2002], os Roteiros são constituídos por:

- Atores;

- Contexto;
- Eventos;
- Objetivos;
- Planos;
- Ações;
- e Avaliação.

Um exemplo de Roteiros com um trecho para mapear uma tarefa ou um grupo de tarefas típicas de um sistema realizadas por um usuário representativo é apresentado abaixo:

Felipe acabou de passar no vestibular e obteve informações sobre o que fazer, quando e onde para conseguir sua vaga na universidade. Ele precisa, em primeiro lugar, preencher a Ficha de Cadastro (exclusivamente pela internet), com o número de inscrição e a senha usados no vestibular. Além disso, ele precisa efetuar o seu registro discente no local e data indicados pelo departamento, levando todos os documentos necessários.

Se Felipe esquecer sua senha, ele pode recuperá-la na página do departamento responsável, no documento onde ele pegou as outras informações. Felipe deve entrar na página indicada e, após preencher as suas informações, ele deve gerar um comprovante de preenchimento dos dados cadastrais que deverá ser impresso e apresentado no momento da confirmação do registro discente.

No período determinado pela universidade, Felipe se dirige para o local indicado e enfrenta uma enorme fila de pessoas que esperam ansiosamente a sua vez de confirmar o registro. Quando atendido, ele entrega todos os documentos necessários para a confirmação. A atendente recebe os documentos e os compara com as informações que ele preencheu na Ficha de Cadastro. Várias situações podem acontecer nesse período: algum documento pode estar errado, os dados que ele preencheu podem não estar corretos, ou ele pode não conseguir comprovar o bônus que declarou ter direito, podendo com isso até perder sua vaga. Enquanto está sendo atendido, Felipe fica bastante apreensivo

para que tudo dê certo e ele venha a ser tornar aluno da universidade.

Como é possível perceber, a técnica de Roteiros permite realizar a Análise de Tarefas com clareza para identificar os possíveis passos e alternativas pensadas por um usuário típico para realizar uma determinada tarefa no sistema. Para o método XPu, a utilização de Personas associadas aos Roteiros é um importante mecanismo de descrição detalhada de usuários típicos realizando tarefas típicas e que permite identificar com facilidade a expectativa desses usuários em relação às situações de uso da aplicação.

4.1.1.3 Análise de Contexto de Uso

As Personas e os Roteiros gerados respectivamente nas atividades de Análise de Usuários e Análise de Tarefas formam as **Histórias de Usabilidade** que é o artefato gerado pela atividade de **Análise de contexto de uso**. As Histórias de Usabilidade são formadas pela descrição do fluxo de uma tarefa (Roteiro) sendo realizada por um usuário típico (Persona). Elas têm um papel similar às Histórias de Usuário já utilizadas pelo método XP padrão, no entanto são artefatos distintos e há duas diferenças importantes entre elas.

Ao contrário das Histórias de Usuário que são escritas pelo próprio Gerente de Produto com o auxílio direto do Usuário no método XP, as Histórias de Usabilidade são escritas pelo Projetista de Interação, também com o auxílio direto do Usuário. A opção de utilizar o Projetista de Interação para escrever as Histórias de Usabilidade se deve ao fato de que apesar das Personas e Roteiros serem escritos em linguagem natural e perfeitamente compreensível para todos os *stakeholders*, essas técnicas contém particularidades e objetivos que geralmente são de conhecimento apenas de um profissional da área de usabilidade. Por essa razão, o Projetista de Interação é o profissional mais indicado para realizar tal atividade. Essa prática não quebra o princípio de satisfação do cliente, visto que a participação do Usuário é fundamental tanto na descrição das Histórias de Usuário (escritas pelo Gerente de Produto) quanto na descrição das Histórias de Usabilidade (escritas pelo Projetista de Interação).

Uma segunda diferença importante entre as Histórias de Usabilidade e as Histórias de Usuário é que a primeira está voltada primordialmente à identificação de requisitos não-funcionais relacionados à usabilidade, enquanto que a segunda está voltada exclusivamente à identificação de requisitos funcionais.

Um exemplo de História de Usabilidade construída a partir da Persona e do Roteiro apresentados nas subseções anteriores é apresentado na Figura 4.3.

FELIPE QUINTÃO
ALUNO APROVADO NO VESTIBULAR



“Quero que tudo corra bem durante o meu registro como aluno da universidade.”

★★★
Computador, internet, tecnologia

★★★
Processos do registro acadêmico

19 anos, segundo grau completo.

Personas - Registro Discente

Atividades que realizará no Sistema:

- Complementa informações cadastrais.
- Gera comprovante de preenchimento das informações cadastrais.

Objetivos com o sistema:

- Preencher corretamente todos os campos obrigatórios para complementar as informações cadastrais.

Conhecimento de tecnologia e Internet:

- Tem facilidade com a Web;
- Costuma acessar e-mails, chats, blogs, sites de relacionamento e sites de busca;
- Utiliza o Internet Explorer e o Mozilla Firefox como navegadores.

Felipe acabou de passar no vestibular e obteve informações sobre o que fazer, quando, e onde, para garantir sua vaga na universidade. Ele precisa, em primeiro lugar, preencher a Ficha de Cadastro (exclusivamente pela Internet), com o número de Inscrição e a Senha usados no Vestibular. Além disso, ele precisa efetuar o seu registro discente no local e data indicados pelo departamento, levando todos os documentos necessários.

Se Felipe esquecer sua senha, ele pode recuperá-la na página do departamento responsável, no documento onde ele pegou as outras informações. Felipe deve entrar na página indicada e, após preencher as suas informações, ele deve gerar um comprovante de preenchimento dos dados cadastrais que deverá ser impresso e apresentado no momento da confirmação do registro discente.

No período determinado pela universidade, Felipe se dirige para o local indicado e enfrenta uma enorme fila de pessoas que esperam ansiosamente a sua vez de confirmar o registro. Quando atendido, ele entrega todos os documentos necessários para a confirmação. A atendente recebe os documentos e os compara com as informações que ele preencheu na ficha de cadastro. Várias situações podem acontecer nesse período: algum documento pode estar errado, os dados que ele preencheu podem não estar corretos, ou ele pode não conseguir comprovar o bônus que declarou ter direito, podendo com isso até perder sua vaga. Enquanto está sendo atendido, Felipe fica bastante apreensivo para que tudo dê certo e ele venha a se tornar aluno da universidade.

Figura 4.3. Exemplo de História de Usabilidade.

A atividade de Análise de Contexto de Uso define os **Requisitos de Usabilidade** e os Cenários de Teste que formam os insumos, respectivamente, para as atividades de Planejamento de Usabilidade e para a Avaliação de Usabilidade. Naturalmente, o Planejamento do *Release* durante as atividades do XP sofrerá o impacto da adição dos Requisitos de Usabilidade, já que o tempo e o custo necessário para a elaboração dessas tarefas devem ser contabilizados nas estimativas que são geradas com as atividades de planejamento.

4.1.1.4 Definição de Requisitos e Metas de Usabilidade

A atividade de **Definição de Requisitos e Metas de Usabilidade** permite que os Projetistas de Interação e Usuários estabeleçam *Benchmarks* que servirão de balizadores para avaliar a qualidade da usabilidade que está sendo entregue ao final de cada iteração. Os *Benchmarks* são gerados a partir dos Atributos de Usabilidade. Um Atributo de Usabilidade é uma característica geral de usabilidade a ser usada como critério para avaliação da interface. Esses atributos são essenciais, já que fornecem parâmetros para se medir a usabilidade de uma versão da interface. Segundo Nielsen [1993], há 5 Atributos de Usabilidade principais. São eles:

- **Facilidade de Aprendizagem:** capacidade com que o usuário começa a interagir rapidamente com o sistema logo na primeira vez que o utiliza.
- **Eficiência de Uso:** Grau de produtividade do usuário utilizando o sistema.
- **Facilidade de Memorização:** retenção, capacidade do usuário de voltar a utilizar o sistema após certo tempo sem precisar aprendê-lo novamente.
- **Prevenção de Erros:** medida do quanto o usuário pode ser induzido ao erro pelo sistema e o quanto pode se recuperar dele.
- **Satisfação Subjetiva:** medida do quanto o usuário se sente feliz de estar utilizando o sistema.

Baseado nesses e em outros atributos que possam ser considerados importantes para o Usuário em um contexto específico da aplicação, são estabelecidos instrumentos de medida para se obter valores quantitativos para um atributo particular de usabilidade. As medidas podem ser objetivas quando avaliam o desempenho observável do usuário durante a realização de tarefas usando a interface; ou subjetivas quando são obtidas baseadas nas opiniões do usuário sobre a interface.

As tarefas de *Benchmark* são utilizadas para guiar o usuário durante uma avaliação da interface. Segundo Gilb [1984], para cada uma das tarefas são estabelecidos níveis de desempenho que serão observados durante a avaliação. Os níveis de desempenho são estabelecidos em quatro escalas: atual, pior aceitável, alvo planejado e melhor possível. O nível atual corresponde ao nível utilizado pela corrente versão do sistema ou por sistemas concorrentes. O pior nível aceitável indica o mais baixo nível de desempenho do usuário com o qual ainda é possível obter algum grau de usabilidade. O nível alvo planejado indica o valor alvo que significa sucesso inquestionável de usabilidade para a interface. E, por fim, o melhor nível possível indica o limite superior ainda realístico, mostra o potencial de um atributo e serve como referência para futuras versões do sistema.

Ao realizar a Avaliação de Usabilidade com os usuários, os níveis de desempenho nas tarefas de *Benchmark* serão medidos e os resultados observados serão utilizados como balizadores para comparar os níveis previstos com os níveis reais alcançados na realização das tarefas por usuários típicos. Essa comparação fornece dados quantitativos e concretos sobre a qualidade da interface entregue pelo time.

Os *Benchmarks*, resultado da atividade de Definição de Requisitos e Metas de Usabilidade, são utilizados para planejar a Avaliação de Usabilidade e para realizar os testes que irão compor cada avaliação. Por essa razão, eles são utilizados como

insumos das atividades de Planejamento de Usabilidade e Avaliação de Usabilidade, como mostrado nas Figuras 4.1 e 4.2.

4.1.1.5 Avaliação de Usabilidade

Os **Cenários de Teste** são utilizados como principal insumo para a atividade de **Avaliação de Usabilidade**. Esses Cenários de Teste são definidos principalmente a partir dos Roteiros escritos durante a Análise de Tarefas. Segundo Rosson & Carroll [2002], os Roteiros podem ser utilizados para:

- Análise do problema:
 - “Quem são os usuários?”
 - “O que eles fazem?”
 - “Como?”
 - “Que problemas enfrentam?”
- Projeto:
 - “Como eu, *designer*, vou apoiar os usuários?”
 - “Como o sistema que estou projetando vai se encaixar no ambiente de uso?”
- Avaliação:
 - “Quem é o usuário?”
 - “O que quer fazer?”
 - “Por que?”

O método XPu utiliza a técnica de Roteiros com as três finalidades descritas acima em diferentes momentos do seu ciclo de vida. Para a Análise de Tarefas e durante a Visão de Projeto, o método utiliza os Roteiros com o objetivo de Análise do problema. Na Visão de Iteração, durante a construção dos *Templates* e Interfaces, o método utiliza os mesmos Roteiros com o objetivo de Projeto. Por fim, para a definição dos Cenários de Teste, o método utiliza os Roteiros com o objetivo de Avaliação. Com este objetivo, ao realizar a Avaliação de Usabilidade, o Projetista de Interação se baseia nas seguintes perguntas:

(1) “Quem é o usuário?”, ou seja, “Quais os requisitos para que a interface esteja adequada ao perfil do usuário?”;

(2) “O que quer fazer?”, ou seja, “A interface corresponde às expectativas do usuário para a realização de tarefas no sistema?” e

(3) “Por que?”, ou seja, “A interface contempla as respostas esperadas pelo usuário ao utilizar o sistema?”.

As técnicas de escolha para realização da Avaliação de Usabilidade no XPU possuem escopo aberto. Os recursos produzidos por eles são os **Testes de Usabilidade** que podem ser feitas através de avaliações preditivas, experimentais ou interpretativas, dependendo da abrangência da iteração que está sendo entregue aos Testes de Aceitação. As técnicas de avaliação utilizadas em diferentes momentos do ciclo de vida serão discutidas em detalhes na subseção Avaliação de Usabilidade da Visão de Desenvolvimento.

4.1.2 Atividades Gerenciais

As Atividades Gerenciais formam um conjunto de atividades necessárias ao controle e ao planejamento das atividades de usabilidade, com o objetivo de monitorar e garantir a efetiva execução das Atividades Técnicas de maneira correta ao longo do ciclo de vida de desenvolvimento. As Atividades Gerenciais para a Visão de Projeto do XPU são apresentadas em detalhes nas seções seguintes.

4.1.2.1 Planejamento de Usabilidade

As Atividades Gerenciais do método XPU se iniciam com o **Planejamento de Usabilidade** que gera um conjunto de **Estimativas de recursos** (relativos à usabilidade) que serão utilizados ao longo do desenvolvimento. Essa Estimativa de recursos envolve previsões de tempo e custo no Planejamento do *Release*. Ou seja, as estimativas relacionadas às atividades de usabilidade exercem influência direta na estimativa do prazo para entrega e no custo de *releases* e iterações.

As atividades de Planejamento de Usabilidade e Planejamento do *Release*, portanto, não ocorrem de maneira independente. A atividade de Planejamento de Usabilidade ocorre simultaneamente à atividade de Planejamento do *Release*, ou seja, nas reuniões de planejamento estão presentes todos os *stakeholders*, inclusive os Gerentes de Produto (fundamentais para o Planejamento do *Release*) e os Projetistas de Interação (fundamentais para o Planejamento de Usabilidade).

O método XP utiliza técnicas como o *Planning Poker* [Cohn, 2005] para gerar suas estimativas durante reuniões de planejamento como, por exemplo, o Planejamento do *Release*. O *Planning Poker* é uma técnica de estimativa em grupo e envolve um

jogo de cartas inspirado nas regras do tradicional jogo de Poker. Os membros do time utilizam as cartas de um baralho personalizado para dar o seu parecer em relação ao tamanho relativo das Histórias de Usuário contidas em uma determinada iteração. Para gerar uma escala de tamanhos válidos e possíveis, o time utiliza convenções de escala como, por exemplo, a série de Fibonacci.

A sugestão do método XPu para o Planejamento de Usabilidade é que o *Planning Poker*, jogado durante as estimativas de *release* e iteração do XP, envolvam a presença do Projetista de Interação para contribuir com o seu parecer relativo às estimativas de tamanho para as Histórias de Usabilidade e ao planejamento das Avaliações de Usabilidade. Para o planejamento das avaliações, por exemplo, podem ser utilizados métodos consolidados como o *framework* DECIDE [Preece et al., 2002] para auxiliar na geração de estimativas sobre o tempo que será gasto com os Testes de Usabilidade. Dessa maneira, o planejamento gerado a cada ciclo de *releases* e iterações contemplará as Estimativas de recursos para aquele *release*/iteração já incluindo os recursos para as atividades de usabilidade propostas pelo método ao longo do ciclo de vida.

4.1.2.2 Controle de Usabilidade

Ao final de cada iteração, a **Velocidade do Projeto** é avaliada e os atrasos ou adiantamentos relacionados à usabilidade que foram observados durante a iteração são incorporados e reavaliados pela atividade de **Controle de Usabilidade**. Essa atividade é realizada pelo Projetista de Interação em conjunto com o *Tracker* do XP. Ela tem como principal objetivo o acompanhamento das tarefas relativas à usabilidade, comparando o esforço previsto com o realizado, de forma a melhorar a qualidade das estimativas para as iterações seguintes. As estimativas revistas e melhoradas geram um **Refinamento de Estimativas** que re-alimenta o Planejamento de Usabilidade para o próximo *release*.

Esse processo de constante revisão das estimativas realizadas já é utilizado pelo XP no planejamento relativo às Histórias de Usuário. A proposta do método XPu é que ao final de cada iteração (quando tipicamente é entregue software funcional ao cliente), as estimativas relacionadas à usabilidade sejam revistas, baseado nos resultados das avaliações realizadas ao longo da iteração. Espera-se que essa revisão melhore a qualidade das estimativas que serão realizadas para os próximos *releases*, considerando que caso os resultados das avaliações não sejam positivos, pode ser necessário que o Projetista de Interação alocue um tempo maior para trabalhar nas Histórias de Usabilidade junto aos Usuários.

4.2 Visão de Iteração

A Visão de Iteração apresenta as atividades do método em um nível de detalhamento mais refinado. Esse nível de detalhamento permite abordar as questões específicas das iterações que compõem um *release* do método XP. Tipicamente, os *releases* em XP são formados por várias iterações de curta duração e uma iteração deve ser projetada para durar em torno de uma semana. O fluxo das atividades do XPu pertencentes à Visão de Iteração e a interação entre elas são apresentados nas Figuras 4.4 e 4.5.

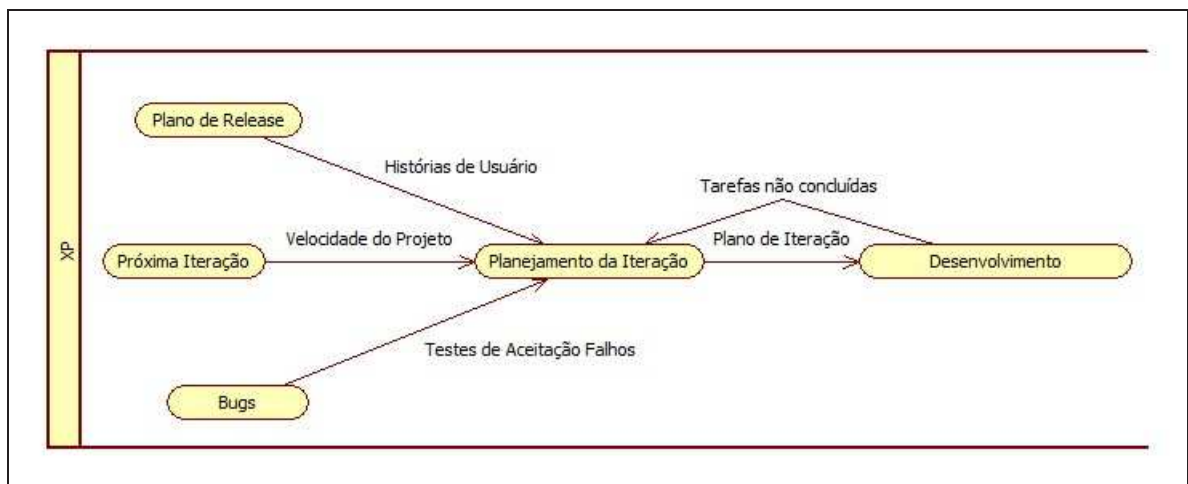


Figura 4.4. Visão de Iteração do XPu e sua integração com o XP - Parte 1.

4.2.1 Atividades Técnicas

As Atividades Técnicas relativas à Visão de Iteração do XPu estão focadas na geração de *Templates* e Interfaces Finais para a construção do produto de software. Esses modelos são gerados a partir das atividades de Definição de Estilos de Interação e Desenho da Interação. As duas atividades são descritas em detalhes nas seções a seguir.

4.2.1.1 Definição de Estilos de Interação

Ao iniciar as atividades de uma iteração, tipicamente os pares de uma equipe XP selecionam Histórias de Usuários a serem desenvolvidas naquela iteração. Após essa seleção, são escritos Testes Unitários para aquela História e em seguida é escrito o código-fonte relativo às tarefas que devem ser contempladas pela História em questão. Esse ciclo de atividades ocorre diariamente.

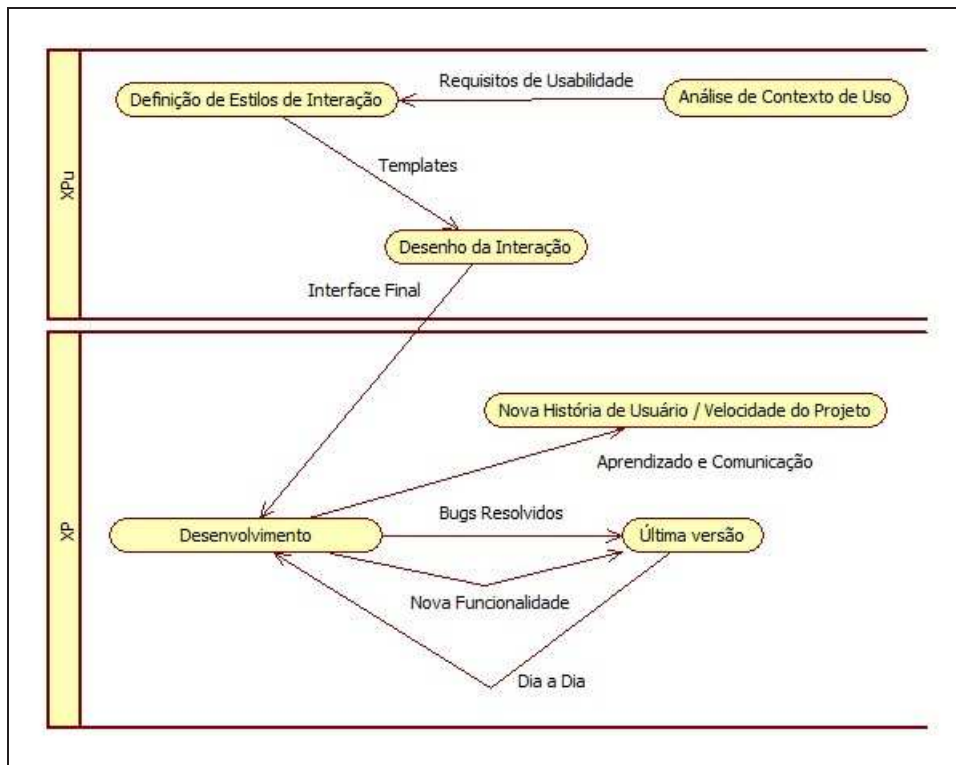


Figura 4.5. Visão de Iteração do XPU e sua integração com o XP - Parte 2.

A proposta do XPU é que sejam agregadas atividades relativas à **Definição de Estilos de Interação** a esse ciclo. A atividade de Definição de Estilos de Interação utiliza como insumo os Requisitos de Usabilidade para definir um padrão de *Templates* para as interfaces que serão desenvolvidas para as várias tarefas de uma História de Usuário. Os *Templates* são modelos estruturados de interface que permitem definir um Guia de Estilo padrão e coerente entre todos os artefatos de interação que serão utilizados pelo produto.

O Guia de Estilo tem como principal objetivo descrever diretrizes e padrões para o desenvolvimento de software, e em alguns casos, descrever também métodos para garantir a consistência entre produtos de uma família de produtos e dentro de um produto. Dentre os benefícios gerados pela adoção de Guias de Estilo, é possível destacar que eles:

- garantem consistência em uma família de produtos;
- provêm manutenção de um repositório de diretrizes e padrões;
- atuam como uma ferramenta para o treinamento de novos desenvolvedores e
- facilitam o trabalho em conjunto de grupos.

Além disso, os Guias de Estilo também geram benefícios para os usuários finais e para os próprios desenvolvedores. Segundo Hix & Hartson [1993], dentre os benefícios gerados para os usuários, é possível citar:

- a redução de erros;
- o aumento da confiança;
- o aumento da eficiência e
- a redução da resistência à nova tecnologia.

Já para os desenvolvedores, os Guias de Estilo:

- minimizam a reinvenção;
- facilitam o treinamento de novos desenvolvedores;
- reduzem tomadas de decisões arbitrárias;
- diminuem o tempo de desenvolvimento e
- beneficiam o re-uso.

No entanto, ao contrário das abordagens centradas em documentação, o Guia de Estilo gerado pelo método XPu não é um documento contendo diretrizes, e sim o próprio *Template* para as interfaces. Esse *Template* pode ser visto como o “esqueleto” padrão sobre o qual as interfaces futuras serão desenvolvidas. Naturalmente, esse “esqueleto” abriga as diretrizes e os padrões fundamentais que devem ser seguidos por qualquer interface que seja gerada para o software em questão e, por essa razão, cumprem o objetivo para o qual os Guias de Estilo são gerados.

Tipicamente, os *Templates* são criados uma única vez na iteração inicial do projeto. No entanto, para o método XPu, é possível que os *Templates* sejam refinados e melhorados ao longo das iterações, mantendo a característica iterativa e incremental do método XP. Por essa razão, eles são descritos na Visão de Iteração do método.

4.2.1.2 Desenho da Interação

Os *Templates* gerados durante a Definição de Estilos de Interação servem de insumo para o **Desenho da Interação** que resulta na definição das **Interfaces Finais** que serão utilizadas ao longo de todo o desenvolvimento do produto. Ao contrário dos

Templates, as Interfaces Finais são desenvolvidas diretamente na ferramenta ou no *framework* de desenvolvimento utilizado pelo projeto.

Diversos trabalhos na literatura [Constantine & Lockwood, 1999; Galitz, 2002; Hix & Hartson, 1993] descrevem uma série de diretrizes para a construção de interfaces gráficas, além de políticas gerais e específicas para a utilização de elementos de interação e navegação. Não é objetivo do método XPu especializar ou propor diretrizes próprias para a criação de interfaces gráficas, portanto todas as diretrizes e políticas já definidas e publicadas pela área de IHC podem ser utilizadas pela atividade de Desenho da Interação.

É importante observar, por fim, que um sistema possuirá diversas interfaces gráficas pelas quais os usuários irão interagir com a aplicação e, naturalmente, cada uma das interfaces geradas terão suas especificidades relativas ao papel que desempenham no processo de interação do usuário com o sistema. No entanto, toda e qualquer interface gerada deverá estar em conformidade e seguir o *Template* definido para o projeto. Por essa razão, é tão importante separar as atividades de Definição de Estilos de Interação e de Desenho da Interação.

4.3 Visão de Desenvolvimento

A Visão de Desenvolvimento propõe uma última instância de refinamento. Esse nível de detalhamento permite abordar as questões específicas ligadas ao desenvolvimento de uma tarefa relativa a uma iteração. O fluxo dessas atividades e a interação entre elas são apresentados nas Figuras 4.6 e 4.7.

4.3.1 Atividades Técnicas

As Atividades Técnicas relativas à Visão de Desenvolvimento do XPu estão focadas na geração de Protótipos e Testes de Usabilidade que comprovem a qualidade de uso dos Protótipos e Interfaces. Esses produtos são gerados a partir das atividades de Prototipação de Interface e Avaliação de Usabilidade. As duas atividades são descritas em detalhes nas seções a seguir.

4.3.1.1 Prototipação de Interface

As atividades do método XPu que são realizadas durante a implementação de uma tarefa se iniciam com a **Prototipação de Interface**. A prototipação é a primeira instância do desenvolvimento da interface e compõe o Desenho da Interação. O **Protótipo Simplificado**, resultado da atividade de Prototipação de Interface, é

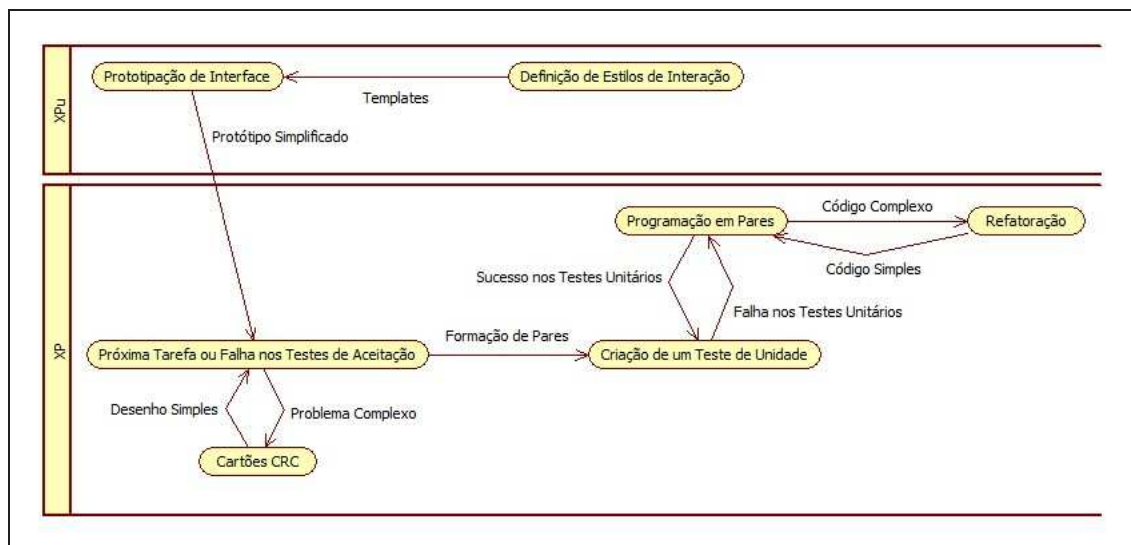


Figura 4.6. Visão de Desenvolvimento do XPu e sua integração com o XP - Parte 1.

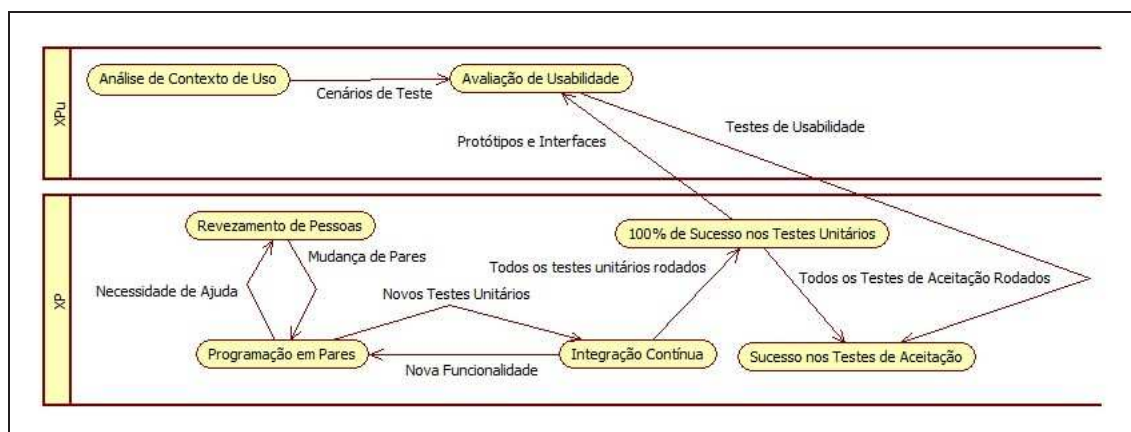


Figura 4.7. Visão de Desenvolvimento do XPu e sua integração com o XP - Parte 2.

gerado a partir dos *Templates* e é constantemente refinado de modo que o Desenho da Interação esteja concluído ao final da iteração correspondente.

Os protótipos em geral oferecem a possibilidade de observar precocemente muitos aspectos sobre a natureza final do produto, avaliando idéias e pesando alternativas antes do comprometimento com a interface final. A atividade de prototipação apresenta bastante afinidade com o modelo de desenvolvimento sustentado pelos métodos ágeis. Ambos se baseiam no desenvolvimento iterativo, com sucessivos refinamentos do produto até que se alcance um resultado final próximo do ideal. A prototipação, assim como os métodos ágeis, permite a detecção precoce de problemas, reduzindo custos e melhorando a qualidade do produto.

A prototipação tem a vantagem de promover a participação e o comprometimento do usuário, característica bastante ressaltada pelo método XP e que estimula a Colaboratividade, a Cooperação e a Orientação a Pessoas. Protótipos também permitem que os desenvolvedores observem o comportamento dos usuários e suas reações de forma semelhante ao que aconteceria com o produto final, já que simplesmente não é factível, somente com o uso de diretrizes de projetos, ter um produto adequado da primeira vez.

De uma maneira geral, as informações que podem ser extraídas de um protótipo são:

- as funcionalidades necessárias ao sistema;
- as seqüências de operação;
- as necessidades de suporte ao usuário;
- as representações necessárias;
- o *look and feel* da interface e
- a comunicabilidade e usabilidade da aplicação.

São diversas as técnicas de prototipação e suas características e benefícios são descritos nos trabalhos de Alavi [1984], Boehm et al. [1984], Ehn [1990], Hix & Hartson [1993] e Nielsen [1993]. O método XPu utiliza em especial a Prototipação Rápida com o objetivo de gerar Protótipos Simplificados. A Prototipação Rápida é uma técnica natural que começa com detalhes específicos de um projeto de interação, depois os estrutura e refina dentro do sistema. Através da Prototipação Rápida, o Projetista da Interação tem a oportunidade de avaliar desenhos propostos o mais cedo possível.

O uso da Prototipação Rápida no método XPu é feito através de protótipos de baixa fidelidade em papel, cujos benefícios de utilização são descritos em Nielsen [1990]. Desenhos manuais do protótipo da interface são feitos pelo Projetista da Interação e apresentados aos Usuários com o objetivo de avaliá-los e discutir alternativas de desenho. O uso da criatividade na construção desses protótipos ajuda bastante na redução de maiores esforços de desenho da interface [Ungar, 2008]. Através das avaliações e observações dos Usuários, o desenho vai sendo refinado até que represente a expectativa destes em relação à sua interação com o sistema. O desenho resultante dos sucessivos refinamentos gera o Protótipo Simplificado para aquela iteração e este é desenhado na Área de Trabalho Informativa utilizada pelo método XP. Com a definição

dos Protótipos Simplificados e dos *Templates*, é possível partir para o Desenho da Interação e gerar as Interfaces Finais necessárias àquela iteração.

4.3.1.2 Avaliação de Usabilidade

Sabe-se que a avaliação de interfaces é um importante passo do processo de *design*, pois permite apreciar se o sistema apóia adequadamente os usuários, nas suas tarefas e no ambiente em que será utilizado. A utilização de métodos e princípios de projeto de interfaces não é suficiente para garantir uma alta qualidade de uso do software. A avaliação permite estimar o sucesso ou insucesso das hipóteses do *designer* sobre a solução que ele está propondo, tanto em termos de funcionalidade, quanto de interação.

Segundo Preece et al. [1994], as avaliações de interface são classificadas quanto:

- ao método de coleta:
 - formativa e
 - somativa.
- aos tipos de dados:
 - qualitativos e
 - quantitativos.
- e ao método de análise:
 - preditiva;
 - interpretativa e
 - experimental.

Com relação aos métodos de coleta, a avaliação formativa ocorre quando o sistema está em estágio inicial de avaliação e os primeiros resultados estão sendo colhidos. Já a avaliação somativa ocorre quando esta contribui com novos resultados e permite a comparação com outras avaliações anteriormente realizadas. Para essas coletas, podem ser utilizadas a opinião dos usuários, a observação e monitoração dos usuários, os registros de uso e a coleta da opinião de especialistas.

Com relação aos tipos de dados, a avaliação será qualitativa quando utilizar resultados subjetivos como a opinião de especialistas ou questionários estruturados respondidos por usuários típicos. Quando os resultados podem ser medidos numericamente como na utilização de níveis para as tarefas de *Benchmark*, a avaliação será quantitativa.

Por fim, com relação aos métodos de análise, a avaliação preditiva normalmente ocorre quando são feitas inspeções da interface por especialistas. A avaliação preditiva identifica potenciais problemas que os usuários podem experimentar. Essas avaliações apresentam a vantagem de serem mais baratas do que as avaliações com usuários, mas apresentam a desvantagem de justamente não avaliarem a experiência do usuário com a aplicação.

A avaliação experimental é realizada em ambiente controlado e envolve usuários na realização das tarefas e especialistas na preparação e análise dos dados. Essas avaliações apresentam a vantagem de serem baseadas em experiências reais dos usuários com a aplicação, mas apresentam as desvantagens de serem realizadas em um ambiente não natural para o usuário, além de serem mais caras que as avaliações preditivas.

Já a avaliação interpretativa envolve a observação do uso feito pelo usuário em seu contexto natural. Essas avaliações apresentam a vantagem de que especialistas podem levar em consideração aspectos reais que influenciam no uso do sistema, mas apresentam as desvantagens de ser difícil comparar dados coletados, além do seu alto custo e dificuldade de implementação.

Ao longo do ciclo de vida do XPu, diversas avaliações podem ser utilizadas. As mais frequentes são feitas através dos Testes de Usabilidade. Idealmente, quando ainda estamos trabalhando com **Protótipos Simplificados**, inclusive protótipos de baixa fidelidade em papel, o método de avaliação indicado é a avaliação preditiva, feita pelo Projetista de Interação com características formativas e qualitativas. À medida que se tenha **Interfaces** mais próximas da definição final, podemos utilizar avaliações mais abrangentes com usuários finais através das avaliações experimentais, com características formativas e qualitativas ou quantitativas (nesse caso, utilizando os *Benchmarks* definidos na atividade de Definição de Requisitos e Metas de Usabilidade). Ao final do ciclo de iteração, quando tipicamente se tem um sistema com Interfaces Funcionais e este é avaliado para entrar em produção, pode-se utilizar avaliações experimentais com características somativas ou avaliações interpretativas com características formativas. Em ambos os casos, as abordagens podem ser quantitativas ou qualitativas. Ao final das avaliações, o sucesso nos **Testes de Usabilidade** é uma das condições necessárias para o sucesso dos Testes de Aceitação, que no método XP são estabelecidos pelo próprio cliente. Seja qual for o instrumento e o estágio de avaliação, os **Cenários de Teste** são utilizados como insumo para a atividade de **Avaliação de Usabilidade**.

Para cada um dos métodos de avaliação existem várias propostas fundamentadas a partir de várias teorias explicativas de IHC. O método XPu não tem o objetivo de especificar quais métodos de avaliação devem ser utilizados ao longo do ciclo de

vida, inclusive abordagens que avaliam outros aspectos da qualidade de uso como a comunicabilidade [de Souza, 2005] também podem ser utilizadas em conjunto com os Testes de Usabilidade. Portanto, para as avaliações preditivas podem ser utilizados, por exemplo, os métodos de Avaliação Heurística [Nielsen, 1994], o Método de Inspeção Semiótica [Prates & Barbosa, 2007] ou o Percorso Cognitivo [Wharton et al., 1994]. Para as avaliações experimentais, podem ser utilizados, por exemplo, os Testes de Usabilidade com Usuários [Rubin, 1994] ou o Método de Avaliação de Comunicabilidade [Prates & Barbosa, 2007]. Finalmente, para as avaliações interpretativas, pode-se utilizar, por exemplo, métodos como o *Think Aloud* [Ericsson & Simon, 1993].

Avaliações complementares também podem ser utilizadas para enriquecer os resultados obtidos com os Testes de Usabilidade. Um método que pode ser utilizado com esse objetivo é o QUIS (*Questionnaire for User Interaction Satisfaction*) [Harper et al., 1997]. O QUIS foi desenvolvido por pesquisadores da Universidade de Maryland e é um exemplo conhecido de questionário para avaliação da satisfação de usuários. O QUIS contém um questionário demográfico, uma parte dedicada à medida de satisfação de modo geral e uma parte dedicada à medida de onze fatores específicos de uma forma organizada hierarquicamente. Os fatores considerados são aspectos de tela, terminologia e *feedback* do sistema, aprendizado, características do sistema, documentação técnica, tutoriais on-line, multimídia, reconhecimento de voz, ambiente virtual, acesso à internet e instalação do software.

Capítulo 5

Avaliação

Este capítulo apresenta os resultados alcançados nas avaliações do método XPu. A primeira seção descreve a inspeção do método em relação às suas características ágeis com o objetivo de avaliar o impacto da inserção de atividades de usabilidade na agilidade do método resultante, segundo abordagens da literatura da área. Já a segunda seção descreve o estudo de caso realizado para avaliar a instanciação das atividades do método em um projeto de desenvolvimento de software realizado com fins experimentais.

5.1 Caracterização de Agilidade

O método XPu foi inicialmente avaliado pelo autor deste trabalho através de uma inspeção de suas características ágeis. Essa inspeção utilizou como referência a caracterização dos métodos ágeis proposta por Abrantes & Travassos [2007b]. Nesse trabalho, os autores, através de uma *quasi-revisão* sistemática de literatura, avaliaram cerca de 1000 artigos acadêmicos publicados nos principais veículos da área com o objetivo de identificar características e práticas comuns aos métodos ágeis. A partir desse conjunto de características, os autores propuseram a seguinte caracterização:

Um método para ser caracterizado ágil, deve apresentar, em um grau adequado ao contexto de desenvolvimento de software em que se insere, as características de adaptabilidade, incrementalidade, iteratividade, colaboratividade, cooperação, orientação a pessoas, parcimônia (leanness) e restrição de prazo.

A partir dessa caracterização, portanto, podemos inspecionar o método XPu e avaliá-lo com o objetivo de constatar (ou não), se a sua integração com o método XP

mantém suas características originais de agilidade. A avaliação apresentada a seguir descreve cada uma das características e sua análise em relação à integração XP/XPu. A análise proposta utiliza uma escala em três níveis: o XPu **mantém na sua totalidade**, **mantém parcialmente** ou **não mantém** as características de agilidade do XP.

Adaptabilidade: *Habilidade e capacidade de adaptação rápida do processo para atender e reagir a mudanças de última hora nos requisitos e/ou no ambiente, ou a situações ou riscos não previstos inicialmente.*

O XPu **mantém parcialmente** essa característica do XP. O método XPu permite que o XP mantenha-se adaptável em relação às constantes mudanças de requisitos, no entanto é possível que haja algum impacto na redefinição de *Templates* e Padrões de Interface e Navegação, por exemplo. O método permanece iterativo e incremental e as atividades de usabilidade propostas também fazem parte desse ciclo. Logo, ainda é possível atender e reagir a mudanças de última hora proporcionadas pelo ambiente ou pelos próprios requisitos, porém com algumas limitações que podem gerar algum retrabalho.

No entanto, é importante ressaltar que se houver necessidade de mudanças no estilo da interface, elas terão que ser feitas seja no XP ou no XPu. O fato de se usar *Templates* como propõe o XPu pode facilitar esta mudança, não necessariamente dificultar, já que facilita a comunicação das mudanças com toda a equipe. É importante também acrescentar que os *Templates* no XPu são feitos de maneira leve (“ágil”), na forma de definições e exemplos comentados disponíveis a toda a equipe através de uma intranet ou espalhados na Área de Trabalho Informativa, por exemplo.

Incrementalidade: *Não tentar construir o sistema todo de uma só vez; o sistema é partido em incrementos (pequenos releases com novas funcionalidades) que podem ser desenvolvidos em paralelo em ciclos rápidos; quando o incremento é completado e testado, ele é integrado ao sistema.*

O XPu **mantém parcialmente** essa característica do XP. O método XPu continua seguindo o ciclo incremental, aonde as Histórias e os Testes de Usabilidade são definidos e revistos separadamente para cada ciclo de entrega, no entanto é possível que haja algum impacto na definição prévia de *Templates* e Padrões de Interface e Navegação, por exemplo, já que uma prospecção inicial desses artefatos é necessária para iniciar o trabalho de desenvolvimento. Logo, ainda é possível partir o sistema em pequenos *releases*, além de mapear e testar apenas as necessidades para determinado instante, porém com algumas limitações que fazem com que alguma parcela dos padrões seja desenvolvida inicialmente.

Iteratividade: *Envolve vários ciclos curtos, dirigidos por características do produto, nos quais um certo conjunto de atividades é completado em poucas semanas; estes ciclos são repetidos muitas vezes para refinar as entregas.*

O XPu **mantém na sua totalidade** essa característica do XP. A iteratividade está bastante relacionada à incrementalidade. Os ciclos de vida nos métodos de desenvolvimento ágeis são baseados em espirais, que possuem características tanto iterativas, quanto incrementais. Sendo assim, o XPu mantém o desenvolvimento em ciclos curtos, que são repetidos diversas vezes a cada *release*/iteração. Logo, as entregas de software funcional e usável (do ponto de vista da usabilidade) são liberadas em versões cada vez mais refinadas, sempre adicionando novos recursos ao produto final.

Colaboratividade: *É uma atitude entre membros da equipe de desenvolvimento, entre os quais se encoraja a comunicação para disseminar informação e apoiar integração rápida de incrementos.*

O XPu **mantém na sua totalidade** essa característica do XP. Valores como Comunicação e práticas como Sentar Junto colaboram para que a Colaboratividade seja acentuada. Todo código desenvolvido pelo método continua sendo em pares e o papel do Projetista de Interação é o de um membro que também participa da equipe de desenvolvimento, recebendo e contribuindo com a colaboração entre os participantes do time. Logo, o XPu encoraja a Comunicação e a disseminação de informação, contribuindo para a integração rápida dos produtos envolvidos, tanto relacionados ao código como à usabilidade.

Cooperação: *Interação aberta e com proximidade entre os vários stakeholders (especialmente entre clientes e desenvolvedores); o cliente deve tomar parte ativa no processo de desenvolvimento e prover feedback de forma regular e freqüente.*

O XPu **mantém na sua totalidade** essa característica do XP. Uma das principais características dos métodos de usabilidade é estimular a troca de informação e a proximidade entre os *stakeholders*, principalmente aqueles relacionados aos usuários finais da aplicação. A participação do Cliente é ativa tanto nos métodos ágeis, quanto nos processos de usabilidade; característica essa que os aproxima e torna possível integrações como a proposta por este trabalho. Logo, o XPu mantém o relacionamento cliente/desenvolvedores de maneira adequada e provendo *feedback* constante em relação à usabilidade do produto.

Orientação a pessoas: *Favorecimento de pessoas sobre processos e tecnologias; desenvolvedores são encorajados a aumentar sua produtividade, qualidade e*

desempenho; a comunicação e a cooperação dentro das equipes de desenvolvimento são consideradas fundamentais e necessárias. As reuniões diárias em pé e os workshops de reflexão dão às pessoas a chance de manifestar suas preocupações.

O XPu **mantém na sua totalidade** essa característica do XP. Outra principal característica dos métodos de usabilidade é focar o desenvolvimento do produto centrado no usuário (UCD). Parte-se do princípio de que o produto não basta atender apenas aos requisitos tecnológicos, mas que tão ou mais importante do que isso, é atender às necessidades e anseios dos usuários finais da aplicação. Essa é mais uma característica comum a métodos ágeis e processos de usabilidade. Logo, o XPu, mantendo todas as práticas do XP que já possuem esse objetivo, contribui para que a orientação a pessoas seja eficiente e constante ao longo do desenvolvimento.

Parcimônia (*leanness*): *Eliminação de perdas ou habilidade de fazer mais com menos recursos; característica que o processo ágil tem, de requerer o mínimo necessário de atividades para mitigar riscos e alcançar metas; deve-se remover todas as atividades desnecessárias no processo de desenvolvimento.*

O XPu **mantém na sua totalidade** essa característica do XP. Algumas atividades do Praxis-u foram retiradas do método e outras foram remanejadas em sua adaptação para o contexto de um método ágil como o XP. Essas adaptações tiveram como objetivo justamente manter essa característica dos métodos ágeis. Logo, o XPu requer o mínimo necessário e suficiente de atividades de usabilidade para avaliar e diminuir riscos ao longo do método, contribuindo para prover produtos de alta usabilidade com menos recursos despendidos para tal.

Restrição de prazo: *Estabelecimento de limite de tempo para cada iteração programada. Grandes volumes de desenvolvimento são quebrados em múltiplas entregas que possam ser desenvolvidas incremental e concorrentemente de modo previsível.*

O XPu **mantém na sua totalidade** essa característica do XP. Como já foi avaliado, características como iteratividade e incrementalidade estão presentes e são mantidas pelo XPu. As iterações permanecem restritas ao prazo estipulado para serem entregues e suas tarefas podem ser executadas de maneira concorrente, já que o Projetista de Interação é membro permanente da equipe de desenvolvimento e realiza suas atividades paralelamente ao andamento das atividades de desenvolvimento. Logo, o XPu mantém as restrições de prazo, características dos métodos ágeis.

Após a análise dessa caracterização, é possível concluir que o XPu mantém em sua totalidade grande parte das características ágeis definidas para métodos dessa natureza.

Com isso, é possível avaliá-lo em diversos outros contextos, certos de manter e respeitar a filosofia de desenvolvimento ágil publicada pelo Manifesto Ágil e utilizada por vários métodos que a seguem.

5.2 Estudo de Caso

Segundo Mafra & Travassos [2006], estudos de caso são estudos conduzidos com o propósito de se investigar uma entidade ou um fenômeno dentro de um espaço de tempo específico. Estudos de caso são usados principalmente para a monitoração de atributos presentes em projetos, atividades ou atribuições. Durante a condução de um estudo de caso, dados são coletados e, baseado na coleta desses dados, análises estatísticas são conduzidas de forma a permitir-se avaliar um determinado atributo ou o relacionamento entre diferentes atributos. Uma vantagem na condução de estudos de caso é a facilidade de planejamento. Entretanto, uma desvantagem é a dificuldade em generalizar-se os resultados obtidos. Geralmente, os resultados de um estudo de caso apontam os efeitos em uma situação em particular, não podendo ser generalizados para cada situação presente em outros contextos. Estudos de caso são discutidos em Robson [1993], Stake [1995], Pfleeger [1994], Yin [1994] e Wöhlin et al. [2000].

Diversas são as abordagens apresentadas na literatura para a condução de experimentos como os estudos de caso [Basili et al., 1994; Solingen & Berghout, 1999; Wöhlin et al., 2000]. De uma maneira geral, todas elas dividem o processo de experimentação em 4 fases básicas: definição, planejamento, execução e apresentação. Segundo Travassos et al. [2002], a definição é a primeira fase onde o experimento é expresso em termos dos problemas e objetivos. A fase de planejamento vem em seguida onde o projeto do experimento é determinado, a instrumentação é considerada e os aspectos da validade do experimento são avaliados. A execução do experimento segue o planejamento. Nesse momento os dados experimentais são coletados para serem analisados e avaliados na fase de análise e interpretação. Finalmente, os resultados são apresentados e empacotados durante a fase de apresentação.

Esta seção, portanto, tem o objetivo de descrever sistematicamente o estudo de caso realizado a partir da abordagem metodológica apresentada por Travassos et al. [2002]. As subseções seguintes apresentam em detalhes cada uma das 4 fases realizadas durante o processo de experimentação, apresentando os objetos de estudo, o escopo e o objetivo, o perfil do sistema e dos participantes, as métricas e comparações realizadas, os resultados obtidos e a discussão e análise desses resultados.

5.2.1 Definição dos Objetivos

A fase de definição descreve os objetivos, o objeto do estudo, o foco da qualidade, o ponto de vista e o contexto. Como resultado, a fase de definição fornece a direção geral do experimento, o seu escopo, a base para a formulação das hipóteses e as notações preliminares para a avaliação da validade. Com esse propósito, as seções seguintes descrevem os objetivos global, da medição e do estudo para o estudo de caso realizado, além da formalização das estratégias de medição definidas para avaliar cada objetivo de medição.

5.2.1.1 Objetivo Global

O objetivo global do estudo é descrever uma situação de uso na qual o método XPu foi avaliado, apontando benefícios, conformidades, limitações e dificuldades na utilização prática do método em um contexto específico.

5.2.1.2 Objetivo da Medição

Tendo em vista a utilização prática do método XPu em um projeto de desenvolvimento ágil, avaliar:

1. a viabilidade de utilização conjunta das práticas do Praxis-u e do método XP;
2. a viabilidade de particionamento das atividades do Praxis-u para utilização em um contexto iterativo e incremental;
3. a qualidade dos protótipos e interfaces de software produzidas utilizando-se o XP em comparação com a utilização do método proposto;
4. a sobrecarga de trabalho propiciada pela adição das atividades de usabilidade ao método XP;
5. a manutenção das características de agilidade do método XP após sua integração com o XPu.

5.2.1.3 Objetivo do Estudo

Analisar a aplicação prática do método XPu

Com o propósito de avaliar

Com respeito à integração de suas atividades com as atividades do método XP

Do ponto de vista de uma equipe de desenvolvimento ágil

No contexto de profissionais recém-graduados na área de computação.

5.2.1.4 Estratégias de Medição

OM1: Avaliar a viabilidade de utilização conjunta das práticas do Praxis-u e do método XP.

EM: Observar a integração das atividades de usabilidade ao longo do ciclo de vida ágil durante o desenvolvimento do produto, registrar possíveis contenções e realizar uma entrevista pós-teste para discutir com os participantes as principais dificuldades encontradas na instanciação das atividades do Praxis-u ao método XP.

OM2: Avaliar a viabilidade de particionamento das atividades do Praxis-u para utilização em um contexto iterativo e incremental.

EM: Verificar a qualidade da interface produzida após o desenvolvimento do produto utilizando o método XPu, que por sua vez utiliza as atividades particionadas do Praxis-u.

OM3: Avaliar a qualidade dos protótipos e interfaces de software produzidas utilizando-se o XP em comparação com a utilização do método proposto.

EM: Aplicar uma avaliação heurística nas interfaces geradas pelo método XP e pelo método XPu e comparar o número de problemas encontrados em cada uma delas.

OM4: Avaliar a sobrecarga de trabalho propiciada pela adição das atividades de usabilidade ao método XP.

EM: Registrar separadamente o tempo de desenvolvimento gasto em cada uma das histórias de usuário utilizando os métodos XP e XPu.

OM5: Avaliar a manutenção das características de agilidade do método XP após sua integração com o XPu.

EM: Observar a permanência das características ágeis definidas em Abrantes & Travassos [2007b] ao longo do desenvolvimento do produto utilizando o método XPu.

5.2.2 Planejamento

A fase de planejamento implementa a fundação do estudo de caso. Nessa seção são apresentados a seleção do contexto, a formulação das hipóteses, a seleção das variáveis, a seleção dos participantes, o projeto do estudo de caso, a preparação conceitual da instrumentação e a consideração sobre validade do estudo realizado. O resultado dessa fase apresenta o estudo de caso totalmente elaborado e pronto para execução.

5.2.2.1 Definição das Hipóteses

Hipótese Nula (H0): A integração proposta pelo XPu acrescenta qualidade à interface final do produto, sem que para isso seja necessária a burocratização do processo, isto é, sem que o método XP seja descaracterizado em sua “agilidade”.

Hipótese Alternativa (H1): A integração proposta pelo XPu acrescenta qualidade à interface final do produto, no entanto a sobrecarga de trabalho imposta à equipe XP sugere dificuldades de utilização do método na prática.

Hipótese Alternativa (H2): As atividades do método XPu não alteram significativamente o tempo de desenvolvimento do produto, no entanto o ganho de qualidade na interface final é considerado pequeno ou insuficiente para justificar a adoção do método na prática.

5.2.2.2 Seleção do Contexto

O contexto selecionado para o estudo de caso em questão foi o desenvolvimento de um software interativo de busca de mídias em locadoras de DVDs. O objetivo do sistema era o de auxiliar usuários em terminais de auto-atendimento na busca por vídeos e na locação automática da mídia escolhida, sem a necessidade da intervenção de um atendente. O autor deste trabalho atuou no papel de Cliente durante o projeto que foi rigorosamente desenvolvido segundo o ambiente e as práticas ágeis propostos pelo método XP. Isto é, toda a configuração do ambiente de desenvolvimento ágil foi utilizada, bem como todas as práticas ágeis pertinentes ao escopo e ao contexto do projeto.

Para a realização do estudo, foi disponibilizado um laboratório de informática de uma universidade pública configurado de acordo com o ambiente ágil característico do método XP. A equipe optou pela utilização do NetBeans como plataforma de desenvolvimento em linguagem Java, utilizando o *framework* JUnit como suporte para criação de testes automatizados. Por fim, o SGBD de escolha foi o MySQL.

5.2.2.3 Seleção dos Indivíduos

A equipe de desenvolvimento foi formada por dois desenvolvedores recém-graduados em Sistemas de Informação em uma universidade pública no interior de Minas Gerais. Um dos desenvolvedores possuía vasta experiência em desenvolvimento de software e o outro era bem menos experiente, limitando-se à habilidade adquirida durante o curso de graduação e seus estágios realizados. No entanto, nenhum dos desenvolvedores

tinha experiência prévia com desenvolvimento ágil e atividades de IHC, limitando-se à utilização de práticas isoladas de Engenharia de Software e Modelagem Orientada a Objetos adquiridas ao longo de quatro disciplinas cursadas em nível de graduação. A equipe esteve disponível para a realização do estudo durante uma semana ininterrupta em período integral e com dedicação exclusiva.

5.2.3 Execução

O estudo de caso descrito por este capítulo foi conduzido durante uma semana no final do mês de Janeiro de 2009, com a equipe e o contexto descritos na seção anterior. A descrição do estudo está apresentada nas subseções abaixo, destacando as etapas de treinamento, especificação do produto e atividades realizadas durante o desenvolvimento utilizando os métodos XP e XPu.

5.2.3.1 Treinamento

O treinamento que antecedeu o início do estudo foi dividido em duas etapas. Como o sistema proposto seria desenvolvido em duas partes (uma utilizando o método XP e outra utilizando o método XPu), foi necessário treinar a equipe da mesma maneira. Inicialmente, portanto, o autor deste trabalho treinou a equipe apenas em aspectos gerais da Engenharia de Software, com foco no surgimento e na adoção dos métodos ágeis. Foram abordados os seguintes temas:

- **O Manifesto Ágil:** surgimento, valores e princípios.
- **O método XP:** surgimento, principais valores, princípios e práticas.
- **Planejamento em métodos ágeis:** técnicas de estimativas e melhores práticas em XP.

Somente após a finalização dos trabalhos da primeira parte do desenvolvimento, o autor deste trabalho treinou a equipe visando à segunda parte do trabalho. Nesse etapa, foram abordados os seguintes temas:

- **Processos de Usabilidade:** introdução, principais normas e modelos.
- **O método XPu:** proposta, visão de projeto, iteração e desenvolvimento.
- **Técnicas de IHC:** Roteiros, Personas, Avaliação Heurística, *Templates* e Prototipação Rápida.

A divisão do treinamento em duas etapas teve como principal motivação o fato de que o conhecimento prévio do método XPu e das atividades e técnicas de usabilidade poderia influenciar na realização das tarefas de desenvolvimento ligadas ao método XP, ou seja, princípios e práticas poderiam ser realizados prematuramente durante a primeira etapa do estudo e dessa forma, influenciar no resultado final.

5.2.3.2 Especificação do Produto

Conforme apresentado na seção deste capítulo que descreve a Seleção do Contexto, o sistema proposto para o estudo de caso foi um sistema de busca interativa de mídias para locadoras de DVDs. Dado esse contexto, o Cliente propôs o desenvolvimento de 2 Histórias de Usuário para o projeto. A primeira delas descrevia que o usuário da locadora deveria ser capaz de realizar uma busca de filmes através de um terminal interativo na locadora. A segunda descrevia que o usuário deveria ser capaz de locar a mídia escolhida através do terminal e receber um comprovante de sua locação contendo os filmes locados, as datas de entrega e o valor a pagar por cada um deles.

Como o escopo do produto foi reduzido de tal forma que o sistema pudesse ser desenvolvido no tempo disponível, as histórias contemplaram apenas requisitos essenciais para o funcionamento da aplicação. Funcionalidades suplementares como cadastro e manutenção das mídias da locadora, bem como a manutenção dos dados cadastrais dos usuários não foram mapeados como requisitos nas histórias propostas para o desenvolvimento.

5.2.3.3 Desenvolvimento utilizando o método XP

Na primeira etapa do desenvolvimento do estudo, os dois membros da equipe atuaram durante todo o projeto formando um Par de Programação (*Pair Programming*). Como foram propostas apenas 2 Histórias de Usuário, estas naturalmente foram organizadas em um mesmo *release* e em uma mesma iteração. Logo, a atividade inicial de planejamento utilizada foi o Planejamento da Iteração, visto que não seria necessário um Planejamento de *Release*, dado o escopo do projeto. Portanto, durante o Planejamento da Iteração e após receberem as histórias a serem desenvolvidas, o par atuou no mapeamento e na estimativa das tarefas para aquela iteração, utilizando a técnica de *Planning Poker*. Inicialmente, a dupla estabeleceu uma escala relativa de 1 a 5 para classificar a complexidade das histórias, sendo a complexidade 1 a mais baixa e a complexidade 5 a mais alta. A primeira história foi classificada com complexidade 5 e a segunda com complexidade 1. Após essa classificação, o tempo estimado para o desenvolvimento de cada uma das tarefas foi registrado através de consenso nas

estimativas fornecidas por cada um dos dois integrantes da dupla separadamente. Esses dados estão apresentados nas Figuras 5.1 e 5.2.

História 1 = ●●●●●			
Descrição: O cliente pode realizar uma busca de filmes através de um terminal na locadora.	Tarefa	Tempo Est.	Tempo Gasto
	Criar Banco de Dados	0:30	0:30
	Criar Classe de Conexão com o Banco de Dados	3:00	2:00
	Desenvolver as funções referentes a Cadastrar Cliente	2:00	1:40
	Criar a Interface de Busca (Login)	1:30	1:50

Figura 5.1. História de Usuário 1 - XP.

História 2 = ●			
Descrição: O cliente deve receber um comprovante de sua locação contendo os filmes locados, as datas de entrega e o valor a pagar por cada um deles.	Tarefa	Tempo Est.	Tempo Gasto
	Gerar comprovante com as informações da Locação		2:00

Figura 5.2. História de Usuário 2 - XP.

Foi percebido durante as atividades de estimativa das tarefas que a dupla preferiu utilizar em conjunto as duas abordagens sugeridas para o planejamento de histórias e tarefas em XP: a abordagem de tamanho relativo (baseada na complexidade estimada das histórias) e a abordagem de tempo (baseada no tempo estimado para o desenvolvimento de cada uma das tarefas). A literatura não descreve resultados efetivos sobre a adoção conjunta das duas abordagens nas estimativas em métodos ágeis, no entanto, já que era a primeira vez que a equipe trabalhava com métodos ágeis, esse trabalho de planejamento baseado nas duas abordagens deu mais segurança ao time na estimativa das histórias.

Após o planejamento das histórias e tarefas, a equipe iniciou o desenvolvimento do produto pelos testes unitários. Conforme abordagem adotada pelo método XP [Beck, 1999; Beck & Andres, 2004], os testes de funcionalidades isoladas do sistema seguem a filosofia do Desenvolvimento Dirigido por Testes (*Test Driven Development* - TDD) [Beck, 2002] e por essa razão, os testes unitários são escritos antes do código-fonte. Seguindo essa abordagem, portanto, os testes foram escritos para cada uma das tarefas da iteração e logo em seguida o desenvolvimento do código-fonte foi iniciado pela equipe. Dado o pequeno escopo do projeto, a dupla modelou o sistema através de

rascunhos em papel e utilizou a comunicação direta entre eles, dado que não se fazia necessário um Ambiente Informativo melhor elaborado para disseminar a informação através de um número maior de colaboradores. Ao longo do desenvolvimento, métricas relativas ao tempo realmente gasto em cada tarefa foram sendo registradas por um dos membros da dupla (no papel de *Tracker*). Os tempos coletados estão registrados nas Figuras 5.1 e 5.2 presentes na subseção anterior.

Seguindo a abordagem do método XP de que a equipe deve desenvolver somente as funcionalidades realmente necessárias ao produto em um determinado momento, o sistema proposto foi desenvolvido em torno do requisito central que era a busca de mídias na locadora, representado pela Figura 5.3. Dessa forma, as funcionalidades necessárias para que esse processo se tornasse efetivo foram desenvolvidas à medida em que a demanda surgia. Para complementar o requisito de busca de mídias, foram desenvolvidas as seguintes funcionalidades:

- Login do usuário para locação de mídias (Figura 5.4).
- Cadastro do usuário, caso seu login não seja válido (Figura 5.5).
- Emissão de comprovante de locação (Figura 5.6).



Figura 5.3. Tela de Busca gerada utilizando o método XP.

Ao final do desenvolvimento, o tempo gasto em cada atividade foi contabilizado e o sistema resultante da iteração foi entregue ao Cliente para ser validado através dos Testes de Aceitação. Novamente, pelo reduzido escopo do projeto, os Testes de

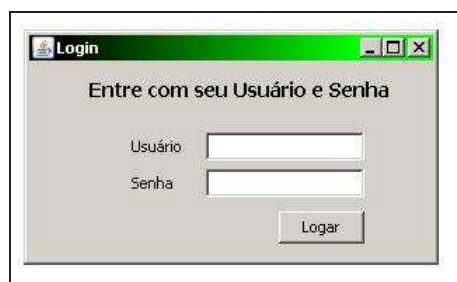
A screenshot of a web browser window titled "Login". The window has a green title bar with standard minimize, maximize, and close buttons. The main content area has a light gray background and the heading "Entre com seu Usuário e Senha". Below the heading, there are two input fields: "Usuário" and "Senha". To the right of the "Senha" field is a small eye icon. Below the input fields is a "Logar" button.

Figura 5.4. Tela de Login gerada utilizando o método XP.

A screenshot of a web browser window titled "Cadastro de Clientes". The window has a green title bar with standard minimize, maximize, and close buttons. The main content area has a light gray background and the heading "Cadastro de Clientes". Below the heading, there are several input fields: "Nome:", "CPF:", "RG:", "Endereço:", "Bairro:", "CEP:", "Cidade:", "Estado:", "Login:", "Senha:", "Telefone:", "Celular:", and "e-mail:". At the bottom of the form, there are five buttons: "Inserir", "Alterar", "Excluir", "Pesquisar", and "Fechar".

Figura 5.5. Tela de Cadastro de Cliente gerada utilizando o método XP.

Aceitação foram manuais e validados pelo Cliente através de uma simples inspeção e verificação das funcionalidades solicitadas e presentes na aplicação.

5.2.3.4 Desenvolvimento utilizando o método XPu

Após a entrega da primeira etapa do estudo, a dupla iniciou a segunda etapa que era o desenvolvimento do mesmo produto utilizando o método XPu, proposto por este trabalho. Naturalmente, era de se esperar o reuso de código na nova aplicação, visto que o sistema já havia sido desenvolvido anteriormente. Esse fator apresenta como ponto negativo e limitador do estudo a característica de considerar de maneira significativa a experiência prévia dos desenvolvedores com a aplicação, característica essa que limita a criatividade e a adoção pura do método XPu em um novo projeto. No entanto, há um ponto positivo importante na adoção dessa abordagem. Pela utilização efetiva do reuso de código e da experiência prévia com o sistema, foi possível analisar claramente a sobrecarga de trabalho imposta pelo método XPu, uma vez que todo

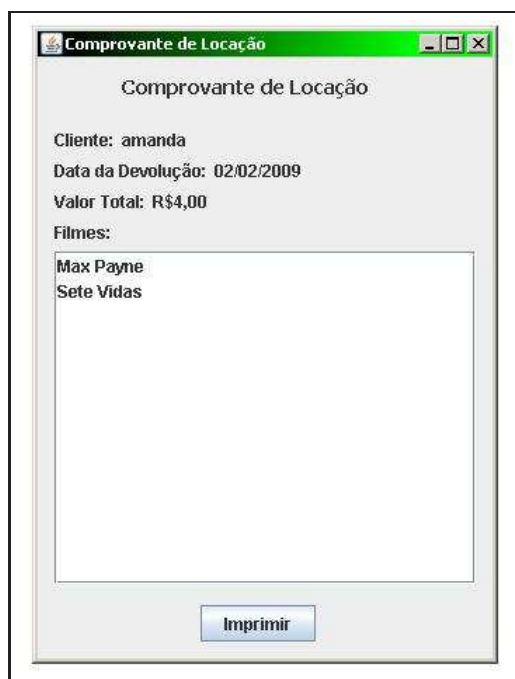


Figura 5.6. Tela de Comprovante de Locação gerada utilizando o método XP.

o tempo de desenvolvimento gasto nessa segunda etapa seria creditado à adição das atividades de usabilidade do método e, posteriormente, à reescrita de código imposta pelas modificações relacionadas à usabilidade do produto.

Nesta segunda etapa, inicialmente a dupla recebeu treinamento em processos e métodos de usabilidade (com foco no método XPu) e em técnicas de IHC, como já descrito anteriormente. Após o treinamento, a equipe iniciou o desenvolvimento pela re-estimativa de tempo das tarefas da iteração (Planejamento de Usabilidade). Para realizar essa re-estimativa, a equipe utilizou novamente a técnica de *Planning Poker* dessa vez considerando tanto nas estimativas de complexidade quanto de tempo, o esforço necessário para a realização das atividades de usabilidade do método XPu. No entanto, devido às características do estudo de caso, a equipe estimou apenas o tempo que previa gastar com as atividades de usabilidade para cada tarefa, desconsiderando o tempo que gastariam com o próprio desenvolvimento, já que fariam reuso de código com as funcionalidades já implementadas na primeira etapa do estudo. As Histórias de Usuário re-estimadas estão apresentadas nas Figuras 5.7 e 5.8. Devido ao reduzido escopo do projeto, a atividade de Definição de Requisitos e Metas de Usabilidade não foi realizada pela equipe.

Em seguida, a equipe iniciou as atividades técnicas do método pela Análise de Contexto de Uso. Para essa atividade, foram construídas as Personas e os Roteiros (Análise de Usuários e Tarefas) para formar as Histórias de Usabilidade. As Histórias

História 1 = ●●●●●			
Descrição: O cliente pode realizar uma busca de filmes através de um terminal na locadora.	Tarefa	Tempo Est.	Tempo Gasto
	Criar Banco de Dados	0:30	0:30
	Criar Classe de Conexão com o Banco de Dados	0:00	0:00
	Desenvolver as funções referentes à Cadastrar Cliente	1:00	1:00
	Criar a Interface de Busca (Login)	1:00	2:20

Figura 5.7. História de Usuário 1 - XPu.

História 2 = ●			
Descrição: O cliente deve receber um comprovante de sua locação contendo os filmes locados, as datas de entrega e o valor a pagar por cada um deles.	Tarefa	Tempo Est.	Tempo Gasto
	Gerar comprovante com as informações da Locação	1:00	2:00

Figura 5.8. História de Usuário 2 - XPu.

construídas mapearam três perfis de usuários típicos (Personas) e suas respectivas situações de uso do sistema (Roteiros). As representações descritas em ambos os casos estão apresentadas abaixo.

Maria das Dores é uma senhora de meia idade, dona de casa, com filhos, gosta de entretenimentos como novelas e filmes. De classe média alta, frequenta academia 3 vezes por semana, tem hobbies como: jardinagem, bordado e recentemente entrar na internet para conversar com as amigas. Consegue interagir de forma razoável com ambientes computacionais, como terminais eletrônicos de caixa de banco, já que o utiliza para saques e extratos, além de acessar internet para utilizar MSN e páginas como orkut.

Persona 1

*Maria das Dores iria passar o final de semana sozinha, já que o seu marido viajou à trabalho e seus filhos estavam na colônia de férias do colégio. Então ela resolve passar na locadora depois da academia e alugar alguns filmes mais românticos, já que estava se sentindo carente. Ela convidou algumas amigas para assistirem os filmes juntas depois do lanche da tarde. Ao chegar à locadora ela se dirige ao terminal de auto atendimento e busca por “romance”, onde não obtém muitos resultados, apenas filmes que possuem no seu título a palavra romance. **Roteiro 1***

História de Usabilidade 1: Maria das Dores

José é um jovem universitário, muito interessado em pesquisas e inovações tecnológicas. Possui perfil em diversas redes sociais como orkut, Myspace e Hi5, além de blog pessoal. Utiliza o MSN e o Google Talk como principal meio de contatar os amigos. Gosta de ler revistas e livros, além de assistir seriados, filmes e animes. Se atualiza das notícias através de sites especializados. Gosta também de fotografia digital, utilizando câmeras fotográficas de última geração e celulares para registrar os lugares por onde passa. Seu celular possui diversos recursos como: fotos, vídeos, agenda de eventos, acesso a internet, leitor de arquivos diversos, TV e jogos.

Persona 2

*José estava se preparando para o carnaval, iria passar como em todos os últimos anos em casa assistindo filmes. Na véspera do feriado, ao chegar a locadora ele percebe que muitos filmes já foram alugados e ele precisa saber quais dos que sobraram não foram ainda alugados por ele, para que ele não os realugue já que ele aluga muitos filmes. Ele então acessa o terminal, faz login e antes de efetuar a pesquisa ele marca a opção “Não mostrar filmes que já aluguei”, pesquisa então pelo assunto do seu interesse: “ação” e seleciona os que ele escolheu, fazendo assim a sua locação. **Roteiro 2***

História de Usabilidade 2: José

*Antônio é um jovem estudioso, identifica-se com a área de filosofia e artes. Gosta de ler livros e artigos científicos, principalmente os utilizados em sua faculdade. Não é muito aberto a inovações tecnológicas, prefere os meios impressos à internet. Seu principal hobby é ir ao cinema. Embora utilize o computador para realizar algumas tarefas do seu dia-a-dia, tem certa dificuldade em desempenhá-las, várias vezes tendo de recorrer a colegas e amigos, mais experientes nesta área. **Persona 3***

Antônio estava interessando em filmes para assistir com sua namorada na noite de sábado, e no domingo pela tarde, já que estava chovendo muito naquele final de semana. Chega o sábado e Antônio vai à locadora em busca de filmes de comédia romântica. Ao fazer a pesquisa por filmes deste gênero no terminal, ele resolve alugar três filmes, mas não consegue selecionar mais de um filme, levando ele à requisitar ajuda do funcionário, que o instrui

a utilizar um atalho presente no sistema mas que não estava explícito e perceptível ao usuários. Roteiro 3

História de Usabilidade 3: Antônio

A partir das Histórias mapeadas, é possível perceber a definição de perfis bastante diversificados em relação às expectativas de uso do sistema que estava sendo desenvolvido. É natural que para um sistema de busca de mídias em locadoras de DVDs, o perfil de usuários seja bastante heterogêneo e com características e necessidades bastante peculiares em relação ao sistema com o qual irão interagir. Por essa razão, a utilização da Análise de Contexto de Uso fornece importantes e indispensáveis contribuições para o desenvolvimento centrado no usuário. Ao final do mapeamento das Histórias, os dois membros da equipe redesenharam a interface de busca de mídias para contemplar os diversos aspectos de usuários e tarefas pertinentes àquele projeto e que não tinham sido observados durante a primeira etapa do estudo. Baseado nesses aspectos, a dupla rascunhou em papel os *Templates* e Protótipos Simplificados para as novas interfaces e em seguida, iniciaram as atividades de redesenho (Desenho da Interação). O *Template* e o Protótipo da interface de busca de mídias são apresentados nas Figuras 5.9 e 5.10, respectivamente.

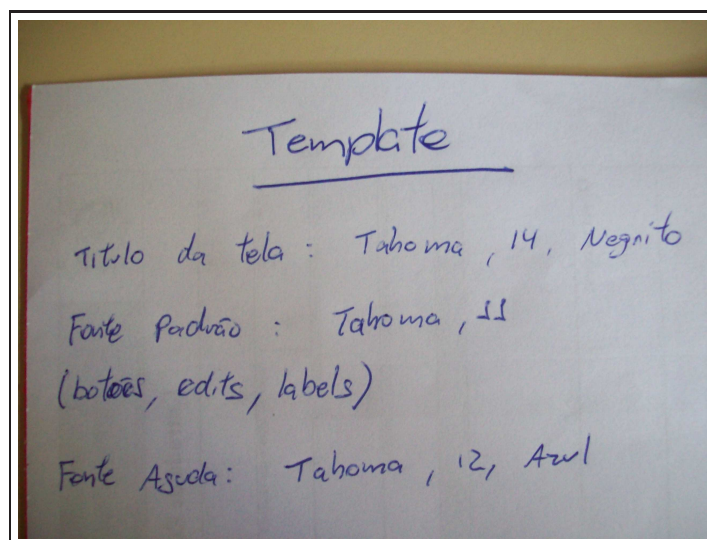


Figura 5.9. *Template* gerado utilizando o método XPu.

As análises realizadas e a geração e discussão do Protótipo proporcionaram um considerável ganho de usabilidade no novo Desenho da Interação. A Figura 5.11 apresenta a nova tela gerada para essa funcionalidade.

As Avaliações de Usabilidade realizadas em todas as interfaces ao final da iteração (assim como todas as demais atividades de usabilidade) foram feitas pelos próprios

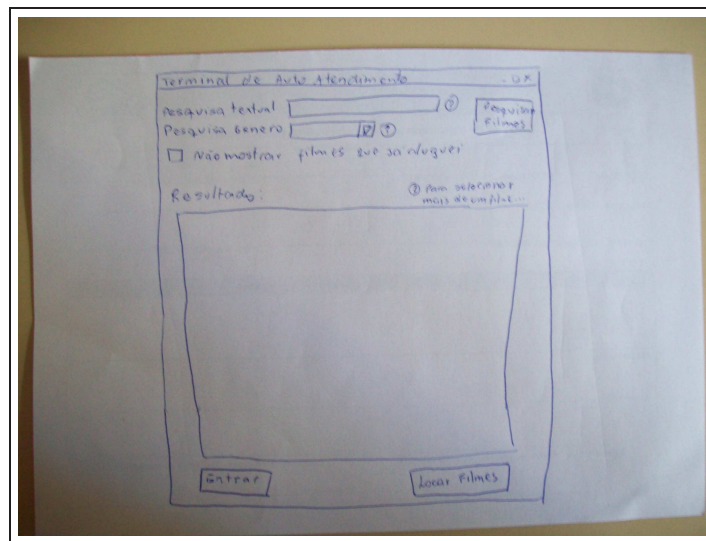


Figura 5.10. Protótipo gerado utilizando o método XPu.

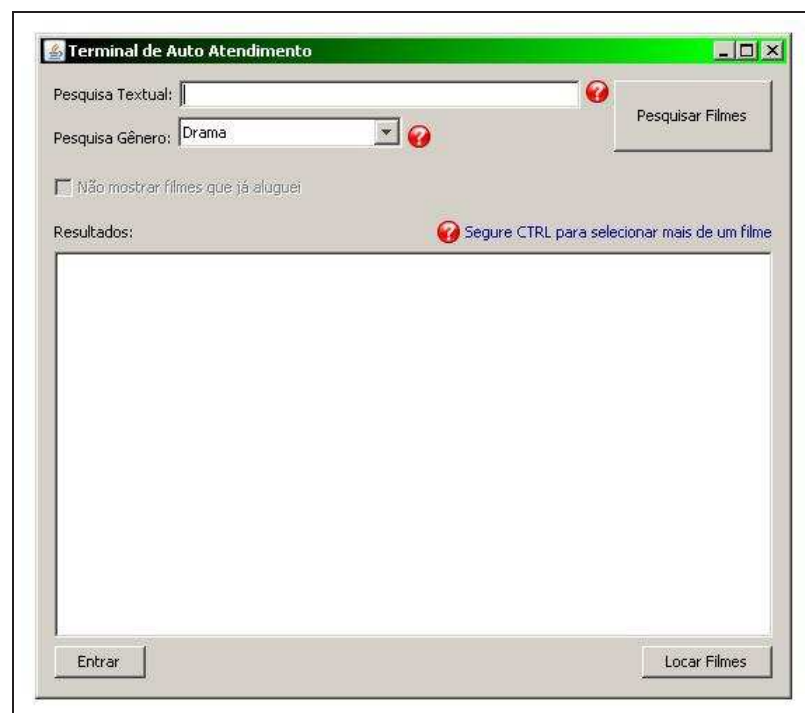


Figura 5.11. Tela de Busca gerada utilizando o método XPu.

desenvolvedores (no papel de Projetista de Interação) utilizando a Avaliação Heurística. A Avaliação Heurística foi realizada por cada um dos membros da equipe separadamente utilizando como insumo os Cenários de Teste providos pelas Histórias de Usabilidade mapeadas na Análise de Contexto de Uso. O relatório com os resultados da avaliação está descrito no Anexo 1 deste trabalho.

Paralelamente, os dados relativos ao tempo consumido pelas atividades do XPu

foram sendo coletados ao longo do desenvolvimento pela atividade de Controle de Usabilidade. Em um projeto real, esses dados seriam coletados apenas para cada uma das tarefas que haviam sido mapeadas no planejamento e estimativa das Histórias de Usuário, no entanto para esse estudo de caso, um dos desenvolvedores, que também atuou no papel de *Tracker* na primeira etapa do estudo, também coletou separadamente os tempos consumidos por cada uma das atividades de usabilidade que foram utilizados naquela iteração. Através desses dados é que foi possível ter uma idéia clara do tempo que foi consumido pelas atividades de usabilidade (ou seja, do tempo que seria efetivamente consumido em um projeto utilizando o método XPu desde o princípio no processo de desenvolvimento) e pelo re-trabalho gerado pela percepção de falhas na interface que havia sido gerada anteriormente pelo método XP tradicional. Os resultados coletados separadamente para as atividades de usabilidade estão descritos na Tabela 5.1.

<i>Atividades relativas ao XPu</i>	<i>Tempo consumido</i>
Personas e Roteiros	1:40
<i>Template</i> e Protótipo de Interface	0:30
Implementação de Tela	0:40
Outras Implementações	2:00

Tabela 5.1. Tempo consumido pelas atividades relativas ao método XPu

Por fim, o produto foi entregue ao Cliente para os Testes de Aceitação, que validou a entrega a partir de uma inspeção manual das funcionalidades apresentadas.

5.2.4 Análise e Interpretação dos Resultados

Esta seção apresenta a discussão e a interpretação dos resultados observados durante a realização do estudo de caso descrito. As subseções seguintes apresentam a análise dos dados coletados, a análise qualitativa dos resultados de medição e a verificação das hipóteses levantadas para o estudo.

5.2.4.1 Análise dos Dados

Baseado nos objetivos do estudo de caso, os dados a serem levantados dizem respeito à qualidade da interface gerada pelo método XPu em relação à sobrecarga de trabalho imposta por ele. Para tanto, a interface de busca de mídias (que sofreu consideráveis modificações devido à adoção do XPu) foi inspecionada pelo autor deste trabalho através da Avaliação Heurística antes e depois da utilização do método proposto para

verificar o número de erros encontrados em cada versão. O número de erros levantado para essas interfaces estão apresentados nas Tabelas 5.2 e 5.3 e a descrição de cada uma das heurísticas utilizadas se encontra no Apêndice 1 deste trabalho. Já os dados relativos ao tempo consumido pelas atividades do método XP e do método XPu foram separadamente isolados e registrados. Para tabular esses dados, a Tabela 5.4 apresenta o número total de erros encontrados nas interfaces geradas pelos dois métodos, bem como os seus respectivos tempos totais consumidos.

As atividades de confecção das Personas e dos Roteiros e da geração do *Template* e do Protótipo de Interface para o projeto descritas anteriormente na Tabela 5.1 foram contabilizadas como atividades diretamente resultantes do método XPu e estão descritas na Tabela 5.4. Já as atividades de Implementação de Tela e Outras Implementações não foram contabilizadas como atividades do XPu (assim como o tempo consumido pelas tarefas na segunda etapa do estudo para correção de erros não relacionados à usabilidade), pois configuraram retrabalho devido ao desenvolvimento do estudo de caso ter sido estruturado em duas etapas de desenvolvimento para o mesmo produto. Isto é, em um cenário de desenvolvimento real, espera-se que esse tempo não seja de fato despendido, já que o produto não terá sido previamente desenvolvido e as interfaces não terão que ser redesenhadas. Finalmente, com relação à Avaliação Heurística, esta também não foi contabilizada no tempo total consumido pelo XPu, já que dado os objetivos de medição do estudo de caso, o interesse do trabalho era o de contabilizar a que custo as interfaces seriam melhoradas; e a avaliação, apesar de fundamental, serve apenas de constatação (ou não) da qualidade da interface produzida.

<i>Heurística</i>	<i>Número de erros</i>
H1	1
H2	1
H3	1
H4	0
H5	1
H6	1
H7	1
H8	1
H9	0
H10	1

Tabela 5.2. Avaliação da tela de busca gerada pelo método XP

A Tabela 5.4 mostra que as atividades de usabilidade do método XPu sobrecarregaram o trabalho da equipe em 2 horas e 10 minutos (Personas, Roteiros,

<i>Heurística</i>	<i>Número de erros</i>
H1	0
H2	1
H3	0
H4	0
H5	1
H6	1
H7	0
H8	0
H9	0
H10	1

Tabela 5.3. Avaliação da tela de busca gerada pelo método XPu

<i>Resultados</i>	<i>XP</i>	<i>XPu</i>
Erros encontrados	8	4
Tempo consumido	10:30	2:10

Tabela 5.4. Comparativo dos resultados para os métodos XP e XPu

Template e Protótipo de Interface) e reduziram pela metade o número de erros encontrados na interface gerada. Ou seja, com cerca de 20% a mais de tempo investido nas atividades de usabilidade, o método XPu reduziu pela metade o número de defeitos encontrados na interface produzida e entregue ao usuário final para esse estudo de caso. Uma outra informação que é possível deduzir a partir dos dados apresentados é a de que o não investimento em usabilidade (como foi feito na primeira etapa do estudo de caso) gerou 2 horas e 40 minutos a mais de re-trabalho (Implementação de Tela e Outras Implementações) para corrigir os erros de interface que foram percebidos na segunda etapa do estudo, isto é, foi investido cerca de 25% a mais de tempo para correção de problemas de interface que poderiam ter sido evitados, caso as atividades de usabilidade do XPu tivessem sido adotadas desde o início do projeto.

No entanto, apesar dos dados apresentados, seria prematuro afirmar que esse quadro de melhoria provido pela adoção do método XPu se repetiria em outros contextos e até mesmo que o conhecimento sobre o sistema adquirido pela equipe na primeira etapa do estudo não influenciou o resultado de alguma maneira. Sabemos através da literatura que o investimento em atividades de usabilidade comprovadamente melhora a qualidade das interfaces geradas. O desafio do estudo, portanto, é analisar a que custo essa melhoria é alcançada através do método XPu. Os dados preliminares apresentados aqui sugerem que esse custo é relativamente baixo, visto que a adoção

das atividades de usabilidade consome um tempo relativamente pequeno da equipe de desenvolvimento, mantém o processo com as características que se espera para abordagens ágeis e ainda evita o re-trabalho relacionado à correção de problemas de interface em etapas avançadas do desenvolvimento. Além disso, em entrevista com os desenvolvedores após o estudo de caso, eles relataram se sentirem mais seguros e confortáveis com a qualidade do produto que estavam desenvolvendo, após realizarem uma análise mais apurada dos usuários finais e das tarefas que estes realizariam sobre o sistema.

5.2.4.2 Análise dos Resultados de Medição

Esta subseção analisa os resultados alcançados no estudo de caso, baseado na definição dos objetivos de medição apresentados no início desta seção. São apresentados abaixo o objetivo da medição e a discussão dos resultados alcançados relativos às estratégias estabelecidas para as medições.

OM1: Avaliar a viabilidade de utilização conjunta das práticas do Praxis-u e do método XP.

A estratégia estabelecida para esse objetivo era observar a integração das atividades de usabilidade ao longo do ciclo de vida ágil durante o desenvolvimento do produto, registrar possíveis contenções e realizar uma entrevista pós-teste para discutir com os participantes as principais dificuldades encontradas na instanciação das atividades do Praxis-u ao método XP. Baseado nessa métrica, a equipe relatou a utilização das práticas propostas sem grandes dificuldades ou limitações, dado o escopo reduzido do estudo de caso realizado. No entanto, um dos membros relatou certa dificuldade em estimar as tarefas na segunda etapa do estudo, já que estas envolveriam as atividades de usabilidade propostas e ele não tinha experiência com as técnicas de IHC utilizadas, logo não se sentiu confortável estimando uma tarefa que envolvia atividades nunca realizadas por ele. Esse fato mostra a importância de que o papel de Projetista de Interação seja desempenhado por um desenvolvedor com alguma experiência multidisciplinar em IHC ou por um profissional de IHC que seja efetivamente integrado à equipe ágil e participe ativamente no acompanhamento das atividades de desenvolvimento. Por fim, não foram observadas contenções na integração entre as atividades ágeis do XP e as atividades de usabilidade do XPu e a utilização das práticas conjuntas foi considerada viável pelo time.

OM2: Avaliar a viabilidade de particionamento das atividades do Praxis-u para utilização em um contexto iterativo e incremental.

A estratégia estabelecida para esse objetivo era verificar a qualidade da interface produzida após o desenvolvimento do produto utilizando o método XPu, que por sua vez utiliza as atividades particionadas do Praxis-u. Baseado nessa métrica e segundo relatos da equipe, as atividades do Praxis-u utilizadas pelo XPu se adaptaram ao ciclo iterativo e incremental e geraram um ganho positivo na qualidade da interface gerada para a aplicação de busca de mídias, como pode ser percebido na Tabela 5.4. Além disso, o time não encontrou dificuldades na utilização prática do processo para esse estudo, demonstrando que o particionamento das atividades para utilização no ciclo de vida ágil foi eficiente nesse caso.

OM3: Avaliar a qualidade dos protótipos e interfaces de software produzidas utilizando-se o XP em comparação com a utilização do método proposto.

A estratégia estabelecida para esse objetivo era aplicar uma avaliação por inspeção nas interfaces geradas pelo método XP e pelo método XPu e comparar o número de problemas encontrados em cada uma delas. Como é possível perceber pela análise das Figuras 5.3 e 5.11, o ganho de qualidade percebido na interface gerada pela equipe com a utilização do método XPu é considerável. Esse fato demonstra que para o estudo de caso em questão, a interface gerada pelo método XPu utilizando as práticas de usabilidade propostas reduziram em 50% o número de erros encontrados através da inspeção realizada, conforme apresentado na Tabela 5.4.

OM4: Avaliar a sobrecarga de trabalho propiciada pela adição das atividades de usabilidade ao método XP.

A estratégia estabelecida para esse objetivo era registrar separadamente o tempo de desenvolvimento gasto em cada uma das histórias de usuário utilizando os métodos XP e XPu. Esses dados foram coletados e estão registrados nas Figuras 5.1 e 5.2 para o método XP e nas Figuras 5.7 e 5.8, além da Tabela 5.1 para o método XPu. Uma análise crítica da sobrecarga de trabalho proporcionada pelo XPu foi apresentada na subseção anterior e demonstra resultados positivos apontando para uma sobrecarga de apenas 20% no trabalho da equipe para um ganho de qualidade em 50% na interface gerada.

OM5: Avaliar a manutenção das características de agilidade do método XP após sua integração com o XPu.

A estratégia estabelecida para esse objetivo era observar a permanência das características ágeis definidas em Abrantes & Travassos [2007b] ao longo do desenvolvimento do produto utilizando o método XPu. Uma caracterização de

agilidade realizada através de uma inspeção do método foi apresentada na primeira seção, no início deste capítulo. O objetivo relacionado à métrica definida para esse objetivo de medição é o de confirmar (ou não) a permanência dessas características na prática de um projeto de desenvolvimento de software. Observou-se ao longo da execução do estudo de caso que as características de incrementalidade, iteratividade, colaboratividade, cooperação, orientação a pessoas e parcimônia (*leanness*) foram mantidas e percebidas claramente pela equipe. Já as características de adaptabilidade e restrição de prazo não puderam ser observadas devido às características limitadas do estudo em questão.

5.2.4.3 Verificação das Hipóteses

Esta subseção analisa os resultados alcançados no estudo de caso, baseado nas hipóteses levantadas e apresentadas no início desta seção. São apresentados abaixo as hipóteses sobre as quais o estudo foi desenvolvido e a discussão dos resultados alcançados relativos à validação ou não de cada uma delas.

Hipótese Nula (H0): A integração proposta pelo XPu acrescenta qualidade à interface final do produto, sem que para isso seja necessária a burocratização do processo, isto é, sem que o método XP seja descaracterizado em sua “agilidade”.

A Hipótese Nula (H0) **foi confirmada** pela execução do estudo de caso. Conforme discutido nas subseções anteriores, o método XPu se mostrou adequado para utilização na prática, já que melhorou consideravelmente a qualidade da interface produzida, mantendo as características ágeis do método XP e proporcionando reduzida sobrecarga de trabalho à equipe de desenvolvimento.

Hipótese Alternativa (H1): A integração proposta pelo XPu acrescenta qualidade à interface final do produto, no entanto a sobrecarga de trabalho imposta à equipe XP sugere dificuldades de utilização do método na prática.

Hipótese Alternativa (H2): As atividades do método XPu não alteram significativamente o tempo de desenvolvimento do produto, no entanto o ganho de qualidade na interface final é considerado pequeno ou insuficiente para justificar a adoção do método na prática.

As Hipóteses Alternativas (H1 e H2) **não se confirmaram** após a execução do estudo de caso.

Capítulo 6

Conclusão

Este capítulo apresenta as conclusões finais sobre o método proposto, bem como suas contribuições para as áreas de pesquisa em Engenharia de Software e IHC. As seções seguintes apresentam as últimas considerações relacionadas à integração de atividades de usabilidade em método ágeis, incluindo um resumo das práticas mapeadas do Praxis-u para o XPu e uma relação dos insumos consumidos e dos artefatos produzidos pelo XPu ao longo das suas atividades. Por fim, são apresentados os trabalhos futuros e as questões de pesquisa em aberto relacionadas à abordagem proposta.

6.1 Considerações Finais

Beck [1999] sempre ressaltou que o XP é um método flexível e adaptável aos diferentes contextos e às diferentes necessidades de desenvolvimento. Esse trabalho propõe uma importante adaptação ao XP do ponto de vista da qualidade de software. Diversos trabalhos na literatura já demonstraram a efetividade da adoção de práticas de usabilidade ao longo do processo de desenvolvimento de software como forma de melhorar a qualidade das interfaces produzidas e conseqüentemente contribuir para a qualidade do produto desenvolvido.

O grande desafio até então era encontrar mecanismos de adaptar complexos processos de usabilidade ao contexto ágil de desenvolvimento ou até mesmo propor novos métodos especialmente definidos de maneira ágil. No primeiro caso, a principal questão era avaliar até que ponto atividades de usabilidade poderiam ser adicionadas ao método, sem descaracterizar sua agilidade. No segundo caso, dificilmente encontraríamos parâmetros suficientes para mensurar a qualidade do processo de usabilidade definido especialmente para aquele fim, sem que para isso utilizássemos técnicas já consagradas nos processos tradicionais, incorrendo novamente

à necessidade de adaptação.

Este trabalho optou por definir o método XPu a partir da primeira opção, haja visto que a comprovada eficiência das técnicas de IHC aqui utilizadas contribuem de maneira inequívoca para o sucesso do desenvolvimento centrado no usuário. A questão de avaliar até que ponto o método definido continua mantendo suas características ágeis foi respondida com a inspeção do método segundo critérios técnicos e objetivos definidos na literatura da área. Um estudo de caso efetivo com usuários em um projeto real contribuiu de maneira significativa para sinalizar a viabilidade de adoção das práticas de usabilidade no contexto ágil, apontando resultados satisfatórios na melhoria da qualidade de uso das interfaces geradas com um reduzido investimento de tempo da equipe de desenvolvimento.

Outros trabalhos relacionados também descreveram importantes contribuições para a integração entre práticas de usabilidade e métodos ágeis, porém alguns deles sem a devida discussão de suas características ágeis, outros pautados em processos de usabilidade não tão bem definidos como o Praxis-u, outros bastante generalistas, não apontando soluções para problemas inerentes ao ciclo diário de desenvolvimento de uma equipe ágil, além de outros focados em métodos ágeis com características gerenciais como o Scrum [Schwaber & Beedle, 2001].

Como já foi discutido ao longo dos capítulos anteriores, durante a definição do XPu, o processo Praxis-u sofreu uma série de adaptações necessárias para que o método resultante preservasse as características ágeis do XP e dos demais métodos ágeis, em especial a característica de Parcimônia (*leanness*). Com o objetivo de sintetizar quais atividades do Praxis-u foram mapeadas para o XPu, a Tabela 6.1 apresenta detalhadamente quais atividades da integração XP/XPu correspondem a cada uma das atividades do Praxis-u. Por fim, as Tabelas 6.2 e 6.3 apresentam uma síntese dos insumos consumidos por cada atividade técnica e gerencial do XPu, além dos artefatos produzidos por elas. A relação das siglas utilizadas para atividades, insumos e artefatos é apresentada no Apêndice 2 deste trabalho.

A partir de uma análise das tabelas apresentadas, é possível perceber que, com exceção das atividades de Revisão da Análise de Usabilidade e Revisão do Desenho da Interação, todas as atividades do Praxis-u foram mapeadas em uma ou mais atividades no XPu, consumindo insumos e produzindo artefatos gerados pelo próprio método XPu ou pelo método XP. A ausência das atividades de revisão é facilmente justificada pela características de Cooperação e Colaboratividade inerentes aos métodos ágeis. A constante interação entre os membros do time proporciona uma revisão natural das atividades desempenhadas pelos seus pares. Uma outra característica que deve ser observada é o papel fundamental das Histórias de Usabilidade que servem de

insumo para diversas outras atividades ao longo do processo de desenvolvimento, fato que justifica o esforço inicial nas atividades que produzem essas histórias, dificultando em alguns casos a total manutenção das características de Adaptabilidade e Incrementalidade dos métodos ágeis.

<i>Atividades do Praxis-u</i>	<i>Atividades do XP/XPu</i>
Análise de Contexto de Uso	AT-AU, AT-AT, AT-ACU
Definição das Funções do Produto	XP-HU
Prototipação de Requisitos de Interface	AT-PI
Revisão da Análise de Usabilidade	-
Definição do Estilo de Interação	AT-DEI
Desenho da Interação	AT-DI
Revisão do Desenho da Interação	-
Avaliação de Usabilidade	AT-AU
Definição de Requisitos e Metas de Usabilidade	AG-DRMU
Planejamento de Usabilidade	AG-PU
Controle de Usabilidade	AG-CU

Tabela 6.1. Mapeamento das atividades do Praxis-u para o XPu.

<i>Atividades Técnicas do XPu</i>	<i>Insumos</i>	<i>Artefatos</i>
AT-AU	XP-HU	IA-P
AT-AT	XP-HU	IA-R
AT-ACU	IA-HU	IA-RU e IA-CT
AT-DEI	XP-HU e IA-RU	IA-T
AT-DI	IA-T	IA-IF
AT-PI	IA-T	IA-PS
AT-AU	IA-B, IA-CT e IA-PI	IA-TU

Tabela 6.2. Atividades Técnicas do XPu - Insumos e Artefatos.

<i>Atividades Gerenciais do XPu</i>	<i>Insumos</i>	<i>Artefatos</i>
AG-DRMU	XP-HU e XP-PR	IA-B
AG-PU	IA-RU, IA-B e IA-RE	IA-ER
AG-CU	XP-VP	IA-RE

Tabela 6.3. Atividades Gerenciais do XPu - Insumos e Artefatos.

6.2 Trabalhos Futuros

Diversos trabalhos futuros podem resultar do método XPu e dos resultados de avaliação aqui apresentados. Os primeiros deles certamente estão relacionados a novas avaliações e validações pelas quais o método poderia passar. Uma proposta de estudo de caso em um ambiente real de desenvolvimento ágil poderia fornecer fortes indícios da aplicabilidade do método. A variação do nível de experiência em técnicas ágeis e de IHC também seria um fator interessante a ser observado em trabalhos futuros, incluindo estudos que demonstrassem uma comparação entre a interdisciplinaridade na equipe (por exemplo, utilizando *designers* profissionais no papel de Projetistas de Interação) e a habilidade individual dos desenvolvedores em lidar com as referidas técnicas de IHC. O revezamento do papel de Projetista de Interação entre os membros do time também é um importante fator a ser observado em estudos posteriores.

Outro trabalho com potencial para ser desenvolvido é a modelagem do XPu em um sistema BPMS (*Business Process Management System*). Como em qualquer processo modelado em um sistema BPMS, o XPu poderia ser executado através de ferramentas que permitem definir papéis, fluxos e responsabilidades ao longo de todo o ciclo de vida de desenvolvimento do produto. Quando esta dissertação estava sendo finalizada, uma iniciativa com esse objetivo estava em desenvolvimento como trabalho final de graduação, realizando a modelagem do XPu através da ferramenta Intalio.

Finalmente, trabalhos relacionados à adequação do XPu aos parâmetros estabelecidos por modelos de qualidade de processos de usabilidade como o UMM (*Usability Maturity Model*) [Earthy, 1999] formam outra importante fonte de experimentos. Definir como aplicar modelos de maturidade e como avaliar processos de usabilidade ainda é um desafio para a comunidade ágil e certamente ainda requer grandes contribuições em pesquisa na área.

6.3 Publicações Resultantes

- SILVA, T. R.; PADUA, C. I. P. S.; RESENDE, R. S. F. XPu: caracterização de agilidade para uma abordagem ágil visando à usabilidade. In: III Workshop de Desenvolvimento Rápido de Aplicações, 2009, Ouro Preto/MG. Anais do VIII Simpósio Brasileiro de Qualidade de Software.
- SILVA, T. R.; PADUA, C. I. P. S. XPu: uma extensão do método XP visando à usabilidade. In: XXXV Conferência Latino-Americana de Informática, 2009, Pelotas/RS. Anais do CLEI 2009.

Referências Bibliográficas

- Abrantes, J. F. & Travassos, G. H. (2007a). Caracterização de métodos ágeis de desenvolvimento de software. Technical report, Programa de Engenharia de Sistemas e Computação - COPPE/UFRJ, Rio de Janeiro/RJ.
- Abrantes, J. F. & Travassos, G. H. (2007b). Caracterização de métodos ágeis de desenvolvimento de software. In *Anais do Workshop de Desenvolvimento Rápido de Aplicações do Simpósio Brasileiro de Qualidade de Software (WDRA - SBQS 2007)*, Porto de Galinhas/PE.
- Alavi, M. (1984). An assessment of the prototyping approach to information systems development. *Commun. ACM*, 27(6):556–563.
- Armitage, J. (2004). Are agile methods good for design? *interactions*, 11(1):14–23.
- Aureliano, V. C. O. (2007). Extreme Communication-Centered Design: um processo ágil para o projeto da interação humano-computador. Master's thesis, Programa de Pós- Graduação em Informática da Pontifícia Universidade Católica do Rio de Janeiro.
- Barbosa, D. F.; Furtado, E. S. & Gomes, A. S. (2008). Uma Estratégia de Apoio à Institucionalização da Usabilidade em Ambientes de Desenvolvimento Ágil. In *IHC '08: Proceedings of the VIII Brazilian Symposium on Human Factors in Computing Systems*, pp. 214–223, Porto Alegre, Brazil. Sociedade Brasileira de Computação.
- Basili, V.; Caldeira, G. & Rombach, H. (1994). *Experience Factory*, volume 1, pp. 469–476. Wiley Publishing, Inc.
- Beck, K. (1999). *Extreme Programming Explained: Embrace Change*. Addison-Wesley Professional.
- Beck, K. (2002). *Test Driven Development: by Example*. Addison-Wesley Longman.

- Beck, K. & Andres, C. (2004). *Extreme Programming Explained: Embrace Change*. Addison-Wesley Professional, 2a. edição.
- Beck, K.; Beedle, M.; van Bennekum, A.; Cockburn, A.; Cunningham, W.; Fowler, M.; Grenning, J.; Highsmith, J.; Hunt, A.; Jeffries, R.; Kern, J.; Marick, B.; Martin, R. C.; Mellor, S.; Schwaber, K.; Sutherland, J. & Thomas, D. (2001). Manifesto for Agile Software Development. Disponível em: <http://www.agilemanifesto.org>.
- Bernardo, P. C. & Kon, F. (2007). Desenvolvendo com agilidade: experiências na reimplantação de um sistema de grande porte. In *Proceedings of the 1st Workshop on Rapid Application Development (WDRA'2007) in the Brazillian Symposium on Software Quality (SBQS'2007)*.
- Boehm, B. (1986). A spiral model of software development and enhancement. *SIGSOFT Softw. Eng. Notes*, 11(4):14–24.
- Boehm, B. W.; Gray, T. E. & Seewaldt, T. (1984). Prototyping versus specifying: a multiproject experiment. *IEEE Trans. Softw. Eng.*, 10(3):290–302.
- Broschinsky, D. & Baker, L. (2008). Using Persona with XP at LANDesk Software, an Avocent Company. In *AGILE '08: Proceedings of the Agile 2008*, pp. 543–548, Washington, DC, USA. IEEE Computer Society.
- Cockburn, A. (2004). *Crystal Clear: a Human-Powered Methodology for Small Teams*. Addison-Wesley Professional.
- Cockburn, A. (2006). *Agile Software Development: The Cooperative Game*. Addison Wesley Professional, 2a. edição.
- Cohn, M. (2005). *Agile Estimating and Planning*. Prentice Hall PTR.
- Constantine, L. L. (2001). Process Agility and Software Usability: Toward Lightweight Usage-Centered Design. *Information Age*, 8(2).
- Constantine, L. L. & Lockwood, L. A. D. (1999). *Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design*. Addison-Wesley.
- Constantine, L. L. & Lockwood, L. A. D. (2002). Usage-Centered Engineering for Web Applications. *IEEE Software*, 19 (2):42–50.
- Constantine, L. L. & Lockwood, L. A. D. (2003). Usage-Centered Software Engineering: an agile approach to integrating users, user interfaces, and usability into Software

- Engineering practice. In *ICSE '03: Proceedings of the 25th International Conference on Software Engineering*, pp. 746–747, Washington, DC, USA. IEEE Computer Society.
- Cooper, A. & Reimann, R. M. (2003). *About Face 2.0: The Essentials of Interaction Design*. Wiley Publishing, Inc., Indiana, USA.
- da Silva, A. F.; Kon, F. & Torteli, C. (2005). XP South of the Equator: an experience implementing XP in Brazil. In *Proceedings of the 6th International Conference on eXtreme Programming and Agile Processes in Software Engineering (XP'2005)*, pp. 10–18.
- de Souza, C. S. (2005). *The Semiotic Engineering of Human-Computer Interaction*. The MIT Press, Cambridge.
- Detweiler, M. (2007). Managing UCD within agile projects. *interactions*, 14(3):40–42.
- Dybå, T. & Dings, T. (2008). Empirical studies of agile software development: a systematic review. *Inf. Softw. Technol.*, 50(9-10):833–859.
- Earthy, J. (1999). Usability Maturity Model: Processes. Technical report, Lloyd's Register of Shipping.
- Ehn, P. (1990). *Work-Oriented Design of Computer Artifacts*. L. Erlbaum Associates Inc., Hillsdale, NJ, USA.
- Ericsson, K. & Simon, H. (1993). *Protocol Analysis: Verbal Reports as Data*. MIT Press, Boston, 2a. edição.
- Ferreira, J.; Noble, J. & Biddle, R. (2007). Agile Development Iterations and UI Design. In *AGILE '07: Proceedings of the AGILE 2007*, pp. 50–58, Washington, DC, USA. IEEE Computer Society.
- Fowler, M.; Beck, K.; Brant, J.; Opdyke, W. & Roberts, D. (1999). *Refactoring: improving the design of existing code*. Addison-Wesley Professional.
- Fox, D.; Sillito, J. & Maurer, F. (2008). Agile Methods and User-Centered Design: How These Two Methodologies are Being Successfully Integrated in Industry. In *AGILE '08: Proceedings of the Agile 2008*, pp. 63–72, Washington, DC, USA. IEEE Computer Society.
- Galitz, W. O. (2002). *The Essential Guide to User Interface Design*. John Wiley.

- Gilb, T. (1984). Software Engineering: using Design by Objectives Tools (DBO). *SIGSOFT Softw. Eng. Notes*, 9(2):104–113.
- Hackos, J. T. & Redish, J. C. (1998). *User and Task Analysis for Interface Design*. John Wiley & Sons.
- Harper, B.; Slaughter, L. & Norman, K. (1997). Questionnaire administration via the WWW: a validation and reliability study for a user satisfaction questionnaire. In *WebNet 97 - Association for the Advancement of Computing in Education*, Toronto, Canada.
- Highsmith, J. (1997). Messy, exciting and anxiety-ridden: Adaptive Software Development. *American Programmer*, 10(4):23–29.
- Hix, D. & Hartson, H. R. (1993). *Developing User Interfaces: Ensuring Usability Through Product & Process*. John Wiley.
- Hodgetts, P. (2005). Experiences Integrating Sophisticated User Experience Design Practices into Agile Processes. In *ADC '05: Proceedings of the Agile Development Conference*, pp. 235–242, Washington, DC, USA. IEEE Computer Society.
- Holzinger, A.; Errath, M.; Searle, G.; Thurnher, B. & Slany, W. (2005). From Extreme Programming and Usability Engineering to Extreme Usability in Software Engineering Education (XP+UE=XU). In *COMPSAC '05: Proceedings of the 29th Annual International Computer Software and Applications Conference (COMPSAC'05) Volume 2*, pp. 169–172, Washington, DC, USA. IEEE Computer Society.
- ISO/IEC-12207 (2002). *Systems and Software Engineering - Software Life Cycle Processes*. International Standards Organization.
- ISO/IEC-13407 (1999). *Human Centred Design Process for Interactive Systems*. International Standards Organization.
- ISO/IEC-9126 (2001). *Software Engineering - Product Quality - Part 1: Quality Model*. International Standards Organization.
- ISO/IEC-9241 (1997). *Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs)*. International Standards Organization.
- Jung, C. F. (2004). *Metodologia para pesquisa e desenvolvimento aplicada a novas tecnologias, produtos e processos*. Axcel Books do Brasil Editora, Rio de Janeiro/RJ.

- Kane, D. (2003). Finding a Place for Discount Usability Engineering in Agile Development: Throwing Down the Gauntlet. In *ADC '03: Proceedings of the Conference on Agile Development*, Washington, DC, USA. IEEE Computer Society.
- Kotonya, G. & Sommerville, I. (1998). *Requirements Engineering: Processes and Techniques*. Wiley Publishing, Inc.
- Larman, C. (2003). *Agile and Iterative Development: a Manager's Guide*. Addison-Wesley Professional.
- Lee, J. C. (2006). Embracing agile development of usable software systems. In *CHI '06: CHI '06 Extended Abstracts on Human Factors in Computing Systems*, pp. 1767–1770, New York, NY, USA. ACM.
- Lee, J. C. & McCrickard, D. S. (2007). Towards Extreme(ly) Usable Software: Exploring Tensions Between Usability and Agile Software Development. In *AGILE '07: Proceedings of the AGILE 2007*, pp. 59–71, Washington, DC, USA. IEEE Computer Society.
- Lievesley, M. A. & Yee, J. S. R. (2006). The role of the interaction designer in an agile software development process. In *CHI '06: CHI '06 Extended Abstracts on Human Factors in Computing Systems*, pp. 1025–1030, New York, NY, USA. ACM.
- Lui, K. M. & Chan, K. C. C. (2004). Test Driven Development and Software Process Improvement in China. In *Proceedings of the 5th International Conference on eXtreme Programming and Agile Processes in Software Engineering (XP'2004)*, pp. 219–222. Lecture Notes on Computer Science.
- Mafra, S. N. & Travassos, G. H. (2006). Estudos Primários e Secundários apoiando a busca por Evidências em Engenharia de Software. Technical Report RT - ES 687/06, Programa de Engenharia de Sistemas e Computação - COPPE/UFRJ, Rio de Janeiro/RJ.
- Mann, C. & Maurer, F. (2005). A case study on the impact of Scrum on overtime and customer satisfaction. In *Agile 2005 Conference*, pp. 70–79.
- Marconi, M. A. & Lakatos, E. (2003). *Fundamentos de metodologia científica*. Editora Atlas, São Paulo/SP.
- Mayhew, D. (1999). *The Usability Engineering Lifecycle: a practitioner's handbook for user interface design*. Morgan Kaufmann Publishers, Inc.

- McInerney, P. & Maurer, F. (2005). UCD in agile projects: dream team or odd couple? *interactions*, 12(6):19–23.
- Meszaros, G. & Aston, J. (2006). Adding Usability Testing to an Agile Project. In *AGILE '06: Proceedings of the conference on AGILE 2006*, pp. 289–294, Washington, DC, USA. IEEE Computer Society.
- Muller, R. A. (2004). Extreme Programming in a university project. In *Proceedings of the 5th International Conference on eXtreme Programming and Agile Processes in Software Engineering (XP'2004)*, pp. 312–315. Lecture Notes on Computer Science.
- Najafi, M. & Toyoshiba, L. (2008). Two Case Studies of User Experience Design and Agile Development. *AGILE Conference*, 0:531–536.
- Nielsen, J. (1990). Paper versus computer implementations as mockup scenarios for Heuristic Evaluation. In *INTERACT '90: Proceedings of the IFIP TC13 Third International Conference on Human-Computer Interaction*, pp. 315–320, Amsterdam, The Netherlands. North-Holland Publishing Co.
- Nielsen, J. (1993). *Usability Engineering*. Morgan Kaufmann, 1a. edição.
- Nielsen, J. (1994). *Heuristic Evaluation*. Usability Inspection Methods. John Wiley & Sons, New York, NY, USA.
- Pádua, C. I. P. S.; Pimentel, B.; Paula-Filho, W. P. & Machado, F. T. (2006). Transitioning model-driven development from academia to real life. In *Proceedings of Educators' Symposium of the ACM / IEEE 9th International Conference on Model Driven Engineering Languages and Systems*, pp. 61–77, Gênova, Itália.
- Pai, M.; McCulloch, M. & Gorman, J. D. (2004). Systematic reviews and meta-analyses: an illustrated, step-by-step guide. *The National Medical Journal of India*, 17(2).
- Palmer, S. R. & Felsing, J. M. (2002). *A Practical Guide to Feature Driven Development*. Prentice Hall.
- Patton, J. (2002). Hitting the target: adding interaction design to agile software development. In *OOPSLA '02: OOPSLA 2002 Practitioners Reports*, New York, NY, USA. ACM.
- Paula-Filho, W. P. (2003). *Engenharia de Software: Fundamentos, Métodos e Padrões*. LTC, 2a. edição.

- Pfleeger, S. (1994). Experimental Design and Analysis in Software Engineering - Part 1-5. *ACM Sigsoft - Software Engineering Notes*, 19-20.
- Poppendieck, M. & Poppendieck, T. (2003). *Lean Software Development: an Agile Toolkit for Software Development Managers*. Addison-Wesley Professional.
- Prates, R. O. & Barbosa, S. D. J. (2007). Introdução à Teoria e Prática da Interação Humano Computador fundamentada na Engenharia Semiótica. In *Jornadas de Atualização em Informática (JAI), JAI/SBC 2007*. Sociedade Brasileira de Computação.
- Preece, J.; Rogers, Y. & Sharp, H. (2002). *Interaction Design*. John Wiley and Sons, London.
- Preece, J.; Rogers, Y.; Sharp, H.; Benyon, D.; Holland, S. & Carey, T. (1994). *Human-Computer Interaction*. Addison-Wesley, Reading, MA.
- Robson, C. (1993). *Real World Research: a resource for Social Scientists and Practitioners - Researchers*. Blackwell.
- Rosson, M. B. & Carroll, J. M. (2002). *Usability Engineering: Scenario-Based Development of Human-Computer Interaction*. Morgan Kaufmann, San Francisco, CA.
- Royce, W. W. (1970). Managing the development of large software systems. In *Proceedings of IEEE Wescon*.
- Rubin, J. (1994). *Handbook of Usability Testing: how to plan, design, and conduct effective tests*. John Wiley and Sons.
- RUP (2000). *Rational Unified Process*.
- Sato, D.; Bassi, D.; Bravo, M.; Goldman, A. & Kon, F. (2006). Experiences tracking agile projects: an empirical study. *Journal of the Brazilian Computer Society, Special Issue on Experimental Software Engineering*, 12(3):45–64.
- Schwaber, K. & Beedle, M. (2001). *Agile Software Development with SCRUM*. Prentice Hall.
- Singh, M. (2008). U-SCRUM: an Agile Methodology for Promoting Usability. *AGILE Conference*, 0:555–560.

- Solingen, R. & Berghout, E. (1999). *The Goal/Question/Metric Method: a Practical Guide for Quality Improvement of Software Development*. McGraw-Hill Companies, UK.
- Sommerville, I. (2003). *Engenharia de Software*. Addison Wesley, São Paulo.
- Sousa, K.; Furtado, E. & Mendonça, H. (2005). UPi: a software development process aiming at usability, productivity and integration. In *CLIHC '05: Proceedings of the 2005 Latin American conference on Human-computer interaction*, pp. 76–87, New York, NY, USA. ACM.
- Stake, R. (1995). *The Art of Case Study Research*. SAGE Publications.
- Stamelos, I. & Sfetsos, P. (2007). *Agile Software Development Quality Assurance*. IGI Global.
- Stapleton, J. (1997). *DSDM: a framework for business centered development*. Addison-Wesley Professional.
- Travassos, G. H.; Gurov, D. & do Amaral, E. A. G. (2002). Introdução à Engenharia de Software Experimental. Technical Report RT - ES 590/02, Programa de Engenharia de Sistemas e Computação - COPPE/UFRJ, Rio de Janeiro/RJ.
- Ungar, J. (2008). The Design Studio: Interface Design for Agile Teams. *AGILE Conference*, 0:519–524.
- Vasconcelos, C. R.; Garcia, F. P. & Turnell, M. F. Q. V. (2003). Integrando Usabilidade e Engenharia de Software: um modelo para o desenvolvimento de sistemas centrado no usuário. In *WIHC-ES2003 - Integrating Human-Computer Interaction and Software Engineering Models and Processes*, Rio de Janeiro/RJ. I Latin American Conference on Human-Computer Interaction - CLIHC 2003.
- Wells, J. D. (2006). Extreme Programming: a gentle introduction. Disponível em: <http://www.extremeprogramming.org>.
- Wharton, C.; Rieman, J.; Lewis, C. & Polson, P. (1994). *The Cognitive Walkthrough Method: a Practitioner's Guide*. John Wiley & Sons, New York, NY, USA.
- Wöhlin, C.; Runeson, P.; Höst, M.; Ohlsson, M. C.; Regnell, B.; Wesslén, A. & von Mayrhauser, A. (2000). *Experimentation in Software Engineering: an Introduction*. The Kluwer International Series in Software Engineering. Kluwer Academic Publishers, Norwell, USA.

- Wolkerstorfer, P.; Tscheligi, M.; Sefelin, R.; Milchrahm, H.; Hussain, Z.; Lechner, M. & Shahzad, S. (2008). Probing an agile usability process. In *CHI '08: CHI '08 Extended Abstracts on Human Factors in Computing Systems*, pp. 2151–2158, New York, NY, USA. ACM.
- Yin, R. (1994). *Case Study Research Design and Methods*. SAGE Publications, Beverly Hills, California.

Apêndice A

Descrição de Heurísticas

Segundo Nielsen [1994], as 10 heurísticas a serem avaliadas são:

- **H1** - Visibilidade do estado do sistema
- **H2** - Correspondência entre o sistema e o mundo real
- **H3** - Controle e liberdade do usuário
- **H4** - Consistência e padronização
- **H5** - Prevenção de erro
- **H6** - Ajuda aos usuários para reconhecerem, diagnosticarem e se recuperarem de erros
- **H7** - Reconhecimento em vez de memorização
- **H8** - Flexibilidade e eficiência de uso
- **H9** - Design estético e minimalista
- **H10** - Ajuda e documentação

Apêndice B

Legenda para o XP/XPu

<i>Sigla</i>	<i>Atividade</i>
AT-AU	Análise de Usuários
AT-AT	Análise de Tarefas
AT-ACU	Análise de Contexto de Uso
AT-DEI	Definição de Estilos de Interação
AT-DI	Desenho da Interação
AT-PI	Prototipação de Interface
AT-AU	Avaliação de Usabilidade

Tabela B.1. Legenda de Atividades Técnicas do XPu.

<i>Sigla</i>	<i>Atividade</i>
AG-DRMU	Definição de Requisitos e Metas de Usabilidade
AG-PU	Planejamento de Usabilidade
AG-CU	Controle de Usabilidade

Tabela B.2. Legenda de Atividades Gerenciais do XPu.

<i>Sigla</i>	<i>Atividade</i>
IA-P	Personas
IA-R	Roteiros
IA-HU	Histórias de Usabilidade
IA-RU	Requisitos de Usabilidade
IA-B	<i>Benchmarks</i>
IA-ER	Estimativa de Recursos
IA-CT	Cenários de Teste
IA-RE	Refinamento de Estimativas
IA-T	<i>Templates</i>
IA-IF	Interface Final
IA-PS	Protótipo Simplificado
IA-PI	Protótipos e Interfaces
IA-TU	Testes de Usabilidade

Tabela B.3. Legenda de Insumos e Artefatos do XPu.

<i>Sigla</i>	<i>Atividade</i>
XP-HU	Histórias de Usabilidade
XP-PR	Planejamento do <i>Release</i>
XP-VP	Velocidade do Projeto

Tabela B.4. Legenda de Atividades do XP.

Apêndice C

Glossário

A seguir são apresentadas as definições dos termos em língua inglesa utilizados pelo método XPu:

Stakeholders: é um termo usado para se referir a qualquer pessoa ou entidade que afeta ou é afetada pelas atividades de uma empresa. De maneira mais ampla, compreende todos os envolvidos em um processo, que pode ser de caráter temporário (como um projeto) ou duradouro (como o negócio de uma empresa ou a missão de uma organização).

Feedback: é um procedimento que consiste no provimento de informação à uma pessoa sobre o desempenho, conduta ou eventualidade executada por ela e objetiva reprimir, reorientar e/ou estimular uma ou mais ações determinadas, executadas anteriormente.

Release: é uma porção do software que passou pelos testes de aceitação e tem potencial de entrar em produção, se assim o cliente desejar.

Build: é uma versão do software que foi integrada ao repositório unificado e tem potencial para passar pelos testes de aceitação.

Coach: é um desenvolvedor, membro da equipe, que atua de forma a facilitar a aplicação e o entendimento das práticas ágeis por todos os demais.

Tracker: é um desenvolvedor, membro da equipe, que atua coletando métricas ao longo das iterações, de forma a alimentar os indicadores do projeto.

Refactoring: é o processo de modificar um sistema de software para melhorar a estrutura interna do código sem alterar seu comportamento externo.

Benchmark: é o ato de executar um programa de computador, um conjunto de programas ou outras operações, a fim de avaliar o desempenho relativo de um objeto, normalmente executando uma série de testes padrões e ensaios nele.

Template: é um documento sem conteúdo, com apenas a apresentação visual (apenas cabeçalhos, por exemplo) e instruções sobre onde e qual tipo de conteúdo deve entrar a cada parcela da apresentação. Por exemplo, conteúdos que só podem aparecer no início e conteúdos que só podem aparecer no final.

Pair Programming: todo código de produção é escrito por dois programadores, dividindo o mesmo teclado e o mesmo monitor. O par divide as responsabilidades, as estratégias, os testes e o código escrito, e trocam de papéis em poucos minutos ou sempre que um deles está cansado ou tem uma boa idéia que possa resolver algum problema de implementação em particular.

Planning Poker: é uma técnica de estimativa baseada em consenso, na maioria das vezes usada para estimar esforço ou tamanho relativo de tarefas no desenvolvimento de software.

Anexo A

Relatório de Avaliação Heurística

Avaliação do primeiro membro da equipe

H1 - Visibilidade do estado do sistema

Problema na interface da Figura 5.5:

1 - Não fica claro o estado do sistema, nem qual operador está fazendo uso do sistema, ou se já está no modo de inserção. Deveria haver uma barra de Status com estas informações.

Gravidade: Média.

Problema na interface da Figura 5.11:

1 - Não fica claro também qual o estado atual do sistema, já que não informa se algum usuário já se autenticou no sistema e se for o caso qual usuário foi este. Deveria haver uma barra de Status com estas informações.

Gravidade: Alta.

H2 - Correspondência entre o sistema e o mundo real

Nenhum problema encontrado.

H3 - Controle e liberdade do usuário

Problema na interface da Figura 5.6:

1 - Não existe nenhuma opção de voltar (*redo*) e alterar os itens da locação caso os mesmos estejam incorretos. Deveria existir a opção “Cancelar locação” antes de

imprimir o comprovante.

Gravidade: Média.

Problema na interface da Figura 5.11:

1 - Depois que o usuário se autentica na interface não existe opção do mesmo sair do sistema sem efetuar uma locação, deveria existir a opção “Sair” em contraposição à opção entrar.

Gravidade: Média.

H4 - Consistência e padronização

Problema na interface da Figura 5.4:

1 - O botão “Logar” deveria se chamar “Entrar” já que na tela inicial do sistema de auto-atendimento, o nome do botão utilizado para chamar a autenticação no sistema foi este.

Gravidade: Baixa.

H5 - Prevenção de erro

Nenhum problema encontrado.

H6 - Ajuda aos usuários para reconhecerem, diagnosticarem e se recuperarem de erros

Nenhum problema encontrado.

H7 - Reconhecimento em vez de memorização

Nenhum problema encontrado.

H8 - Flexibilidade e eficiência de uso

Nenhum problema encontrado.

H9 - Design estético e minimalista

Nenhum problema encontrado.

H10 - Ajuda e documentação

Problema nas interfaces das Figuras 5.5, 5.6, 5.4 e 5.11:

1 - Esta interface não possui *tags* com dicas para os usuários nem opções de acesso à ajuda e documentação. Deveria haver atalhos disponíveis à documentação e ajuda do sistema, esclarecendo possíveis dúvidas do usuário.

Gravidade: Média.

Avaliação do segundo membro da equipe

H1 - Visibilidade do estado do sistema

Problema em todas as interfaces:

1 - O sistema não sinaliza o seu estado, não indica o local que o usuário se encontra, como “Terminal -> Login” ou “Terminal -> Comprovante de Locação”, e também não há indicação se o usuário está logado no sistema.

Gravidade: Média.

Solução: Criar uma barra de status contendo essas informações sinalizadas.

H2 - Correspondência entre o sistema e o mundo real

Problema na interface da Figura 5.11:

1 - O botão “Locar Filmes” da tela acima não está muito adequado ao vocabulário dos usuários.

Gravidade: Pequena.

Solução: Alterar “Locar” para “Alugar” no botão.

H3 - Controle e liberdade do usuário

Problema na interface da Figura 5.6:

1 - O sistema não possibilita desfazer a locação na hora de imprimir o comprovante de locação, conforme tela acima.

Gravidade: Média.

Solução: Inserir o botão “Cancelar” e desfazer a locação no banco de dados.

H4 - Consistência e padronização

Nenhum problema encontrado.

H5 - Prevenção de erro

Nenhum problema encontrado.

H6 - Ajuda aos usuários para reconhecerem, diagnosticarem e se recuperarem de erros

Nenhum problema encontrado.

H7 - Reconhecimento em vez de memorização

Nenhum problema encontrado.

H8 - Flexibilidade e eficiência de uso

Nenhum problema encontrado.

H9 - Design estético e minimalista

Nenhum problema encontrado.

H10 - Ajuda e documentação

Problema em todas as interfaces:

1 - O sistema não possui documentação explicando os passos na realização das atividades.

Gravidade: Média.

Solução: Inserir uma ajuda que liste os passos a serem realizados na locação de um filme e cadastro de clientes.