

UNIVERSIDADE FEDERAL DE PERNAMBUCO

GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

CENTRO DE INFORMÁTICA

2012.1

**Ferramenta para apoio à estimativa baseada em
Planning Poker utilizando a metodologia Scrum**

TRABALHO DE GRADUAÇÃO

Dennis Williams Alves da Silveira

Recife

Julho de 2012

UNIVERSIDADE FEDERAL DE PERNAMBUCO

GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

CENTRO DE INFORMÁTICA

2012.1

Ferramenta para apoio à estimativa baseada em Planning Poker utilizando a metodologia Scrum

TRABALHO DE GRADUAÇÃO

Dennis Williams Alves da Silveira

Monografia apresentada junto ao curso de Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, na área de engenharia de software, sendo requisito parcial para a obtenção do título de Bacharel.

Orientador: Alexandre Marcos Lins de Vasconcelos

Co-orientador: Sandro Ronaldo Bezerra Oliveira

Recife

Julho de 2012

“A virtude está toda no esforço.”

(Anatole France)

Agradecimentos

Primeiramente, gostaria de dedicar este trabalho aos meus pais, Fernando e Socorro, pelo esforço e pela dedicação depositada em minha formação e pela forte influência que tiveram em minha vida. Foram eles quem sempre me incentivaram a seguir sempre em frente e nunca cair diante das dificuldades. Tudo que conquistei até hoje eu devo isso a eles.

Um agradecimento muito especial ao meu irmão, Júnior, que sempre esteve junto nos momentos de alegria e tristeza e pelas boas conversas que nós tivemos.

À Quitéria, que sempre ajudou a mim e minha família nos momentos que nós mais precisávamos e é como uma segunda mãe pra mim.

À toda minha família pela grande amizade e por acreditarem em mim. Em especial aos meus primos Bruno, Marcus, Rafa (que também é aluno no CIn), Marina, Geny, Romero, Thiago, Felipe e mais outros tantos primos.

Ao professor Alexandre Marcos Lins de Vasconcelos, pela oportunidade e orientação neste trabalho. E também ao professor Sandro Ronaldo Bezerra Oliveira, do Centro de Tecnologia de Informação e Comunicação da Universidade Federal do Pará, pelas explicações e indicação de materiais valiosos para o desenvolvimento deste trabalho.

Aos amigos que fiz no CIn e em outros centros da UFPE, e ao pessoal que marcava as peladas em Boa Viagem.

Aos professores do CIn pelo ensino de qualidade e a dedicação em passar os conhecimentos durante o curso de graduação, fazendo com que este seja um centro de excelência em computação.

Enfim, a todas as pessoas com quem tive oportunidade de conversar e conhecer, pois sempre aprendemos algo novo com a troca de conhecimentos e experiências.

Resumo

Um projeto consiste de atividades realizadas por uma equipe de trabalho que atua em busca de um objetivo comum. Para conduzir as atividades, existem diversas metodologias que abordam uma prática de gerenciamento de projeto, para avaliar fatores como o cronograma, custos, qualidade e disponibilidade de recursos humanos. Dentro desse processo, pode-se considerar que a atividade de estimar o esforço necessário para o desenvolvimento de um projeto é um elemento fundamental para o sucesso do mesmo. Essa prática é usada tanto em metodologias tradicionais e ágeis.

Dados mensuráveis relevantes desses fatores são oriundos da utilização de técnicas de estimativas sobre a métrica do tamanho de um projeto, e que podem ser as mais diversas, variando desde analogias entre projetos e baseadas na experiência do desenvolvedor, até abordagens mais metódicas utilizando modelos matemáticos.

Uma delas é o *Planning Poker*, que é uma técnica de estimativa ágil. A natureza interativa e objetiva deste método permite que equipes ágeis possam estimar seus requisitos com uma velocidade impressionante e que possam ser bem assimiladas por todos os envolvidos.

Apesar disso, o *Planning Poker* possui limitações, como o fato de não funcionar bem em equipes geograficamente distribuídas, já que a própria essência dela é a interação presencial. Existem ferramentas disponíveis para extrair estimativas nessa técnica, cada uma com características positivas e negativas.

Assim, o objetivo deste trabalho é fazer um estudo das características dessa técnica juntamente com uma abordagem das ferramentas já existentes. A partir daí, a ideia é propor uma ferramenta prática e objetiva para estimar utilizando *Planning Poker*, com foco em equipes Scrum, e que possa ser aplicada para o gerenciamento e acompanhamento da evolução do projeto.

Abstract

A project has activities carried out by a team of people who work for a common goal. There are several methodologies for software development that are used to create a practical approach to project management, to organize these activities and to assess factors such as time, cost, quality and human resources. In this process, the effort estimation activity is a key element for the success of the project. This practice has been used in both traditional and agile methods.

Measurable data are derived from the use of estimation techniques on the project size metric, ranging from analogies between projects based on developer experience to methodical approaches using mathematical models.

One of these techniques is the Planning Poker, which is an agile estimation technique. Due to the interactive and practical nature of it, agile teams can estimate and assimilate task sizes very fast.

Nevertheless, the Planning Poker has some limitations, such as the fact that it doesn't work well in distributed agile teams and the face-to-face interaction is the very essence of it. There are some tools available to estimate by playing it, each one has positive and negative features.

This final work presents a Planning Poker approach. From there, the main idea is to propose a practical tool to estimate using this technique, focused on Scrum teams, and to be applied to the project management and monitoring.

Sumário

1. Introdução.....	12
1.1. Objetivos	13
1.2. Estrutura do Documento.....	14
2. Metodologias Ágeis.....	15
2.1. Conceitos.....	16
2.2. Considerações Finais	19
3. Scrum.....	20
3.1. Ciclo de Desenvolvimento.....	21
3.2. Papéis no Scrum	22
Scrum Master	22
Product Owner	23
Scrum Team.....	23
3.3. Reuniões.....	24
Sprint Planning Meeting.....	24
Daily Scrum Meeting	25
Sprint Review Meeting.....	25
Sprint Retrospective Meeting	25
3.4. Artefatos.....	26
Product Backlog.....	26
Estórias de Usuário.....	26
Impediment Backlog	29
Sprint Backlog.....	29
Burndown Chart	29
3.5. Planejamento da <i>Sprint</i>	30
3.6. Scrum para Equipes Distribuídas.....	31
3.7. Considerações Finais	33
4. Estimativa de Software	34
4.1. Análise de Pontos de Função	35
Determinar o tipo de contagem.....	36
Identificar a fronteira da aplicação	36
Contar as funções de dados	36
Contar as funções transacionais.....	37
Determinar o PF não ajustado	38

Calcular Valor do Fator de Ajuste.....	39
Calcular os Pontos de Função Ajustados.....	39
4.2. Análise de Pontos de Casos de Uso.....	39
Calculando o peso dos Atores do sistema.....	40
Calculando o Peso dos Casos de Uso	40
Cálculo dos Pontos de Casos de Uso não Ajustados	41
Cálculo dos Fatores Técnicos	41
Cálculo dos Fatores Ambientais	42
Cálculo dos Pontos de Caso de Uso ajustados	43
4.3. Planning Poker.....	43
4.4. Considerações Finais	47
5. Ferramentas para Estimativas Usando o Planning Poker	49
5.1. Planning Poker.....	49
5.2. FireScrum	51
5.3. Comparação das Ferramentas	53
5.4. Considerações Finais	54
6. A Ferramenta Proposta	55
6.1. Diagrama de Casos de Uso	56
6.2. Diagrama de Classes.....	60
6.3. Diagrama Entidade-Relacionamento	61
6.4. Arquitetura do Sistema	62
6.5. Protótipo	65
6.6. Considerações Finais	66
7. Conclusão	67
7.1. Trabalhos Futuros.....	68
Referências Bibliográficas	69
Apêndice A – Product Backlog	73
Cadastrar Usuário.....	73
Efetuar Login	74
Efetuar Logout.....	75
Listar Projetos.....	76
Alterar Usuário	77
Cadastrar Projeto	78
Ver Projeto	79

Alterar Projeto.....	80
Manter Sprints	81
Enviar Convites.....	82
Aceitar Convites	83
Ver Participantes de um Projeto	84
Remover Projeto	84
Exibir Product Backlog.....	85
Cadastrar Item.....	86
Ver Item.....	87
Alterar Item	88
Remover Item.....	89
Ver Gráfico de Acompanhamento	90
Iniciar Sessão Planning Poker.....	91
Entrar numa Sessão Planning Poker.....	92
Escolher Carta	93
Mostrar Cartas	94
Definir Estimativa para um Item	95
Realizar Nova Rodada	96
Sair da Sessão.....	97
Finalizar Sessão	98
Chat da Sessão	99
Apêndice B – Backlog de Requisitos Não Funcionais	100
Performance.....	100
Usabilidade.....	100
Disponibilidade.....	100
Manual de Instalação	100
Manual do Usuário.....	101
Escalabilidade	101
Integridade.....	101
Privacidade.....	101
Legibilidade do Código	101

Lista de Figuras

Figura 1. Metodologias ágeis nas empresas.	19
Figura 2. Ciclo de Desenvolvimento do Scrum.....	21
Figura 3. Exemplo de um cartão de estória.	28
Figura 4. Exemplo de testes de um cartão de estória.....	28
Figura 5. Exemplo de um <i>Sprint Backlog</i>	29
Figura 6. Gráfico <i>Burndown</i> da <i>Sprint</i>	30
Figura 7. Baralho de <i>Planning Poker</i> (Fonte: CRISP Konsulter)	44
Figura 8. Coordenador solicita aos participantes a estimativa de uma determinada estória.	45
Figura 9. Integrantes da equipe técnica escolhendo suas estimativas da estória.	46
Figura 10. Integrantes da equipe técnica mostrando suas cartas.	46
Figura 11. Integrantes da equipe técnica refazendo suas escolhas para a estimativa.	47
Figura 12. Tela de descrição de uma estória no <i>Planning Poker</i>	49
Figura 13. Tela para escolha de uma estimativa no <i>Planning Poker</i>	50
Figura 14. Tela para o moderador definir uma estimativa no <i>Planning Poker</i>	50
Figura 15. Tela do módulo de <i>Planning Poker</i> do <i>FireScrum</i>	52
Figura 16. Diagrama de Casos de Uso da ferramenta.	56
Figura 17. Diagrama de Casos de Uso de relacionamento dos atores da ferramenta.....	57
Figura 18. Diagrama de Casos de Uso do ator Usuário.....	57
Figura 19. Diagrama de Casos de Uso do ator Coordenador.	58
Figura 20. Diagrama de Casos de Uso do ator Participante.....	59
Figura 21. Diagrama de Classes da ferramenta.....	60
Figura 22. Diagrama E-R da ferramenta.....	62
Figura 23. Arquitetura da ferramenta.....	63
Figura 24. Camada de apresentação.....	63
Figura 25. Camada de controle.	64
Figura 26. Camada de negócio.	64
Figura 27. Camada de persistência.	65

Lista de Tabelas

Tabela 1. Exemplo de um <i>Product Backlog</i>	26
Tabela 2. Exemplos de estórias em um <i>Product Backlog</i>	27
Tabela 3. Complexidade funcional para os ALI e AIE.	37
Tabela 4. Complexidade funcional para os EE.....	38
Tabela 5. Complexidade funcional para os SE e CE.....	38
Tabela 6. Pesos de atores.....	40
Tabela 7. Pesos de Casos de Uso por número de transações e classes.	40
Tabela 8. Fatores e pesos de complexidade técnica na PCU.	41
Tabela 9. Fatores e pesos de complexidade ambiental na PCU.....	42
Tabela 10. Comparativo entre as ferramentas abordadas.	53

Capítulo 1

Introdução

A atividade de determinar o esforço necessário para o desenvolvimento de um projeto é um elemento fundamental para o sucesso do mesmo. O esforço abrange diversos fatores como o cronograma, o custo exigido para o desenvolvimento de produtos e os recursos humanos do projeto e a capacidade de estimar com precisão esses elementos torna-se vital para uma conclusão efetiva. Dessa forma, pode-se dizer que estimar e planejar estão correlacionados e que são atividades críticas em todas as abordagens de desenvolvimento de *software*, inclusive em metodologias ágeis.

O Scrum, como uma dessas metodologias, é baseada no controle de processos empíricos e emprega uma abordagem iterativa e incremental para melhorar a previsibilidade e controlar riscos. Isso indica que as equipes Scrum não possuem tempo para formular teorias e gerar documentos, tão utilizados nas metodologias tradicionais. O foco está em ter as coisas prontas [Kniberg 2007]. Portanto, para esta abordagem é imprescindível um modelo de estimativas que seja rápido e eficiente.

Existe uma técnica muito usada no Scrum, conhecida como *Planning Poker*, que trabalha para estimar objetivamente. É uma prática que tem sido bem utilizada quando equipes de trabalho se encontram num mesmo ambiente. No entanto, a dificuldade pela falta de contato entre equipes geograficamente distantes pode atrapalhar na realização das estimativas. As práticas ágeis têm a abordagem baseada em interações intensas entre pessoas, portanto é grande a colaboração e a comunicação informal. Infelizmente, uma das maiores dificuldades do desenvolvimento distribuído é o fato de não haver uma coordenação, controle e comunicação constante pela ausência de interações presenciais [Huzita *et al* 2008] e isso acarreta grandes dificuldades ao tentar interligar esses dois conceitos [Highsmith & Cockburn 2001]. Quando se encontram juntas, criam uma nova série de desafios que precisam ser trabalhados.

No entanto, existe um grande interesse em aplicar novas práticas que combinem as características específicas das metodologias ágeis com as de desenvolvimento distribuído de *software* [Šmiteet *al* 2010], [Santos 2010]. Uma dessas práticas é a utilização de ferramentas que contribuam para integrar os dois conceitos, pois elas oferecem um grande suporte de comunicação e reduzem impactos provocados pela falta de interação direta [Santos 2010]. Além disso, elas contribuem para a centralização das informações.

As ferramentas descritas têm a possibilidade de auxiliar projetos ágeis em alguns pontos. A prática do *Planning Poker*, por exemplo, seria facilitada através de uma ferramenta na qual todos os membros da equipe pudessem interagir sobre ela. Na mesma, as estimativas seriam realizadas sobre estórias de usuário (*user stories*) e o jogo é realizado por todos os membros, sob a orientação de um coordenador (ou *Scrum Master*, dentro do Scrum). No caso de precisar debater elementos importantes de uma estória, os participantes podem debater entre si em um espaço disponibilizado na ferramenta para a comunicação entre eles.

Outro ponto, não menos importante, é que as informações do projeto são sempre preenchidas após cada reunião realizada pela equipe de trabalho. E esses dados costumam ser armazenados em ferramentas à parte para o controle interno. Esses documentos são habilitados para que várias pessoas possam editá-lo, pois muitas vezes, desenvolvedores podem abrir o documento para alterar alguma informação válida [Kniberg 2007]. Normalmente essas ferramentas funcionam bem com equipes locais e em pequenos grupos. No entanto, para equipes maiores e geograficamente distantes, o compartilhamento dessas informações pode se tornar um problema maior. Portanto, percebe-se que a ferramenta também atenda a necessidade de centralizar essas informações para trabalhos que envolvam equipes remotas.

1.1. Objetivos

O objetivo deste trabalho é propor uma ferramenta de estimativas baseada em *Planning Poker* para determinar e analisar as estimativas de esforço e custo para um projeto de desenvolvimento de software utilizando a metodologia Scrum, contribuindo para o acompanhamento e gerenciamento efetivo.

A proposta visa que seja possível reduzir os impactos provocados pelo distanciamento geográfico entre membros de equipes de trabalho Scrum. Além disso, a ideia principal é criar uma especificação para a ferramenta a partir do levantamento de funcionalidades. Isso vai servir como base para que outros projetos, que tenham a intenção de estimar com *Planning Poker*, possam desenvolver suas próprias aplicações. Nesse sentido, vai ser necessário um estudo apropriado da técnica de *Planning Poker* e da própria metodologia Scrum, para compreensão e utilização de suas práticas comuns, como por exemplo, a definição de um *Product Backlog* ao invés dos tradicionais requisitos funcionais e não funcionais, visto que o presente trabalho destina-se a equipes Scrum.

1.2. Estrutura do Documento

Além do capítulo de introdução, este trabalho conta com a seguinte estrutura:

O Capítulo 2 vai apresentar conceitos de metodologias ágeis para melhor entendimento de suas práticas.

O Capítulo 3 vai apresentar a metodologia Scrum, por possuir as práticas necessárias para o desenvolvimento da ferramenta em questão. Vão ser abordados conceitos importantes como o ciclo de vida, os papéis, artefatos, reuniões e também vai ser introduzida uma visão de como o Scrum trabalha com equipes remotas.

O Capítulo 4 vai mostrar conceitos sobre estimativas de *software* e sua importância para o processo de desenvolvimento. Será feita um estudo de algumas técnicas já existentes em metodologias tradicionais e também a principal técnica ágil, o *Planning Poker*, objeto de estudo deste trabalho.

O Capítulo 5 apresentará algumas das ferramentas já existentes que trabalham com estimativas em *Planning Poker* e serão apontadas algumas das vantagens e desvantagens.

O Capítulo 6 tratará da proposta de ferramenta para estimar com *Planning Poker*, destacando a estrutura da mesma e alguns diagramas elaborados. Também vai ser explanado o contexto do desenvolvimento do protótipo.

Por fim, o Capítulo 7 será composto da conclusão do trabalho. Nele será feita a conclusão sobre o presente trabalho e como poderá ser aplicado em trabalhos futuros.

Capítulo 2

Metodologias Ágeis

Na era da informação, é visível a necessidade de sistemas de *software* como ferramentas de trabalho para as diversas organizações. A crescente demanda por sistemas de qualidade e que atendam a um mercado competitivo muito grande, fez surgir uma variedade de estratégias para o desenvolvimento de *software*. Estas têm como objetivo de planejar e organizar um projeto e uma equipe de trabalho, aumentando a eficácia e diminuindo os prazos e custos.

Ao longo do tempo, foram apresentadas várias metodologias de desenvolvimento. Dentre elas, existem as metodologias tradicionais, cujo foco é voltado para a documentação e surgiram em um contexto muito diferente do atual, baseado apenas em um mainframe e em terminais burros [Royce 1970]. Devido ao fato de que alterar um sistema era muito custoso, ele era planejado e documentado antes de seu desenvolvimento, permanecendo inalterados ao longo do desenvolvimento. Nestes tipos de metodologias, [Clifton & Dunlap 2003] apontam os maiores problemas encontrados no processo de desenvolvimento de *software*:

1. **Complicações que surgem pela mudança constante de requisitos:** normalmente, os requisitos de um projeto costumam mudar drasticamente desde sua concepção até a sua implantação. Em grande parte dos métodos de produção de *software*, o projeto é feito no início, então não são permitidas alterações quando os requisitos mudam (modelo cascata).
2. **Estimativas de tempo, custo e qualidade do produto que não condizem com a realidade:** apesar do fato de não ser possível prever o cronograma e os recursos despendidos a um projeto, existe uma tendência da equipe de projeto em subestimar estes fatores.
3. **Os desenvolvedores são forçados a não dizer a verdade acerca do andamento do projeto:** quando a gestão subestima o tempo e o custo necessário para atingir certo grau de maturidade, os desenvolvedores normalmente mentem sobre quanto progresso foi feito sobre o projeto. É isso ou enfrentar a indignação do gerente.

Estes pontos aumentam a complexidade e a imprevisibilidade do processo de desenvolvimento de *software*. Um sistema nunca é construído da mesma forma, com a mesma equipe sob as mesmas condições. Uma das mais importantes características do processo na construção de *software* é a mudança constante. É preciso apresentar uma abordagem empírica que vê a mudança como uma situação que irremediavelmente vai acontecer e

aceitá-la ao invés de tentar prever o futuro, e a partir disso saber avaliar e responder a essas mudanças [Soares 2004].

Nesse contexto, as chamadas metodologias ágeis surgiram como alternativa aos métodos de desenvolvimento tradicionais, considerados burocráticos e lentos devido ao foco excessivo na geração de documentos e o cumprimento pontual dos processos. Por outro lado, a proposta ágil concentra as atenções no desenvolvimento e na relação entre os envolvidos [Mundin *et al* 2002].

2.1. Conceitos

A popularidade das metodologias ágeis começou em 2001, quando 17 especialistas em processo de desenvolvimento de *software* estabeleceram princípios utilizados por métodos de natureza similar. O resultado foi a criação do “*The Agile Manifesto*” no qual abordaram o nome Metodologia Ágil e estabeleceu uma união entre diferentes metodologias ágeis [Agile 2012].

As metodologias ágeis possuem seus próprios procedimentos e práticas. No entanto, elas possuem alguns fundamentos em comum. [Soares 2009] cita características como desenvolvimento iterativo e incremental, comunicação e a redução de artefatos, como a documentação extensa.

A abordagem ágil tem como principais conceitos a colaboração e integração entre os membros da equipe. As metodologias ágeis se caracterizam por um gerenciamento de projeto em que um líder da equipe esteja organizando, apoiando, inspecionando e garantindo o bem-estar da equipe de desenvolvimento, ao mesmo tempo em que os resultados do projeto vão atendendo as solicitações do cliente. Segundo [Highsmith & Cockburn 2001], não existe novidade alguma relacionada a aplicação das práticas usadas pelos métodos ágeis. O diferencial está em considerar o fator humano como o principal responsável pelo sucesso do projeto, focando na eficácia e na capacidade de gerenciar. Esta ideologia produz uma nova combinação de valores e princípios que definem uma visão do mundo ágil.

A participação do cliente torna-se fundamental nesse contexto para determinar as tarefas que são realmente importantes para o desenvolvimento do projeto, não desperdiçando os recursos em funcionalidades pouco relevantes. E diferentemente das metodologias tradicionais, nas quais o cliente tem pouco conhecimento sobre o andamento do projeto, nos métodos ágeis existe uma grande interação com os desenvolvedores durante o processo de desenvolvimento do sistema [Pereira *et al* 2007]. Além de contribuir para

aprimorá-lo, poderá facilitar a compreensão de todos os envolvidos no entendimento do negócio.

As metodologias ágeis permitem que a natureza de desenvolvimento seja adaptativa e flexível. Elas são indicadas para cenários onde a mudança de requisitos é constante e os resultados precisam ser entregues ao cliente em curtos espaços de tempo. Por trabalhar com desenvolvimento iterativo e incremental, metodologias ágeis realizam pequenas versões de um produto a cada iteração para validá-lo junto ao cliente. Este, por sua vez, pode exigir modificações nos requisitos, seja pelas variações no mercado ou para atender a necessidade de modificar o negócio e seus processos. A proposta é quebrar o desenvolvimento do projeto em pequenas interações, de modo que, ao final de cada etapa, o cliente possua uma versão que agregue valor ao seu negócio [Dantas 2003]. A integração e o teste contínuo também possibilitam a melhora na qualidade do *software*, devido ao fato de serem realizados continuamente a cada ciclo. Assim, os eventuais problemas são resolvidos constantemente.

Segundo [Ambler 2009] e [Clifton & Dunlap 2003], as bases do Manifesto Ágil são:

1. **Indivíduos e interações ao invés de processos e ferramentas:** os sistemas são desenvolvidos por pessoas que compõem uma equipe. A comunicação entre os mesmos é fundamental para que possam trabalhar juntas. E para auxiliar isso, é que processos e ferramentas são importantes.
2. **Software executável ao invés de documentação:** a documentação é importante para o entendimento do sistema. Mas é mais intuitivo entendê-lo através do funcionamento. E para isso ele deve ser construído.
3. **Colaboração do cliente ao invés de negociação de contratos:** o objetivo é entender as necessidades que um cliente tem. Deve-se resolver um problema que ele normalmente nem sabe descrevê-lo exatamente e que pode mudar ao longo do tempo à medida que vêm o sistema funcional. O contrato deve ser importante apenas para estabelecer responsabilidades e compromissos, mas não substituir a comunicação com o cliente.
4. **Respostas rápidas a mudanças ao invés de seguir planos:** mudanças ocorrem naturalmente num ambiente de desenvolvimento e o sistema em questão precisa produzir uma resposta a essas mudanças. Um plano de *software* deve ser flexível o suficiente para receber as mudanças, senão ele é dispensável.

A agilidade pode ser alcançada através de 12 princípios apresentados pela Aliança Ágil, que são descritos como [Agile 2012]:

1. A maior prioridade é satisfazer ao cliente desde o início por meio de entrega contínua de *software* o mais cedo possível.
2. As modificações de requisitos são bem-vindas. Mesmo que tardias, as modificações devem ser aproveitadas como vantagens para a competitividade do cliente.
3. A entrega de *software* funcionando deve ser frequente, a cada duas semanas até dois meses, de preferência no menor espaço de tempo.
4. O pessoal de negócio e os desenvolvedores devem trabalhar juntos diariamente durante todo o projeto.
5. Os projetos devem ser construídos em torno de pessoas motivadas.
6. Encontros presenciais é o modo mais eficiente de levar informação para dentro da equipe.
7. A entrega de *software* funcional deve ser considerada a principal medida de progresso.
8. Os processos ágeis promovem o desenvolvimento sustentável, buscando um ritmo constante, indefinidamente.
9. A atenção a excelência técnica e ao bom projeto deve ser contínua.
10. A busca pela simplicidade é essencial, maximizando a não realização de trabalho que não seja útil.
11. Os melhores trabalhos surgem de equipes auto-organizadas.
12. Em intervalos regulares, a equipe deve refletir regularmente sobre como se tornar mais efetiva, ajustando adequadamente seu comportamento.

Existem algumas abordagens ágeis de processo utilizadas no mercado, que seguem estes conceitos. Dentre as mais conhecidas, podem ser citados: XP (*Extreme Programming*), Scrum, FDD (*Feature Driven Development*), DSDM (*Dynamic Systems Development Method*), Lean Development e Crystal. A Figura 1 mostra a relação das metodologias ágeis mais utilizadas pelas empresas:

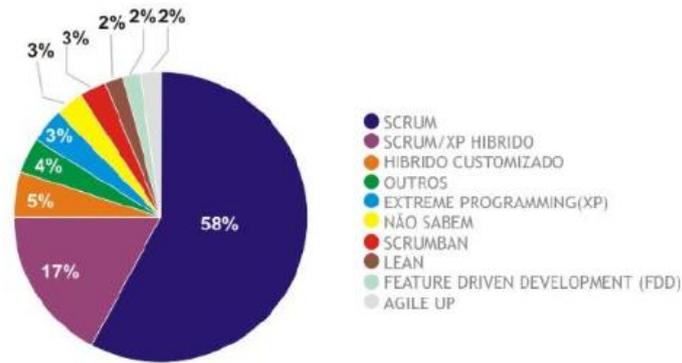


Figura 1. Metodologias ágeis nas empresas. Fonte: <http://www.versionone.com> (2011)

Como visto, a metodologia Scrum é a que está sendo mais difundida, assim como a sua variante a Scrum/XP, que se trata de uma versão híbrida. O presente trabalho de graduação tem como objeto de análise os conceitos utilizados por esta metodologia.

2.2. Considerações Finais

Este capítulo abordou uma breve introdução sobre as metodologias ágeis, revelando suas origens e apontando características que são comuns a todas as práticas, listando algumas de suas bases e princípios. Partindo do conhecimento adquirido aqui, torna-se mais fácil compreender os fundamentos da abordagem Scrum, que será o assunto do próximo capítulo.

Capítulo 3

Scrum

O Scrum, cujo nome se origina de uma formação clássica de uma partida de rúgbi, é um modelo ágil de processo que foi desenvolvido por Jeff Sutherland, Ken Schwaber e sua equipe no início da década de 1990 [Schwaber 2004]. Sua criação foi baseada num artigo que introduz as 10 melhores práticas em empresas, escrito pelos japoneses Hirotaka *Takeuchi* e Ikujiro Nonaka, cujo título é "O jogo do desenvolvimento de novos produtos" que foi publicado na *Harvard Business Review* em janeiro de 1986. Na concepção original, o Scrum foi projetado para ser utilizado por equipes de produção de consumo. Porém, pode ser aplicada em qualquer empresa que trabalhe com processos de *software* [Schwaber 1995].

[Zanatta 2004] considera que o Scrum é um método para gerenciar o processo de desenvolvimento de *software* definindo como as equipes devem trabalhar em ambientes onde requisitos mudam constantemente, sofrendo alterações constantes. Em contrapartida ao que ocorre normalmente em uma indústria de manufatura qualquer, onde os processos são repetíveis e muitas vezes bem definidos.

Segundo [Schwaber 2004], o Scrum não é um processo previsível. Ele é usado em trabalhos complexos nos quais não é possível prever tudo o que irá ocorrer e oferece um *framework* com um conjunto de práticas que torna tudo visível. Assim, os praticantes do Scrum acompanham o que acontece e podem fazer os devidos ajustes para dar sequência ao projeto. O Scrum é adequado para identificar problemas com mais facilidade, mas sua proposta não vai dizer quais são os problemas ou como resolvê-los. Ele servirá como um guia de boas práticas para alcançar o sucesso. Entretanto, as decisões de quando e como usar, quais táticas e estratégias seguir para obter produtividade e realizar as entregas fica por conta de quem aplicar. O conhecimento das suas práticas permite a aplicação das mesmas de forma variada, fazendo com que a adaptabilidade seja um dos aspectos positivos do Scrum.

O Scrum possui seis características básicas: flexibilidade dos resultados, flexibilidade dos prazos, times pequenos, revisões frequentes, colaboração e orientação a objetos [Schwaber 1995]. É uma metodologia que não requer ou fornece qualquer técnica específica para a fase de desenvolvimento, apenas estabelece conjuntos de regras e práticas gerenciais que devem ser adotadas para o sucesso de um projeto [Carvalho & Mello 2009].

3.1. Ciclo de Desenvolvimento

O Scrum é composto basicamente por três fases durante o desenvolvimento do *software*. Nesse processo está incluída a criação do *Product Backlog*, Planejamento da *Sprint* e a entrega de uma versão. As fases do ciclo de vida são abordadas a seguir.

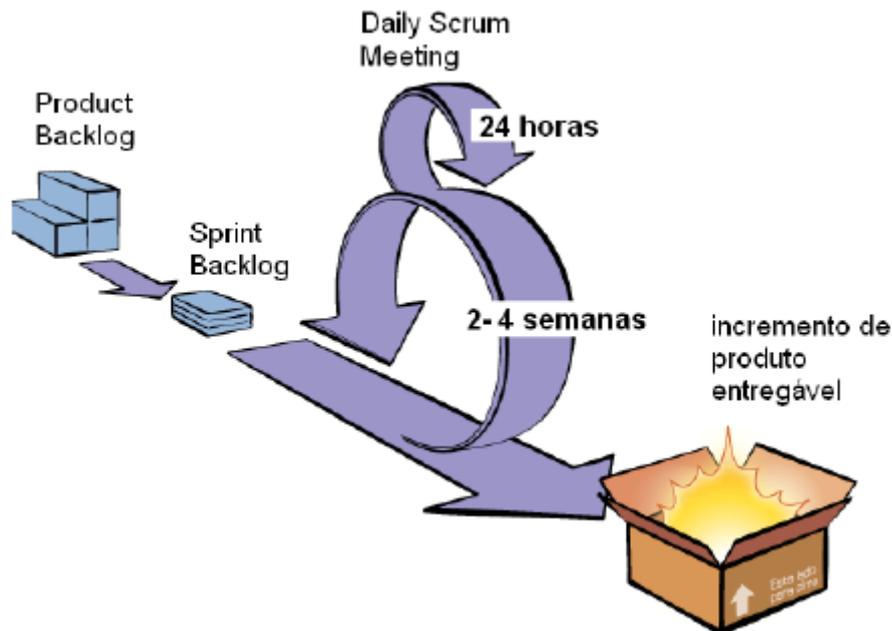


Figura 2. Ciclo de Desenvolvimento do Scrum

A primeira fase do Scrum é o pré-planejamento (*pre-game*), na qual um projeto se inicia com uma visão de alto nível do produto que será desenvolvido, normalmente muito abstrata a princípio e que vai se tornando mais esclarecida a medida que vai sendo desenvolvido o produto [Clifton & Dunlap 2003], [Schwaber 1995]. O responsável pelo projeto (*Product Owner*) é quem vai transformar os elementos dessa visão em uma lista de requisitos que reflitam as funcionalidades do projeto. Essa lista é chamada de *Product Backlog*, é priorizada pelo *Product Owner* de forma que os itens que gerem maior valor ao produto tenham maior prioridade. Inicialmente deve-se dividir o *Product Backlog* em *releases* e é esperado que o conteúdo, a prioridade e o conjunto sofram mudanças, que serão refletidas nas regras e requisitos de negócios a partir do momento que o projeto começa [Pereira *et al* 2007].

O planejamento inclui também, entre outras atividades, a definição da equipe de desenvolvimento, identificação de possíveis alterações no *Product Backlog*, avaliação e controle dos riscos e uma proposta de arquitetura de desenvolvimento [Schwaber 1995].

A próxima fase é a de desenvolvimento (*mid-game*), na qual ocorre execução da *Sprint* e a equipe determina como as tarefas devem ser executadas. O ciclo do Scrum tem o seu

progresso baseado em *Sprints*, que são iterações bem definidas, cada uma com durações de 2 a 4 semanas. Cada uma delas é iniciada com uma Reunião de Planejamento (*Sprint Planning Meeting*) onde a equipe de desenvolvimento entra em contato com o *Product Owner* para priorizar o trabalho que precisa ser feito e estimar o esforço para o desenvolvimento de cada tarefa [Pereira *et al* 2007].

Todos os dias são realizadas reuniões rápidas (*Daily Meeting*) e não devem passar mais do que 15 minutos. Qualquer problema encontrado durante essas reuniões deve ser tratado em outra reunião, apenas com os envolvidos. Durante a execução da *Sprint*, a alocação de recursos para cada tarefa é realizada pela própria equipe. Cada membro seleciona as tarefas que podem realizar e o grupo estabelece a ordem e dependência entre elas. Todos os participantes devem reportar o tempo gasto em tarefas para que o valor de horas restantes seja calculado corretamente e a equipe possa verificar o seu progresso. Para o acompanhamento, é usado um gráfico chamado de *Sprint Burndown*, que mostra o progresso diário em função do total de horas estabelecido pela soma de horas das tarefas dos itens do *Product Backlog* selecionados [Pereira *et al* 2007].

Ao final da *Sprint*, deve ser realizada uma Reunião de Revisão (*Sprint Review Meeting*), na qual a equipe de desenvolvimento apresenta ao *Product Owner* o que foi desenvolvido durante a *Sprint* e este avalia se o objetivo foi atingido. Logo em seguida, é feita a Reunião de Retrospectiva (*Sprint Retrospective*) [Pereira *et al* 2007].

O ciclo do Scrum é repetido até que todos os itens do *Product Backlog* tenham sido finalizados e/ou o produto final tenha sido validado pelo cliente, sendo esta a fase de pós-planejamento (*post-game*) [Clifton & Dunlap 2003].

3.2. Papéis no Scrum

[Schwaber 2004] afirma que a estrutura de um projeto desenvolvido através do Scrum é de responsabilidade de três papéis fundamentais: o *Scrum Master*, o *Product Owner* e o *Scrum Team*.

Scrum Master

O *Scrum Master* atua como um líder que gerencia os interesses do *Product Owner*. É o responsável pelo sucesso do projeto e por garantir que todos os envolvidos estejam aderindo aos valores e regras do Scrum. Atua como um facilitador, ajudando a equipe e o *Product Owner* em todos os processos envolvidos e estimulando a criatividade e o conhecimento da

equipe, a fim de maximizar os resultados do projeto [Yoshima 2007]. O Scrum Master também é o responsável por atualizar o *Sprint Burndown* durante a execução de uma *Sprint*.

O papel do *Scrum Master* é fundamental, sendo durante a fase inicial de implantação uma função extremamente desafiadora. A função de evitar impedimentos na equipe e de proteger o time de interferências externas ou na própria empresa faz com que seja revisto até o processo organizacional da mesma [Varaschim 2009].

Não é necessário um conhecimento técnico por parte do *Scrum Master*, pois sua principal função é compreender as dificuldades da equipe e garantir a boa comunicação.

Product Owner

O *Product Owner* é o responsável pelo gerenciamento do *Product Backlog*, mantendo-o atualizado e visível a todos os membros da equipe. Responsável pela definição das prioridades das estórias dos itens a serem desenvolvidos junto à equipe e também por garantir o retorno de investimento do trabalho realizado [Yoshima 2007]. É importante salientar que sugestões de estórias podem ser feitas por qualquer pessoa, mas a inclusão destas no *Product Backlog* é de gerência do *Product Owner* [Varaschim 2009].

Para [Reis 2010], o *Product Owner* é quem faz a ligação entre o cliente e a equipe, sob a perspectiva de negócios. Ele deve avaliar a entrega de seu produto para os clientes e rever funcionalidades. O *Product Owner* tem o poder para interromper uma *Sprint* em caso de urgência e até mesmo de suspender um projeto caso detecte que seu retorno não será o esperado pela empresa. Portanto, ele deve inteirar-se sobre o impacto técnico e organizacional de suas decisões [Varaschim 2009].

O perfil de um *Product Owner* está relacionado à área de produto, sendo necessário que o mesmo consiga compartilhar os objetivos do produto que está sendo construído pela equipe. Ele deve ser visto como um líder pela equipe deve ser o principal motivador e ter bom relacionamento com a equipe [Varaschim 2009].

Scrum Team

A equipe (*Scrum Team*) é a responsável pelo desenvolvimento dos itens que compõem o produto, de acordo com as prioridades definidas pelo *Product Owner* no *Product Backlog*. O conhecimento da equipe deve ser englobado para que possa realizar a implementação do trabalho [Varaschim 2009].

Além disso, a equipe tem o dever de verificar se suas práticas de desenvolvimento são compatíveis com as necessidades de qualidade e disponibilidade da aplicação. Se a equipe

necessitar de refazer o seu código ou elaborar testes automatizados, ela deve alterar estórias e demonstrar esta necessidade ao *Product Owner* para que as mesmas sejam priorizadas e desenvolvidas pela equipe. Ela deve ser autônoma e ter controle sobre o seu processo de desenvolvimento, sendo de sua responsabilidade ao final de cada *Sprint* mostrar os resultados do trabalho para o cliente [Varaschim 2009].

Para [Libardi & Barbosa 2010], as equipes Scrum são compostas de seis a dez pessoas autogerenciáveis e multifuncionais. Todos trabalham em conjunto para entregar um produto completo e confiável. Sua composição pode mudar ao final da *Sprint*, mas há de se levar em consideração que haverá uma perda de rendimento inicial caso ocorra.

3.3. Reuniões

Sprint Planning Meeting

De acordo com [Libardi & Barbosa 2010], A *Sprint Planning Meeting* é uma reunião realizada como primeira atividade de uma *Sprint*. Nela, estão presentes o *Product Owner*, o *Scrum Master* e o *Scrum Team* e são realizadas as seleções das estórias que serão implementadas durante aquele ciclo. Durante esta reunião o *Product Owner* descreve as funcionalidades de maior prioridade para a equipe. Esta reunião leva cerca de 8 horas.

Nas primeiras 4 horas, o *Product Owner* apresenta os itens por prioridade do *Product Backlog* para a equipe. Esta, por sua vez, faz perguntas para compreender as funcionalidades e ser capaz de dividir em tarefas técnicas, que irão dar origem ao *Sprint Backlog* e então, o *Product Owner* decide o que poderá entrar no desenvolvimento do próximo *Sprint*. Para isso, ele deve considerar o tamanho e produtividade da equipe e a quantidade de horas disponíveis [Libardi & Barbosa 2010].

Nas próximas 4 horas, a equipe planeja seu trabalho definindo o *Sprint Backlog*. A equipe deve realizar um detalhamento de todas as necessidades para desenvolver uma estória e verificar se a estimativa inicial dada está de acordo com todas as atividades existentes. Para isto é necessário que eles façam o seguinte [Varaschim 2009]:

1. Para cada item, o time descreve as atividades e estima o tempo necessário para o desenvolvimento de cada uma delas. As tarefas devem demandar até um dia de trabalho para que possam ser acompanhadas no *Daily Meeting*. Este processo pode ser definido pela empresa.

2. Ao finalizar as estimativas é verificado se o tempo do *Sprint* é suficiente para a realização das tarefas. Caso contrário, é preciso comunicar ao *Product Owner* para rever as histórias que compõem o *Sprint*.

Daily Scrum Meeting

A cada dia do *Sprint*, o *Scrum Master* realiza uma reunião de 15 minutos com a equipe. Essa reunião, chamada *Daily Scrum Meeting*, tem como objetivo discutir sobre o estado do projeto, o que pode ser feito até a próxima reunião. [Libardi & Barbosa 2010] afirma que no caso de identificar dificuldades e obstáculos, eles devem ser tratados pelo *Scrum Master* o mais rápido possível.

As *Daily Scrum Meetings* devem ser realizadas no mesmo lugar e na mesma hora do dia. Idealmente são feitas pela manhã, para ajudar a estabelecer as prioridades do novo dia de trabalho. Embora qualquer pessoa possa participar da reunião, somente os membros do *Scrum Team* estão autorizados a falar [Libardi & Barbosa 2010].

Essas reuniões têm o intuito auxiliar o *Scrum Master* na melhoria a comunicação com os membros da equipe, remover impedimentos para que as metas sejam alcançadas, além de promover e melhorar o nível do conhecimento acerca do projeto.

Sprint Review Meeting

No final do *Sprint*, a *Sprint Review Meeting* é realizada. Essa reunião é planejada para ser realizada em no máximo 4 horas [Libardi & Barbosa 2010], [DIVUS 2012]. Nesta reunião o *Scrum Team* mostra o que foi desenvolvido durante o *Sprint*, discute o que não foi concluído e o que será feito nas próximas *Sprints*.

Durante a *Sprint Review Meeting*, o projeto é avaliado em relação aos objetivos do *Sprint*. Nesta reunião, a equipe discute como ela foi conduzida, aponta as dificuldades e os itens realizados. O resultado da *Sprint* é apresentado ao *Product Owner*, que estabelece a situação do *Product Backlog* e faz as projeções de datas de entrega dos próximos itens [Libardi & Barbosa 2010], [DIVUS 2012].

No caso de histórias não finalizadas a equipe deve discutir os motivos com o *Product Owner*, para se levantar um balanço dos problemas [Libardi & Barbosa 2010].

Sprint Retrospective Meeting

A *Sprint Retrospective Meeting* é uma reunião de 3 horas que ocorre ao final de um *Sprint*, logo após a *Sprint Review Meeting* e serve para identificar o que funcionou bem, o que

pode ser melhorado e que ações podem ser tomadas. Este é o momento em que o *Scrum Master* tem o trabalho de ajudar o time a encontrar seus verdadeiros problemas [DIVUS 2012].

3.4. Artefatos

Product Backlog

O *Product Backlog* é uma lista contendo todas as funcionalidades desejadas para um produto. O conteúdo desta lista é definido e priorizado pelo *Product Owner*. Oficialmente ele é o dono do documento, mas outras pessoas podem abri-lo, como algum componente do *Scrum Team* que deseja esclarecer ou atualizar algo [Kniberg 2007]. Normalmente as equipes Scrum organizam as informações dos itens de um *Product Backlog* em um documento do Excel, com disponibilidade para compartilhamento. Para [Libardi & Barbosa 2010], o *Product Backlog* é bem dinâmico, sempre modificado quando se identifica algo que o produto precisa para ser mais adequado ao negócio.

Os itens do *Product Backlog* possuem os atributos de descrição, estimativa e prioridade. Eles são definidos pelo *Product Owner*, e apresentados à equipe no *Sprint Planning Meeting*. Esta, por sua vez, determina quais funcionalidades do sistema são possíveis de realizar em uma *Sprint*, sendo estas transferidas para o *Sprint Backlog*. Os itens são divididos em várias tarefas para ser distribuído entre as pessoas da equipe [Reis 2010].

[Schwaber 2004] afirma que este documento costuma sofrer alterações, como mudanças nos requisitos do negócio, custos, alterações no mercado externo, tecnologia entre outros; devendo ser atualizado.

Tabela 1. Exemplo de um *Product Backlog*

Backlog item	Estimativa
Como um visitante do site eu gostaria de mandar um e-mail de contato	8
Como um cliente eu gostaria de buscar por um produto	13
Como um cliente eu gostaria de colocar um produto no carrinho de compras	13
Como um cliente eu gostaria de pagar a compra com o cartão de crédito	40

Estórias de Usuário

O *Product Backlog* é composto por itens que possuem descrições próprias. Eles são denominados de estórias de usuário (*user stories*), que comumente contêm os seguintes campos [Kniberg 2007]:

- **ID:** uma identificação única, normalmente um número com autoincremento, com o objetivo de evitar a perda do controle sobre as estórias, caso estas mudem os nomes.
- **Nome:** um nome curto e descritivo para a estória, como: “Ver o histórico de transações”. Deve ser objetivo para que os desenvolvedores e o *Product Owner* compreendam o contexto, além de distingui-la de outras estórias.
- **Prioridade:** a pontuação de importância dessa estória para o *Product Owner*.
- **Estimativa:** a estimativa inicial da equipe sobre o esforço necessário para implementar uma estória, se comparada a outras. A unidade é pontos por estória (*Story Points*) e geralmente corresponde mais ou menos a “relação homem/dias” ideal.
- **Descrição:** uma descrição em alto nível de como a estória será demonstrada na apresentação do *Sprint*. As descrições podem seguir um padrão, que normalmente é feito da seguinte forma. “como um <sujeito> eu gostaria de <ação>”.
- **Notas:** quaisquer outras informações adicionais.

Tabela 2. Exemplos de estórias em um *Product Backlog*.

ID	Nome	Prior.	Est.	Descrição	Notas
1	Depósito	30	5	Como um cliente eu gostaria de abrir a página de depósito, depositar R\$ 10,00, ir para a página do meu saldo e verificar que este aumentou em R\$ 10,00.	Não é necessário se preocupar com criptografia por enquanto.
2	Verificar histórico de transações	10	8	Como um cliente eu gostaria de fazer um depósito e verificar se o novo depósito é listado.	Usar paginação para evitar consultas muito grandes ao banco de dados. Projetar de forma similar à página de visualização de usuários.

Estórias podem ser representadas por cartões. Essa é uma técnica de captura de requisitos de sistemas e surgiu junto com XP (extreme programming), mas ela pode ser utilizada em qualquer metodologia de desenvolvimento de sistemas ágeis. Esses cartões se concentram em informações do tipo “quem, o quê e porquê de um recurso”, e não “como vai ser feito” [Abu 2010].

O cartão de história é feito a partir da visão do usuário. Assim, ele não possui detalhes técnicos do desenvolvimento, mas apenas informações das regras de negócio, para que possa ser compreendido por todos [Abu 2010].

A Figura 3 apresenta um exemplo de cartão de estória. Além de alguns dos campos descritos anteriormente, o cartão pode conter um esboço de como deve funcionar a tela e anotações do comportamento esperado.

#00001	Consulta de Livros	Tamanho: 3
Como um cliente, quero consultar os livros, para que eu possa encontrar o produto que desejo comprar.		

Consulta de Livros			
Titulo	ISBN	categoria	autor
Busca: <input style="width: 80%;" type="text"/> <input style="width: 15%; border: none; border-bottom: 1px solid black; cursor: pointer;" type="button" value="Buscar..."/>			
Titulo	ISBN	categoria	autor
Ruby On Rails	123456789	Técnico	Jose da Silva
Java	234567890	Técnico	Maria da Silva

1 Define o tipo de consulta a ser realizado

3 Mostra o resultado da consulta

2 Executa a consulta

2 e 3 Mostrar uma mensagem de nenhum produto encontrado para busca vazia

2 Busca é realizada com partes da informação

Figura 3. Exemplo de um cartão de estória.

Confirmações (Testes) são os casos de testes no verso do cartão. Os testes facilitam o entendimento do cartão, ajuda na visualização dos cenários por parte dos envolvidos [Abu 2010]. A Figura 4 mostra as informações dos testes, com os possíveis cenários.

Comportamento de Sucesso
Entra com parte do nome do livro – Pressiona Buscar – Mostra os livros encontrados
Entra com o nome completo do livro – Pressiona Buscar – Mostra o livro encontrado
Entra com o ISBN – Pressiona Buscar – Mostra o livro encontrado
Entra com parte do nome do autor – Pressiona Buscar – Mostra os livros encontrados
Entra com o nome completo do autor – Pressiona Buscar – Mostra o livro encontrado
Comportamento de Erros
Entra com parte do ISBN – Pressiona Buscar – Mostra mensagem de erro
Resultado da busca vazio – Mostra mensagem de erro
Campo de Busca vazio – Pressiona Buscar – Mostra mensagem de erro

Figura 4. Exemplo de testes de um cartão de estória.

Impediment Backlog

Segundo [Schwaber 2004], o *Impediment Backlog* contém todos os itens que impedem o progresso do projeto e que estão associados a riscos. Eles não possuem uma priorização, mas estão vinculados a um *Product Backlog* ou a tarefas dos itens. Dada a importância desses itens, o *Scrum Master* deve ser o responsável pela eliminação dos impedimentos, para que a equipe de desenvolvimento possa executar as tarefas sem maiores problemas.

Sprint Backlog

O *Sprint Backlog* é uma lista que contém as tarefas que a equipe vai trabalhar durante a *Sprint*. Essas tarefas surgem dos itens do *Product Backlog*. A equipe escolhe estes itens analisando o grau de dificuldade, a prioridade e o tempo que será necessário. Uma vez definido, o *Sprint Backlog* não pode ser alterado.

Para [Reis 2010], a elaboração do *Sprint Backlog* pode ser feita através de uma planilha Excel, da mesma forma que o *Product Backlog*, embora existam sistemas apropriados para o gerenciamento em Scrum que inclusive suportem *Sprint Backlog*.

O *Sprint Backlog* costuma ser representado pelo *Sprint Burndown Chart*, que determina a quantidade restante do trabalho ao longo do tempo. Ao *Scrum Master* cabe a responsabilidade de acompanhar, atualizar e analisar da situação das tarefas deste artefato.

Tarefas	Responsável	Estimativa	Status	Dia 1	Dia 2	Dia 3	Dia 4
Criar serviço de email	Paula	5	Complete	5	2	0	0
Implementar envio de email de contato	Marcos	2	In progress	2	2	1	0
Testar o envio de e-mail de contato	Mirian	1	Not Started	1	1	0	0
Criar classe de produto		2	Not Started	2	2	2	2
Criar serviço de busca		4	Not Started	4	4	4	4
...							

Figura 5. Exemplo de um *Sprint Backlog*.

Burndown Chart

O monitoramento do progresso do projeto é realizado através de dois gráficos principais: *Product Burndown Chart* e *Sprint Burndown Chart*. Estes gráficos refletem o progresso da equipe de projeto ao longo do tempo de acordo com a quantidade de trabalho restante [Cohn 2005] e essa informação pode ser mensurada em quantidade de horas para conclusão do projeto, como mostra a Figura 6, ou por pontos de estimativa. A atualização do *Burndown* deve ser diária, pois facilita a tomada de decisão para melhorar a produtividade da equipe.

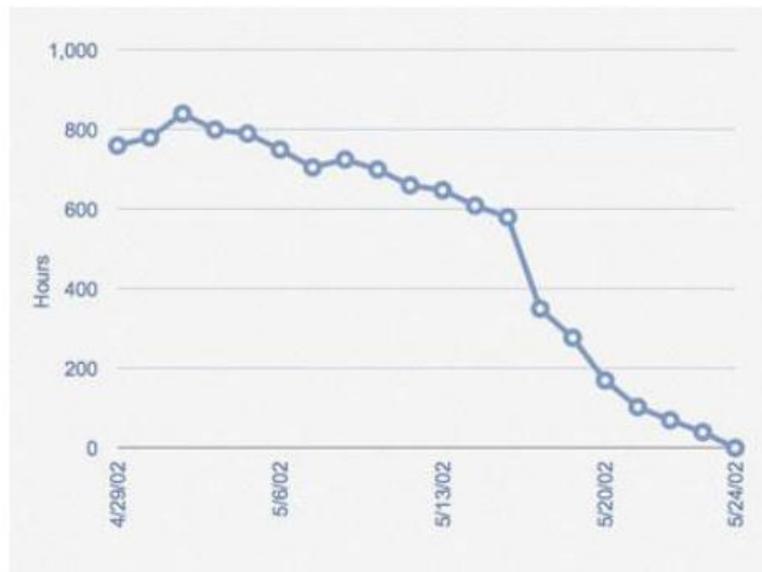


Figura 6. Gráfico *Burndown* da *Sprint*. Fonte: <http://epf.eclipse.org/wikis/scrumpt/Scrum/>

3.5. Planejamento da *Sprint*

A primeira atividade para iniciar uma *Sprint* é a Reunião de Planejamento (*Sprint Planning Meeting*). Essa atividade é muito importante e por isso necessita de preparação prévia. No planejamento da *Sprint*, o *Product Backlog* deve estar pronto antes de cada reunião. Isso significa que é necessário estabelecer os seguintes pontos [Pereira *et al* 2007]:

- Existência do *Product Backlog* e que cada item esteja estimado.
- Deve existir apenas um *Product Backlog* e um *Product Owner*.
- Todos os itens devem ter uma escala em função de sua importância para organizá-los.
- O *Product Owner* deve entender todos os itens do *Product Backlog*, mas em alguns casos outras pessoas podem ter colocado itens nele;
- Ele não precisa saber como cada item deverá ser feito, mas precisa saber o motivo de estar lá.
- Apenas o *Product Owner* pode atribuir a nota de importância aos itens do *Product Backlog*.

Tendo em vista que os itens acima foram atendidos, pode-se iniciar a reunião de planejamento. Ela tem a finalidade de priorizar os itens que serão executados na *Sprint*, além de dar informação suficiente à equipe para que possa validar e estimar o esforço em horas para cada item. Esta reunião é muito crítica para o sucesso do projeto e que o mal

planejamento pode afetar o andamento da *Sprint* e causar impactos ao cronograma do projeto [Schwaber 2004].

Com o *Product Backlog* priorizado, a equipe seleciona os itens podem ser executados durante a *Sprint*. As dúvidas do time são esclarecidas e ao final temos então o *Sprint Backlog*. Para cada item, a equipe inicia o detalhamento de suas atividades, estimando em horas, a duração de cada uma delas. Uma vez que todas as tarefas foram estimadas, a equipe verifica se consegue assumir o compromisso de realizar as tarefas dentro da *Sprint* [Schwaber 2004].

Esse processo continua até que todos os itens do *Sprint Backlog* sejam validados. Após a estimativa refinada, é possível calcular e obter o total de horas necessário para realização das tarefas [Pereira *et al* 2009].

O próximo passo é iniciar a execução da *Sprint*. Ela possui um limite de horas disponível, que é conhecido por LHS (Limite de Horas da *Sprint*). O valor deste limite é possível ser medido utilizando uma fórmula simples [Pereira *et al* 2009]:

$$\text{LHS} = (\text{R} \times \text{H}) \times \text{D}$$

Onde, R é o total de recursos (membros) da equipe, H é o total de horas disponíveis para cada recurso e D é o total de dias úteis da *Sprint*.

É importante considerar na fórmula que o valor de H costuma variar para diferentes recursos. Além disso, é deve-se avaliar apenas o tempo produtivo para cada *Sprint*. Por exemplo: considerar 6 horas efetivas para um recurso que normalmente trabalha 8 horas diárias. Esta sobra é importante para dar uma ideia mais realista da produtividade de cada recurso e também garante uma margem de segurança para imprevistos [Pereira *et al* 2009].

Esses são cálculos bem simples e é preciso usá-los antes de cada planejamento da *Sprint*, pois o LHS é o que vai indicar se as horas estão sendo alocadas corretamente para a equipe [Schwaber 2004].

3.6. Scrum para Equipes Distribuídas

O Scrum utiliza um enfoque de trabalho com um conjunto de pessoas compartilhando o mesmo ambiente, pois facilita a comunicação frequente, além de aumentara interação da equipe [Kniberg 2007]. No entanto, a abordagem do Scrum deve sofrer adaptações quando se trata de equipes geograficamente distantes, devido a uma maior dificuldade em conseguir realizar reuniões diárias.

De acordo com [Sutherland 2007], existem três tipos de equipes distribuídas com Scrum:

- Equipes Isoladas: Há uma independência dentre os membros geograficamente separados, sendo que, alguns integrantes trabalham de maneira distribuída e cada equipe possui seu próprio *Product Owner*.
- *Scrum* Distribuído de *Scrums*: Cada local possui uma equipe multidisciplinar e as equipes realizam reuniões de *Scrum of Scrums* (reunião de várias equipes *Scrums*) periódicas. Um *Product Owner* coordena o trabalho de múltiplas equipes.
- *Scrums* Integrados: Cada equipe possui integrantes em diversos locais. As reuniões ajudam a romper as barreiras culturais e as disparidades de estilos de trabalho.

A estratégia mais comum para diminuir os impactos causados pelo inconveniente da distância é a utilização de ferramentas para aumentar a comunicação entre os membros separados. Normalmente, se utilizam canais de comunicação para isso, como videoconferência, audioconferência, salas de chat ou software de compartilhamento de desktop.

Muitos são os motivos para que existam equipes distribuídas: redução dos custos, ganhos de escala, acesso aos recursos especializados, acesso a novos mercados e outros [Santos 2010]. Sabe-se que equipes distribuídas por atuar em diferentes culturas e pontos de vista, ampliam horizontes e garantem novas experiências, o que contribui para o aprendizado e a criatividade geral. No entanto, é importante ressaltar que esse processo também acarreta em dificuldades, pois é preciso tratar com cuidado questões como: barreiras naturais da língua, tradições, crenças e princípios diferentes [Lima & Reis 2008].

Outra dificuldade que normalmente acontece se refere ao controle e planejamento de projetos que envolvem equipes geograficamente separadas. Neste caso, existem algumas estratégias para solucionar o problema. Alguns trabalhos expõem que a alocação de membros temporariamente pode facilitar a comunicação e a sincronização das atividades [Huzita et al 2008], [Bavani 2009]. Outra estratégia inclui a atribuição de um local físico para definir os processos a serem aplicados no projeto [Bavani 2009]. Por fim, a solução mais comum é a utilização de um sistema de gerenciamento de projetos como ferramenta de apoio às boas práticas de gestão, incluindo um repositório para os artefatos do projeto e ferramentas que facilitem a comunicação instantânea [Santos 2010], [Woodward et al 2010].

3.7. Considerações Finais

Este capítulo abordou os conceitos elementares sobre o Scrum, apontando um breve histórico, as principais características e os fundamentos mais importantes como todo o ciclo de desenvolvimento, os principais papéis, artefatos produzidos, reuniões realizadas, o planejamento de uma *Sprint*.

Também foi realizada uma breve análise de como equipes distribuídas podem trabalhar em projetos Scrum, que como visto, possui sérias diferenças com as práticas de um projeto distribuído, como a falta de comunicação e de controle. É necessário que soluções sejam estabelecidas para reduzir os impactos de se trabalhar com as duas abordagens. Uma das soluções apontadas é o desenvolvimento de ferramentas que facilitem o controle e gerenciamento de projetos, mas de uma maneira que seja estabelecida a comunicação entre os componentes da equipe.

O conhecimento do Scrum será útil para entender melhor a aplicação da técnica de estimativa *Planning Poker*, que é um dos temas deste trabalho de graduação e cujo conteúdo será visto no próximo capítulo, que também abordará alguns dos principais modelos para estimativas de projetos de software.

Capítulo 4

Estimativa de Software

As empresas de TI têm a produção de projetos de *software* com qualidade como um dos maiores entraves a ser enfrentados, a partir de um planejamento determinado. Diversos trabalhos apontam que a falta de uma estimativa precisa dos processos para o desenvolvimento de *software* tem sido uma das maiores causas de problemas enfrentados. Quanto maior a precisão necessária para estimar, maior será a complexidade para obtê-la.

De acordo com [DeMarco 1991], a necessidade de facilitar o planejamento e o controle um projeto de *software*, com o objetivo de torna-lo o mais correto possível, faz da atividade de estimar uma das tarefas mais críticas e de fundamental importância dentro do ciclo de vida de desenvolvimento do projeto. Para [Florac *et al* 1997], a medição de produtos e processos, se feitas da maneira correta, podem fornecer um apoio efetivo para a iniciação e gerência de atividades de melhoria de processos.

A estimativa de *software* é uma atividade de processo contínuo que pode ser aplicado em todas as etapas do ciclo de vida do projeto. Com a estimativa, é possível determinar métricas que impactam diretamente na execução do projeto. [Magela 2006] afirma que a palavra estimativa é utilizada quando se tem a proposta de aplicar um modelo a um problema que contenha um conjunto mínimo de dados quantitativos e qualitativos, para prever seu comportamento. A obtenção da estimativa consiste em presumir que o modelo não tem condições de determinar precisamente o resultado desejado ou não possui dados suficientes para fazê-lo.

As principais técnicas de estimativa de projetos de desenvolvimento de *software* trabalham com o princípio de que o tamanho de um *software* é uma métrica decisiva para a determinação do esforço para sua construção. Além disso, essa é uma variante importante para determinar a execução, o custo e tempo de desenvolvimento do projeto [Hazan 2008]. Com essas informações bem definidas, possível garantir um melhor gerenciamento do projeto pela organização, obtendo-se assim maior exatidão no processo.

Para estimar um projeto, é preciso que o responsável pela atividade seja capaz de garantir a qualidade do produto através da análise dos requisitos. A estimativa pode ser feita por um engenheiro de *software* ou algum especialista capaz de analisar os requisitos. Em caso de mudanças nos requisitos pode ser necessário re-estimar [Hazan 2008].

De acordo com [Pressman 2011], as estimativas poder ser confiavelmente determinadas através de várias maneiras. Uma delas seria obtê-las ao final do projeto. No

entanto, essa prática não é válida, pois as estimativas precisam ser obtidas no início do projeto para dar sequência às atividades de planejamento e controle. Outra forma é se basear em estimativas de projetos antigos para estimar o projeto atual. E também existe o uso de técnicas de decomposição ou modelos empíricos para gerar estimativas de custo, esforço e tempo de projeto.

Existem diversas técnicas para a obtenção de estimativas. [Conte & Shen 1985] afirma que os métodos mais conhecidos são: o Modelo de Alocação de Recursos; a Análise de Pontos de Função (APF), que mede o tamanho do *software* pela quantificação de sua funcionalidade externa, baseada no projeto lógico ou a partir do modelo de dados; o *Modelo RCA PRICE S2*; e o COCOMO (*Constructive Cost Model*), que é um modelo voltado para a produção, e que atualmente está na sua segunda fase de desenvolvimento, conhecida como COCOMO II. Com o advento da tecnologia de orientação a objeto no desenvolvimento de projetos e que é adotada por boa parte dos sistemas, a estimativa passou a ser realizada também através de Análise de Pontos de Casos de Uso (UCP). Esta abordagem explora conceitos dos casos de uso e substitui algumas características do modelo APF.

Em metodologias ágeis também existem modelos para estimativas, com a finalidade de obter rapidamente estimativas sobre os requisitos (ou histórias) sem perder muito tempo com métodos mais burocráticos e permitindo a interação entre os membros da equipe. A mais conhecida é a técnica de *Planning Poker*, o modelo de estimativa que será objeto de estudo deste trabalho de graduação.

Este capítulo foca o estudo a três dessas técnicas. Para efeitos de comparação, inicialmente vão ser abordadas duas das principais estimativas usadas por metodologias tradicionais: a Análise de Pontos de Função e a Análise de Pontos de Casos de Uso. E por fim, vai ser realizada uma análise dos conceitos da principal técnica de estimativa para metodologias ágeis, que é o *Planning Poker*, objeto de estudo deste trabalho de graduação e vai comparar com os modelos dos métodos tradicionais.

4.1. Análise de Pontos de Função

A Análise de Ponto de Função (APF) é uma técnica de medição de estimativa das funcionalidades de um determinado projeto, sendo estas avaliadas por um ponto de vista de seu usuário. Surgiu em 1977, quando foi elaborada por Alan Albrecht [Albrecht 1979]. Ponto de função é a unidade de medida desta técnica, cujo propósito é fazer com que o valor de uma estimativa obtida não esteja interligado com quaisquer tecnologias utilizado para a construção

da aplicação, ou seja, esta unidade deve avaliar o que o produto faz desconsiderando os meios necessários para construí-lo [Vazquez *et al* 2010].

A ideia da APF é medir as funcionalidades do sistema, identificar uma forma para medir a produtividade e qualidade na área de sistemas e fornecer uma solução para obter estimativas para o desenvolvimento de *software*. Com ela, é possível dimensionar os sistemas que estejam em produção ou em desenvolvimento, possibilitando assim extrair informações mensuráveis que sejam úteis para estimativa de custos e recursos requeridos para o desenvolvimento e manutenção de *software* [CTIS Informática 2004].

[Albrecht 1979], [CTIS Informática 2004], [Vazquez *et al* 2010] apontam que o processo de contagem dos pontos de função pode ser dividido em algumas etapas, que serão mostradas nas subseções a seguir.

Determinar o tipo de contagem

Essa etapa consiste na identificação do objeto a ser medido. Existem três tipos de contagem: a de projeto em desenvolvimento, a de projeto de melhoria e a de aplicação. Na contagem de projeto de desenvolvimento, são medidas todas as funcionalidades fornecidas ao usuário desde sua primeira instalação. Na contagem de projeto de melhoria, são medidas as funcionalidades que foram adicionadas, modificadas, ou removidas do sistema. Na contagem de aplicação, são medidas as funcionalidades informadas pelos usuários existentes na aplicação atual instalada. Esse último tipo é feito ao final da contagem de projeto de desenvolvimento.

Identificar a fronteira da aplicação

Indica o que faz parte do projeto que está sendo medido, distingue os processos dele das aplicações externas ao domínio e também o relacionamento entre o projeto. Ela determina quais funcionalidades serão incluídas no processo de contagem dos pontos de função.

Contar as funções de dados

Aqui as funcionalidades da aplicação são identificadas e contadas. As funções de dados consistem de:

1. Arquivos Lógicos Internos (ALI), que são grupos relacionados de dados que são alterados na própria aplicação. Como por exemplo: tabelas de um banco de dados mantidas no próprio sistema.

2. Arquivos de Interface Externa (AIE), que são grupos logicamente relacionados de dados que são alterados fora da aplicação. Como por exemplo: tabelas de um banco de dados mantidas num sistema fora da aplicação.

Cada ALI e AIE possuem dois tipos de elementos que devem ser contados para cada função identificada:

1. Itens de Dados: campo único, que é reconhecido pelo usuário e não é repetido, como os campos das tabelas.
2. Registros Internos: subgrupo de dados, reconhecido pelo usuário. Por exemplo: generalização/especialização de classes.

Baseado no número de itens de dados e registros é possível classificar a complexidade do ALI e AIE, conforme a Tabela 3. São atribuídos os pesos de 7, 10 ou 15 PF para os ALIs e os pesos de 5, 7 e 10 para os AIE respectivamente à complexidade baixa, média ou alta [IFPUG 2000].

Tabela 3. Complexidade funcional para os ALI e AIE.

	Itens de Dados		
Registros Lógicos	1 a 19	20 a 50	51 ou mais
1	Simple	Simple	Média
2 a 5	Simple	Média	Alta
6 ou mais	Média	Alta	Alta

Contar as funções transacionais

As funções transacionais representam as funcionalidades de processamento dos dados fornecidas pelo sistema ao usuário. Podem ser:

1. Entrada Externa (EE): funções da aplicação que processa dados que vêm de fora da aplicação.
2. Saída Externa (SE): funções da aplicação que gera dados que são enviados para fora da fronteira da aplicação.
3. Consulta Externa (CE): funções da aplicação que representam solicitações enviadas para a aplicação, a qual gera uma consulta correspondente e a posterior exibição de dados.

Cada EE, SE e CE possuem dois tipos de elementos que devem ser contados para cada função identificada:

1. Itens de Dados: campo único, que é reconhecido pelo usuário e não é repetido, como os campos das tabelas.
2. Arquivos Referenciados: arquivos lógicos utilizados para processar a entrada e/ou saída. É o total de ALI e AIE utilizados pela transação.

Baseado no número de itens de dados e arquivos referenciados é possível classificar a complexidade do EE, conforme a Tabela 4. São atribuídos os pesos de 3, 4 ou 6 PF respectivamente à complexidade baixa, média ou alta [IFPUG 2000].

Tabela 4. Complexidade funcional para os EE.

	Itens de Dados		
Arquivos Referenciados	1 a 4	5 a 15	16 ou mais
0 a 1	Simple	Simple	Média
2	Simple	Média	Alta
3 ou mais	Média	Alta	Alta

Baseado no número de itens de dados e arquivos referenciados é possível classificar a complexidade do EE, conforme a Tabela 5. São atribuídos os pesos de 3, 4 ou 6 PF para o CE e os pesos de 4, 5 ou 7 PF para o SE respectivamente à complexidade baixa, média ou alta [IFPUG 2000].

Tabela 5. Complexidade funcional para os SE e CE.

	Itens de Dados		
Arquivos Referenciados	1 a 5	6 a 19	20 ou mais
0 a 1	Simple	Simple	Média
2 a 3	Simple	Média	Alta
4 ou mais	Média	Alta	Alta

Determinar o PF não ajustado

Após identificar as funções de dados e transacionais, deve-se multiplicar o total de ALI, AIE, EE, SE e CE pela respectiva complexidade para determinar o valor de PF não ajustado (PNFA). Após a determinação dos valores para cada tipo de função, a equação final para os pontos de função não ajustados é:

$$PFNA = \Sigma (\text{funções de dados}) + \Sigma (\text{funções transacionais})$$

Onde “funções de dados” é a multiplicação da quantidade de ALI e AIE pelas respectivas complexidades, e “funções transacionais” é a multiplicação da quantidade de EE, SE e CE pelas respectivas complexidades.

Calcular Valor do Fator de Ajuste

O número de pontos de função não ajustados reflete a funcionalidade que o sistema fornecerá ao usuário, sem considerar que outros fatores, como os requisitos não funcionais, também afetam o tamanho do sistema.

O fator de ajuste tem como finalidade corrigir as distorções da etapa anterior em 35%, baseadas na influência de 14 características gerais, que serão analisadas e fornecerão o valor do fator de ajuste. São elas: Comunicação de Dados, Processamento Distribuído, Performance, Configuração Altamente Utilizada, Taxa de Transações, Entrada de Dados On-Line, Eficiência do Usuário Final, Atualização *On-Line*, Processamento Complexo, Reutilização, Facilidade de Operação, Facilidade de Instalação, Múltiplos Locais e Modificações Facilitadas.

Para cada característica deve ser atribuído um nível de influência numa escala de 0 (nenhuma influência) a 5 (grande influência). O valor do fator de ajuste deve-se segue a seguinte equação:

$$\mathbf{VFA = (GIT * 0,01) + 0,65}$$

Onde VFA é o valor do fator de ajuste e GIT é o grau de influência total (soma de todos os valores dos níveis de influência).

Atualmente, a IFPUG considera essa uma etapa opcional do processo de contagem, para se adequar a norma ISO/IEC 20296:2002 de medição funcional. Isso porque algumas das características são consideradas não aderentes pelo padrão da ISO. Assim, muitas organizações usam apenas a medição dos pontos de função não ajustados [Vazquez *et al* 2010].

Calcular os Pontos de Função Ajustados

Depois de calcular o valor do fator de ajuste, os pontos de função são ajustados, multiplicando-se o mesmo pelo valor dos pontos de função não ajustados (PFNA):

$$\mathbf{APF = VFA \times PFNA}$$

4.2. Análise de Pontos de Casos de Uso

A técnica de Pontos de Casos de Uso é uma métrica para o cálculo do tamanho de *software* orientado a objetos. Tem a sua origem em 1993, quando foram criados numa tese de

doutorado de Gustav Karner, da *Rational Software* [Karner 1993] e sua natureza é baseada na técnica de Análise de Pontos por Função (APF) e em uma metodologia conhecida como Mark II [Albrecht 1979]. Suas determinações são realizadas de acordo com um diagrama de Casos de Uso (UC). Assim, podem ser feitas estimativas com mais antecedências, apesar da técnica de Pontos de Função (PF) ser mais precisa.

Este método visa estimar o tamanho de um sistema de acordo com a maneira como os usuários o utilizam, em uma análise de alto nível dos passos necessários para a realização de cada tarefa em um nível mais abstrato. A contagem dos Pontos de Caso de Uso (PCU) é realizada através de seis atividades:

Calculando o peso dos Atores do sistema

Essa etapa envolve identificar os atores, quais possuem um nível de complexidade. Os atores podem ser classificados em simples, médio e complexo, com pesos 1, 2 e 3, respectivamente, conforme demonstrado na Tabela 6.

Tabela 6. Pesos de atores.

Tipo de Ator	Peso	Descrição
Ator Simples	1	Outro sistema acessado através de uma API de programação
Ator Médio	2	Outro sistema interagindo através de um protocolo de comunicação ou interação de usuário através de linhas de comando.
Ator Complexo	3	Um usuário interagindo através de uma interface gráfica.

Calculando o Peso dos Casos de Uso

Essa etapa objetiva o cálculo do peso bruto dos casos de uso. Para obtenção de valores, os casos de uso são divididos em três níveis de complexidade, baseadas no número de transações envolvidas em seu processamento. A transação é uma série de atividades que podem ser realizados em conjunto. Também se deve considerar a complexidade através da quantidade de classes de análise por caso de uso. A Tabela 7 mostra o peso para cada um dos tipos de caso de uso classificados.

Tabela 7. Pesos de Casos de Uso por número de transações e classes.

Tipo de Caso de Uso	Descrição	Peso
Simples	Até 3 transações e até de 4 classes de análise	5
Médio	De 4 a 7 transações e de 5 a 10 classes de análise	10
Complexo	Mais de 7 transações e mais de 10 classes de análise	15

Cálculo dos Pontos de Casos de Uso não Ajustados

Este cálculo considera a soma do peso total dos atores com o peso total dos casos de uso para obtenção dos pontos de casos de uso não ajustados (PCUNA). Logo,

$$PCUNA = \Sigma (\text{pesos dos atores}) + \Sigma (\text{pesos dos casos de uso})$$

Cálculo dos Fatores Técnicos

Assim como na técnica de Pontos de Função, este método tem um ajuste bastante similar e é composta por duas partes. A primeira delas considera um cálculo de fatores técnicos, que abrange uma série de requisitos funcionais do sistema. Ela basicamente prevê o nível técnico do sistema e dificuldade para desenvolvê-lo.

O objetivo é determinar o Fator de Complexidade Técnica (FCT), a partir do conhecimento técnico do sistema em relação a algumas características, conforme a Tabela 8. Cada uma delas é classificada por uma escala de 0 a 5, que varia de acordo com o grau a influência na complexidade da aplicação a ser construída.

Tabela 8. Fatores e pesos de complexidade técnica na PCU.

Descrição	Peso
Sistemas Distribuídos	2,0
Desempenho da Aplicação	1,0
Eficiência do Usuário Final	1,0
Processamento Interno Complexo	1,0
Reusabilidade de Código em outras aplicações	1,0
Facilidade de Instalação	0,5
Usabilidade	0,5
Portabilidade	2,0
Facilidade de Manutenção	1,0
Concorrência	1,0
Características Especiais de Segurança	1,0
Acesso Direto para Terceiros	1,0
Facilidades Especiais de Treinamento	1,0

Após a determinar os valores para cada fator, o resultado dos Fatores Técnicos (FT) é calculado pela soma dos produtos de cada peso multiplicado por seu respectivo valor na escala

de 0 a 5. A partir do valor dos Fatores Técnicos é obtido o Fator de Complexidade Técnica (FCT), conforme fórmula abaixo:

$$\text{FCT} = 0.6 + (0.01 * \text{FT})$$

Cálculo dos Fatores Ambientais

A segunda parte da técnica do ajuste dos Pontos de Casos de Uso consiste em determinar o cálculo de fatores de ambiente, considerando aspectos não funcionais relacionados ao processo de desenvolvimento, tais como experiência da equipe e estabilidade do projeto.

O objetivo é determinar o Fator de Complexidade Ambiental (FCA), a partir do conhecimento do ambiente do sistema em relação a algumas características, conforme a Tabela 9. Elas são obtidas seguindo a mesma lógica dos fatores técnicos, com pesos pré-determinados.

Tabela 9. Fatores e pesos de complexidade ambiental na PCU.

Fator	Regra de Definição	Peso
F1	Familiaridade com o processo de desenvolvimento de software	1,5
F2	Experiência na aplicação	-1
F3	Experiência em OO, na linguagem e na técnica de desenvolvimento	0,5
F4	Capacidade do líder de análise	0,5
F5	Motivação	1
F6	Requisitos estáveis	1
F7	Trabalhadores com dedicação parcial	-1
F8	Dificuldade da linguagem de programação	2

Após a determinar os valores para cada fator, o resultado dos Fatores Ambientais (FA) é calculado pela soma dos produtos de cada peso multiplicado por seu respectivo valor na escala de 0 a 5. A partir do valor dos Fatores Ambientais é obtido o Fator de Complexidade Ambiental (FCA), conforme a fórmula a seguir:

$$\text{FCA} = 1.4 - (0.03 * \text{FA})$$

Cálculo dos Pontos de Caso de Uso ajustados

Finalmente, o cálculo do valor total do sistema em Pontos de Caso de Uso Ajustados (PCUA) representa a estimativa de tamanho do projeto em Pontos de Casos de Uso. Ele é determinado assim:

$$\text{PCUA} = \text{PCUNA} \times \text{FCT} \times \text{FCA}$$

4.3. Planning Poker

O *Planning Poker* é uma técnica de estimativa voltada para as metodologias ágeis, inclusive o Scrum, diferentemente dos métodos de estimativas anteriores, cujo foco é para abordagens mais tradicionais. Algumas denominações apresentadas aqui farão parte do contexto do Scrum, portanto exigirá um conhecimento básico do mesmo, que foi abordado no Capítulo 3.

[Grenning 2002] afirma que esta é a melhor ferramenta encontrada para estimar tamanho de projetos em metodologias ágeis. O *Planning Poker* pode ser considerado a combinação de três técnicas menos comuns de métodos de estimativas ágeis: opinião de especialista, analogia e desagregação [Cohn 2005], o que a torna uma prática bastante interativa e prática.

O *Planning Poker* consiste na obtenção de estimativa através de um jogo de cartas. A ideia principal por trás dele é permitir que todos os membros da equipe de desenvolvimento (programadores, testadores, design, analistas, etc.) participem colocando a sua visão de complexidade para que juntos possam chegar a um denominador comum para a equipe. Nas metodologias ágeis, as estimativas de esforço e tempo são aplicadas a estórias e é essa atividade é de responsabilidade da equipe técnica por alguns motivos [CTIC-UFPA 2011]:

1. Quando se planeja, normalmente não se sabe exatamente quem vai implementar quais partes de quais estórias;
2. Estórias normalmente envolvem diversas pessoas com diferentes conhecimentos (*design* de interface de usuário, codificação, teste, etc.);
3. Para prover uma estimativa, os membros da equipe precisa de algum tipo de compreensão da natureza de cada estória. Um maior entendimento do contexto aumenta a probabilidade de que os membros se ajudarão durante a iteração. Isso também aumenta a probabilidade de que questões importantes sobre a estória surjam cedo;

4. Quando todos estimam uma estória, frequentemente descobrem-se desconexões onde duas pessoas da equipe têm estimativas bastante diferentes para a mesma estória. É preciso descobrir e discutir essas questões antes de se iniciar a iteração.

No Scrum, o *Planning Poker* é realizado durante o *Sprint Planning Meeting* [Pereira et al 2007]. O objetivo é atribuir uma complexidade em comum a todos os membros da equipe para as estórias de cada item do *Product Backlog* de uma forma mais dinâmica. Estes, portanto, são os principais responsáveis pelas estimativas das estórias.

No *Planning Poker*, cada integrante tem a sua disposição um baralho de 13 cartas, numeradas numa sequência similar a encontrada nos números de Fibonacci, conforme mostra a Figura 7. De acordo com [CTIC-UFGA 2011], existe um propósito ao utilizar uma sequência não linear determinação de estimativas. O objetivo de utilizar estes números é que o intervalo entre os valores permite visualizar melhor as diferenças de complexidade entre as funcionalidades. Essa ideia diminui a granularidade das estórias, o que evita uma falsa ideia de precisão de estimativas. Então se para um item foi dada uma estimativa de 20 pontos, não faz muita diferença se pudesse ser dado um valor próximo. Portanto, este valor trata-se de um palpite aproximado.



Figura 7. Baralho de *Planning Poker* (Fonte: CRISP Konsulter)

Valores acima de 20 são considerados altos e, portanto trata-se de uma estória grande, que podem não ser completamente finalizadas numa única iteração. Uma ideia seria tentar entrar em maiores detalhes, quebrando a estória maior em menores, com redução em suas complexidades. O recomendável é que ao final do *Planning Poker*, se consiga fazer com que as estórias da iteração possuam algum valor do intervalo de 1/2 a 13. No entanto, é necessário evitar outro problema que é deixar que as estórias tornarem-se muito pequenas, pois isso provavelmente tornará a equipe vítima do microgerenciamento [Kniberg 2007].

O baralho do *Planning Poker* ainda pode conter algumas cartas com propósitos especiais. A carta com valor '0' (zero) indica que uma estória está finalizada ou é tão pequena que pode ser resolvida em questão de minutos [Kniberg 2007]. A carta com '?' (interrogação)

indica que o participante não tem o conhecimento apropriado para estimar o esforço necessário para a estória. Não é frequentemente utilizada, mas se for, a equipe precisa trabalhar a respeito da estória para tentar chegar a um melhor consenso entre os membros da equipe. Por último, a carta com xícara de café indica que o participante não está em condições de pensar sobre a estória e necessita de uma pausa para refletir sobre a mesma [CTIC-UFPA 2011].

Durante o *Planning Poker*, devem ser realizadas rodadas para obter a estimativa da estória, com base nestes valores. As diferenças que surgirem durante estas rodadas deverão ser mediadas por um coordenador. No Scrum, este papel é de responsabilidade do *Scrum Master*. O *Product Owner*, por sua vez, será responsável por explicar as estórias, sendo importante para retirar as possíveis dúvidas que surgirem a respeito destas estórias. Não intervém na estimativa [CTIC-UFPA 2011].

Os seguintes procedimentos que devem ser tomados ao se estimar uma estória, quando realizado por uma equipe Scrum [CTIC-UFPA 2011], [Haugen 2006], [Cohn 2005]:

1. Inicialmente pontua-se a estória que mais se aproxima do valor '3' (três), pois servirá como referência para as demais.
2. As estórias são apresentadas por vez. O *Product Owner* descreve para a equipe Scrum a estória e fornece informações a respeito do seu valor de negócio. A seguir, o coordenador vai perguntar o esforço necessário para concluir a história aos membros da equipe, conforme a Figura 8.

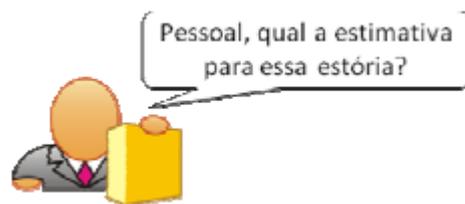


Figura 8. Coordenador solicita aos participantes a estimativa de uma determinada estória.

3. Cada membro da equipe deve pensar a respeito do tempo e esforço necessário para se implementar a estória lida, conforme a Figura 9. Os membros da equipe devem estimar considerando todas as tarefas envolvidas pelos responsáveis em desenvolver, testar, criar *design*, etc. Então, deve escolher uma carta no baralho correspondente ao valor desta estimativa e coloca-a virada para baixo.

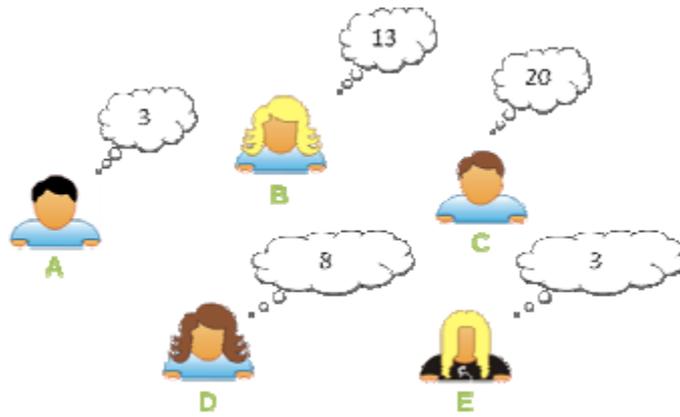


Figura 9. Integrantes da equipe técnica escolhendo suas estimativas da estória.

4. Quando todos os membros fizerem o procedimento acima, então devem revelar as cartas escolhidas simultaneamente, conforme a Figura 10. Isso faz com que cada membro da equipe pense por si próprio na hora de uma decisão de estimativa.



Figura 10. Integrantes da equipe técnica mostrando suas cartas.

5. Todos avaliam os resultados e verificam se houve convergência entre as cartas mostradas, ou seja, todas as estimativas possuam valores aproximados para a mesma estória.
6. Caso contrário, o *Scrum Master* solicita aos membros, que mostraram o menor e o maior valor estimado, que expliquem o motivo que os levaram a tal estimativa. Isto faz com que os integrantes reflitam sobre alguns pontos da estória, o que pode fazer com que mudem os valores propostos para as estimativas. Então, uma nova rodada é realizada até que as estimativas de esforço cheguem a uma convergência, conforme a Figura 11.

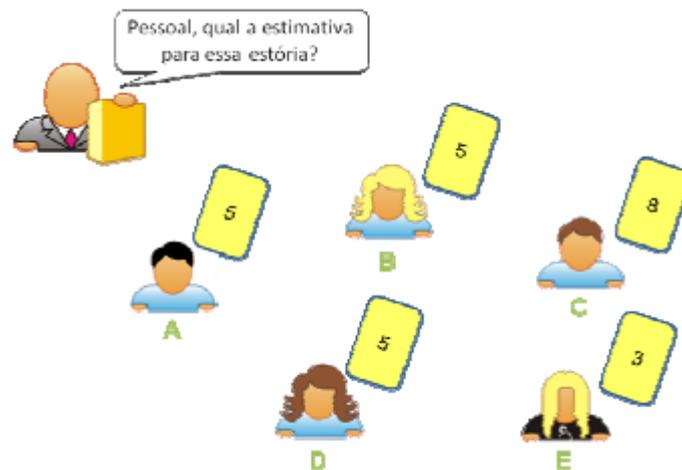


Figura 11. Integrantes da equipe técnica refazendo suas escolhas para a estimativa.

7. A estimativa final da estória será o valor que tiver maior ocorrência ou a média entre as estimativas informadas. Então começa uma nova rodada com a leitura da próxima estória pelo *Product Owner*. O processo se repete até que tenha sido concluída a estimativa das estórias dos itens do *Product Backlog*.

Alguns trabalhos [Haugen 2006] afirmam haver uma melhora na eficiência das estimativas com pontos por estórias quando utilizada esta técnica para atribuição dos pontos. No entanto, um ponto a se levantar é que na utilização dessas técnicas não estruturadas, algumas pessoas com melhor entendimento do contexto da estória podem influenciar no resultado proposto pelo restante da equipe. Ou seja, vai provavelmente fazer com que os outros membros sigam o raciocínio apresentado.

Conforme visto, o *Planning Poker* é uma alternativa muito satisfatória para realização de estimativas na metodologia Scrum. No entanto, este jogo torna-se inviável de ser realizado em equipes geograficamente distantes, pois estas não compartilham um mesmo espaço físico e um mesmo baralho de cartas [Kniberg 2007].

4.4. Considerações Finais

Este capítulo abordou os conceitos de estimativas de software, realizou um estudo das principais técnicas usadas por metodologias tradicionais e por fim foi visto uma técnica de estimativa ágil, que é o *Planning Poker*.

O estudo desses modelos foi importante para avaliar as vantagens de se aplicar uma modelo de estimativas dentro de práticas ágeis. A natureza interativa do *Planning Poker* possibilita que as equipes tenham melhor acompanhamento do esforço necessário para se desenvolver um requisito. Isso é feito dentro de um prazo muito curto, diferente dos modelos

utilizados nas metodologias tradicionais, que costumam aplicar fórmulas matemáticas para extrair uma estimativa de uma funcionalidade. Normalmente estes são modelos mais precisos, mas também são mais burocráticos e metódicos.

O *Planning Poker*, apesar de ser uma excelente técnica para estimativa, não é boa o suficiente quando se considera equipes Scrum distribuídas geograficamente, devido à falta de comunicação entre os membros. Sem o auxílio de uma ferramenta que possa proporcionar a prática desse jogo, dificilmente essa técnica seria realizada por essas equipes. O próximo capítulo vai apontar algumas das ferramentas já existentes e que tratam desse inconveniente do *Planning Poker*.

Capítulo 5

Ferramentas para Estimativas Usando o Planning Poker

Este capítulo aborda algumas das principais ferramentas já disponíveis que trabalham com estimativas usando *Planning Poker*, ou seja, possuem uma finalidade semelhante ao da ferramenta proposta neste trabalho. Foram identificadas duas ferramentas de relevante importância: o Planning Poker e o FireScrum. A seguir, será realizada uma análise mais aprofundada delas.

5.1. Planning Poker

O Planning Poker é uma ferramenta web gratuita e desenvolvida pela *Mountain Goat Software*, cuja finalidade é permitir que equipes ágeis utilizem-na sem necessitar de um *software* específico.

Para utilizar a ferramenta, basta que o usuário realize um cadastro no site e forneça o link para acesso a sessão da(s) partida(s), aos demais componentes da equipe através do e-mail. As partidas funcionam avaliando-se cada item de um *Product Backlog* (não há este termo na ferramenta), cada um com sua estória e o usuário deve descrevê-las. Dentro da ferramenta, o usuário que inicia a sessão é o moderador das partidas, possivelmente o *Scrum Master*.

Planning Poker

Create a new game

You can start inviting estimators after you've created the game.

Name

For example: 'FooBar development project', 'CMS rewrite', or 'Legacy DB migration'

Description (optional)

Import stories (optional)

Copy and paste from a spreadsheet to import stories into your game. Make sure the first row contains field names.

Create game Cancel

Figura 12. Tela de descrição de uma estória no Planning Poker.

Nesta ferramenta, o usuário possui a sua disposição um baralho e pode selecionar uma das cartas dela. Cada uma corresponde a um valor diferente de estimativa para a estória. A seleção é feita através do mouse de maneira bastante intuitiva.

O usuário pode iniciar uma nova rodada, caso exista uma grande divergência de valores. É possível visualizar as rodadas que foram realizadas e quais as cartas definidas por cada usuário. Enquanto ela segue, é possível visualizar quais os participantes até o momento já escolheram suas cartas, conforme a Figura 13. Se alguém ainda não escolheu, todas as cartas são dispostas de modo que ninguém veja qual o seu valor. Ao final quando todos escolherem suas cartas da rodada, só então todas as cartas são mostradas a todos.

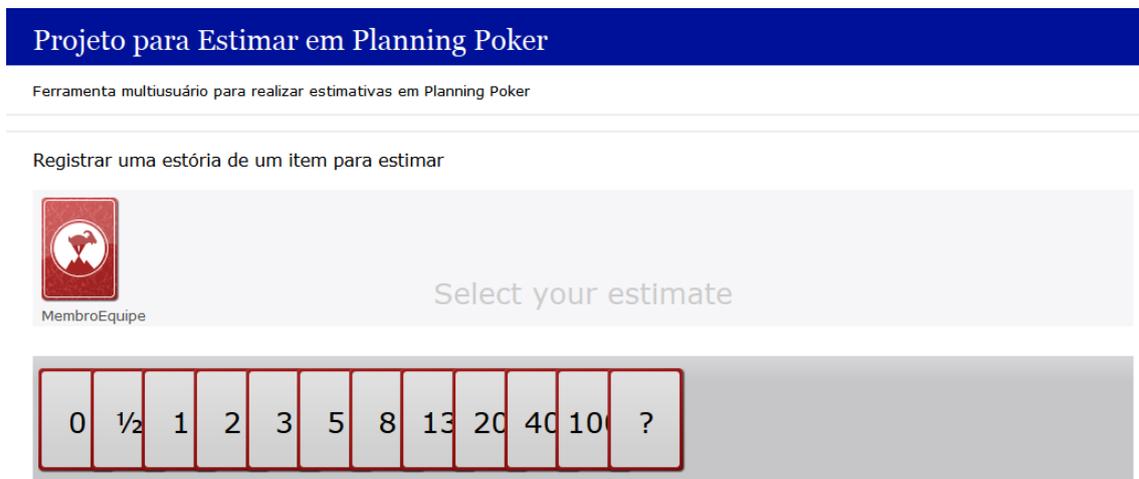


Figura 13. Tela para escolha de uma estimativa no Planning Poker.

O usuário que inicia a sessão tem a opção de definir um valor para o esforço estimado para a estória, para isso basta inserir um valor que ele acha coerente, e se necessário, os membros podem discutir ela. Ao final, o usuário pode estimar uma nova estória ou então encerrar a partida, caso todas as estórias do *Product Backlog* já tenham sido estimadas.



Figura 14. Tela para o moderador definir uma estimativa no Planning Poker.

De modo geral, a ferramenta Planning Poker se mostrou bastante prática e intuitiva, mas carece de vários aspectos que seriam importantes para uma ferramenta que estima com a técnica que leva o nome na ferramenta. Por exemplo, a ferramenta não disponibiliza uma visualização adequada para listar os componentes do *Product Backlog*. Dentro dessa lista, as informações das estórias são exibidas nessa lista, de modo que se uma estória descrita for muito grande, a tendência é que ocupe a tela com informação desnecessária, podendo simplesmente exibir o conteúdo da estória em outro lugar na sua totalidade.

Outro ponto é que não existem maiores detalhes de um item avaliado que vá além das estórias. Elas não podem ser alteradas por nenhum membro, nem mesmo o próprio moderador. Além disso, não existe um meio de comunicação entre os membros da equipe dentro da ferramenta. Provavelmente, a comunicação é realizada através de uma ferramenta a parte. Para finalizar, a ferramenta limita-se a disponibilizar apenas o resultado da estimativa, mas não trabalha com esta informação de modo que possa ser aproveitada para análise pela equipe de trabalho, como por exemplo, a geração de gráficos para acompanhar o desenvolvimento do projeto.

Apesar de grandes limitações, é uma ferramenta que dentro da finalidade intrínseca de estimar, atende bem ao seu papel na tentativa de objetivar informações mensuráveis a partir da avaliação das funcionalidades de um sistema.

5.2. FireScrum

O FireScrum é uma ferramenta *open source* de gerenciamento de equipes de projeto Scrum. Ela contém aplicações integradas para suportar essas equipes no desenvolvimento de seus projetos, contribuindo para equipes que trabalham remotamente. Ele teve início a partir de um trabalho de conclusão de curso e sua ideia foi evoluída em um mestrado. O sistema foi desenvolvido por um grupo de cerca de 60 estudantes de pós-graduação da Universidade Federal de Pernambuco [Cavalcanti 2009].

O FireScrum utiliza abordagens da *Web 2.0* e de RIA (Rich Internet Applications) [Cavalcanti 2009]. É uma aplicação *web* que pode ser acessível através de um ambiente de Internet ou Intranet. Apesar disso, sua configuração exige aplicativos de terceiros para que seja possível instalar o aplicativo. A estrutura do FireScrum foi desenvolvida para possuir funcionalidades modularizadas e integradas, porém independentes entre si.

Dentre os principais módulos da ferramenta, o *Core* é mais elementar deles, sendo capaz de abranger todo o ciclo de um projeto Scrum. As principais funcionalidades incluem:

controle de acesso; cadastro de usuários, projetos, itens de *Backlog*, histórias, *Sprints*, tarefas; priorização de itens de *Backlog*; geração do *Burndown Chart*.

Existem outros módulos do FireScrum, que são considerados de apoio. São eles: *TaskBoard*, *Test Management*, *Bug Tracking*, *Desktop Agent* e *Planning Poker*. Esta seção vai focar apenas neste último módulo.

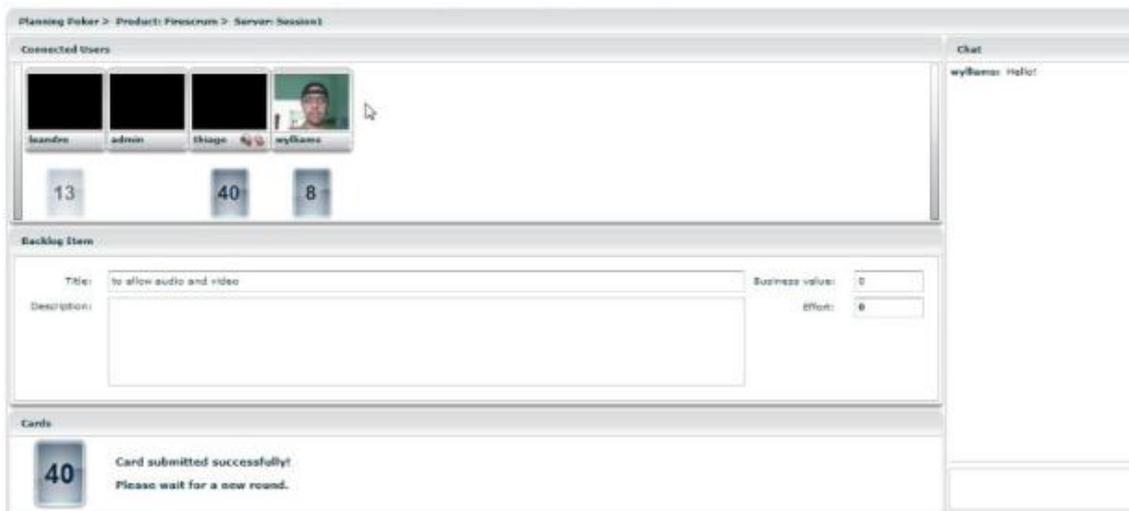


Figura 15. Tela do módulo de Planning Poker do FireScrum.

O *Planning Poker* do FireScrum possui muitos recursos de apoio às equipes de projetos Scrum. Ele contém dispositivos de multimídia para realização de reuniões de estimativas utilizando comunicação através de troca de mensagens ou por videoconferência.

A ferramenta permite a visualização dos itens do *Product Backlog* a serem estimados, e também permite definição dos valores das estimativas. Os itens possuem informações adicionais como o valor de negócio e um valor para expressar o esforço em desenvolvê-lo. Cada participante deve entrar numa sessão definida por um usuário, e através dela, os mesmos irão realizar uma partida de *Planning Poker* no intuito de obter uma estimativa desejada para a história do item avaliado.

Cada participante tem um baralho a sua disposição, do qual é possível escolher uma carta para afirmar a estimativa desejada para a história e aguardam até que todos façam o mesmo. Os participantes podem discutir sobre ela, e uma nova rodada pode ser realizada. No final, o líder registra o valor da estimativa para a história.

O FireScrum parece atender a vários dos requisitos para estimar utilizando o *Planning Poker*, mas existem algumas dificuldades. Uma delas é a necessidade de várias aplicações externas para o funcionamento do sistema. Some-se isso ao fato de que a configuração e instalação destes componentes são um tanto difíceis, exigindo alguns passos trabalhosos para ver o FireScrum rodar. Outro ponto é que apesar do fato de ser *open source*, a falta de uma

documentação ou manual apropriado para uso e instalação da ferramenta pode se tornar um problema ao usuário, visto que pode exigir do mesmo um desperdício de tempo no entendimento de funcionalidades que não são tão intuitivas ou que aparentemente são mais complexas do que realmente é.

Como abordado, as equipes de projeto podem requerer o desenvolvimento de ferramentas próprias para estimar em *Planning Poker*. Por ser *open source*, o FireScrum é uma ferramenta em potencial para servir como referência para o desenvolvimento de aplicações próprias por essas equipes. No entanto, alguns pontos podem dificultar e inviabilizar a compreensão da estrutura da ferramenta por parte um analista de sistema. Um deles é a complexidade do código, pois se trata de uma ferramenta de grande porte e com integração a ferramentas de terceiros; e também a indisponibilidade de uma documentação dos requisitos e da arquitetura do sistema pode ser considerada um contraponto.

5.3. Comparação das Ferramentas

A seguir, será realizado um comparativo entre as ferramentas encontradas e a ferramenta proposta. A ideia é facilitar visualização das características de ambas; verificar os pontos fortes e fracos; e comparar estes elementos com o que vai existir na ferramenta proposta.

Tabela 10. Comparativo entre as ferramentas abordadas.

	Planning Poker	FireScrum	A Ferramenta Proposta
Facilidade de instalação e configuração	Sim	Não	Sim
Visualização das informações na tela	Razoável	Razoável	Boa
Edita <i>Product Backlog</i>	Não	Sim	Sim
Promove comunicação entre os usuários	Não	Sim	Sim
Gráfico de acompanhamento com base nas estimativas obtidas	Não	Sim	Sim
Disponibilidade da documentação da ferramenta	Não	Não	Sim
Manual do usuário e de instalação	Não	Não	Sim
Complexidade do código-fonte	Não disponível	Alta	Média ou Baixa

5.4. Considerações Finais

As ferramentas explicadas aqui trabalham bem quando o objetivo é estimar em *Planning Poker*, apesar de possuírem algumas limitações próprias. A ferramenta proposta deste trabalho de graduação visa trabalhar em cima de alguns dos problemas encontrados, para que novos projetos possam ser desenvolvidos utilizando como base o trabalho da documentação a ser realizada. O capítulo a seguir vai tratar da estrutura dessa nova proposta de ferramenta, que tem como base todo o conhecimento adquirido até aqui.

Capítulo 6

A Ferramenta Proposta

Este capítulo apresenta a proposta de ferramenta deste trabalho. Após identificar as características positivas e limitações das ferramentas similares apresentadas no Capítulo 5, o objetivo será combinar esses elementos, simplificá-los da melhor forma possível e determinar uma lista de funcionalidades que possam ser utilizadas em trabalhos futuros de desenvolvimento.

Após entender o funcionamento destas ferramentas e combinando os conceitos de *Planning Poker* e da metodologia Scrum, o primeiro passo foi elicitar os requisitos funcionais e as principais regras de negócio que vão ser aplicadas na ferramenta.

Os requisitos funcionais aqui apresentados seguem o padrão da metodologia Scrum. Ou seja, eles foram definidos através de histórias de usuário, que servem para descrever as funcionalidades. Portanto, este trabalho realizou um levantamento de um *Product Backlog*, conforme visto nos conceitos de Scrum. De acordo com [Mike 2008], os requisitos não funcionais podem ser expressos através de histórias dos usuários, que seguem o padrão descrito no Capítulo 3. O resultado deste esforço encontra-se no Apêndice A para os requisitos funcionais e no Apêndice B para os requisitos não funcionais.

Para cada história do *Product Backlog* referente aos requisitos funcionais, foi apresentado um protótipo de tela que representa a navegação visual do usuário quando utilizar tal funcionalidade. Além disso, foram apresentados comportamentos de sucesso e erro para cada uma delas, indicando um fluxo de eventos até a ocorrência do fato. Também foi destinado um espaço para notas, que são informações adicionais acerca das histórias.

Por hora, não vão ser definidas prioridades ou estimativas aqui. Essa atividade é de responsabilidade da equipe de desenvolvimento que utilizará as informações levantadas neste trabalho e então vão definir essas informações para cada história, de acordo com as *Sprints*, prazos, custos, riscos e outros fatores relevantes ao projeto de desenvolvimento. A finalidade deste trabalho é definir uma estrutura de análise e projeto da ferramenta proposta.

Foram utilizados neste trabalho alguns dos diagramas UML, que são comuns em metodologias tradicionais, tais como o diagrama de classes e o diagrama de casos de uso. Apesar de ambas não serem comumente utilizadas em metodologias ágeis, o presente trabalho abordou recursos visuais, para uma maior compreensão do sistema. Ou seja, a ideia é utilizar as vantagens de cada metodologia para o maior entendimento do domínio por parte de todos os envolvidos. Isso vai exigir um conhecimento mínimo sobre UML por parte das equipes

projeto, o usuário não pode ser atribuído aos dois papéis, ele só pode ter uma responsabilidade ou outra. No entanto, ele pode assumir os dois papéis em projetos diferentes.

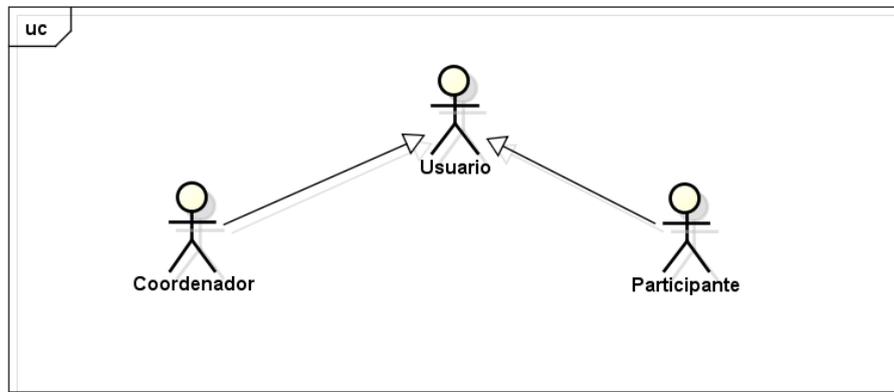


Figura 17. Diagrama de Casos de Uso de relacionamento dos atores da ferramenta.

O diagrama da Figura 18 representa os itens do *Product Backlog* que estão relacionados ao ator Usuário. Como descrito anteriormente, este ator é uma generalização dos demais papéis. Isso implica que os outros atores utilizarão as funcionalidades do usuário.

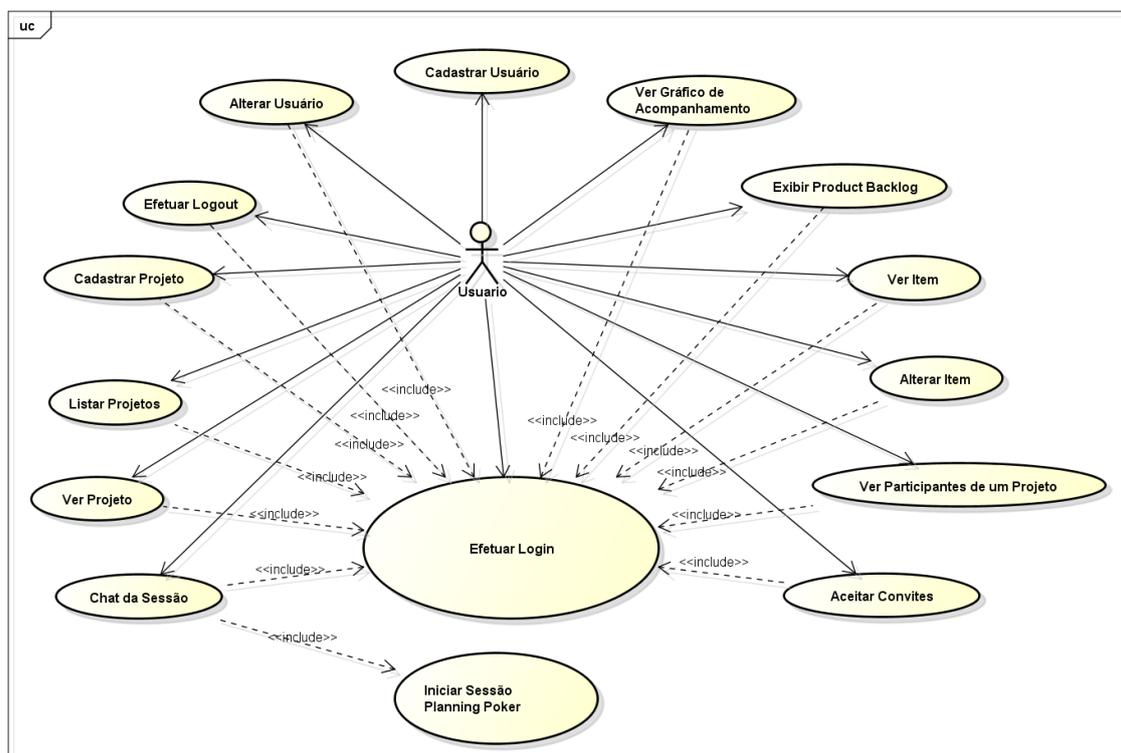


Figura 18. Diagrama de Casos de Uso do ator Usuário.

Algumas atividades são importantes destacar. Uma delas é o “Cadastrar Projeto”, que pode ser realizada por qualquer usuário, e uma vez executada, automaticamente o sistema vai

considerar que ele será um Coordenador (*Scrum Master*) daquele projeto, e assim executar as funcionalidades específicas, conforme mostra a Figura 19.

Já a atividade de “Aceitar Convites”, é a aceitação por parte de um usuário para participar de um projeto. Uma vez realizada, deve considerar que o usuário se torne um participante daquele projeto, e assim, ele será capaz de executar as também funcionalidades específicas (ver Figura 20). Já a atividade de “Chat da Sessão”, possui uma dependência da funcionalidade “Iniciar Sessão Planning Poker”, realizada pelo coordenador (ver Figura 19).

Outra atividade importante é a de “Alterar Item”. Ela pode ser feita por qualquer usuário, visto que as práticas do Scrum estabelecem que não só o coordenador possa editar informações de um item, mas o participante também possa fazê-lo. Na realidade, essa função não é do *Scrum Master*, mas sim do *Product Owner*, no entanto este trabalho assume que as atividades deste último sejam de responsabilidade do coordenador do projeto na ferramenta. Ou seja, ambos os papéis serão considerados coordenadores.

A seguir, o diagrama da Figura 19 representa os itens do *Product Backlog* que estão relacionados ao ator Coordenador. Essas funcionalidades são específicas dele. Assim sendo, outros atores não possuem autorização suficiente no sistema para executá-las.

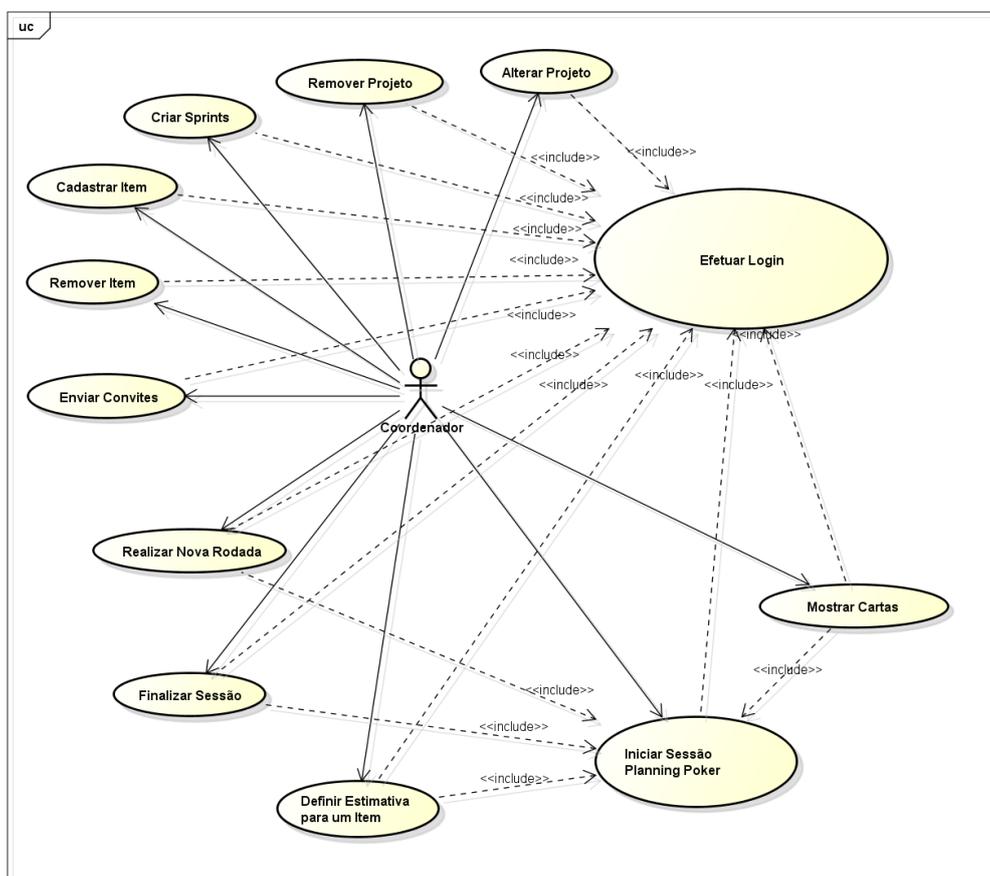


Figura 19. Diagrama de Casos de Uso do ator Coordenador.

As seguintes atividades são exclusivas de um *Scrum Master*: “Remover Projeto”; “Alterar Projeto”; “Criar *Sprints*”, para estabelecer o intervalo das iterações do projeto; “Enviar Convites” para convidar participantes para estimar em um projeto; “Iniciar Sessão *Planning Poker*”; “Realizar Nova Rodada”; “Finalizar Sessão”; “Mostrar Cartas”; e “Definir Estimativa para um Item”. Dentre as atividades do *Product Owner*, que são incluídas como sendo as de um coordenador, tem: “Cadastrar Item”, para ser adicionado ao *Product Backlog* do projeto; e “Remover Item”, para retirá-lo do *Product Backlog*.

Observe que todas as atividades exigem que o coordenador deva ter efetuado o *login* antes. Esta é uma atividade do ator Usuário, portanto também deve ser executada pelo coordenador. As atividades envolvendo a obtenção de estimativas no *Planning Poker* também precisam da atividade de iniciar a sessão. Essas funcionalidades

A seguir, o diagrama da Figura 20 representa os itens do *Product Backlog* que estão relacionados ao ator Participante.

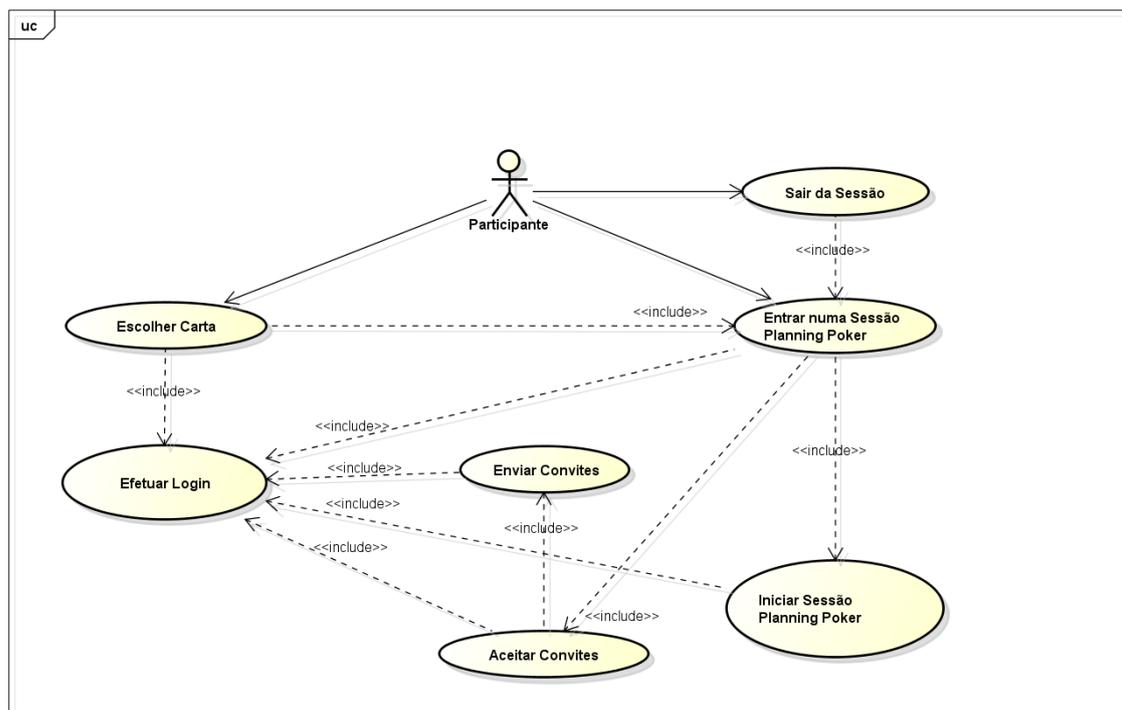


Figura 20. Diagrama de Casos de Uso do ator Participante.

As atividades específicas do ator Participante são: “Entrar numa Sessão *Planning Poker*”, que só pode ser feita após o participante efetuar *login*, haver aceitado um convite (que fora enviado por um coordenador) e a sessão do jogo ser iniciado pelo ator Coordenador; “Escolher Carta”, que só pode ser realizada após efetuar o *login* e ter realizado a atividade anterior; e “Sair da Sessão”, que só pode ser realizada após o participante entrar nela.

6.2. Diagrama de Classes

O diagrama de classes é uma representação da estrutura e das relações entre as classes que modelam um objeto. A figura 21 representa o diagrama de classes da ferramenta.

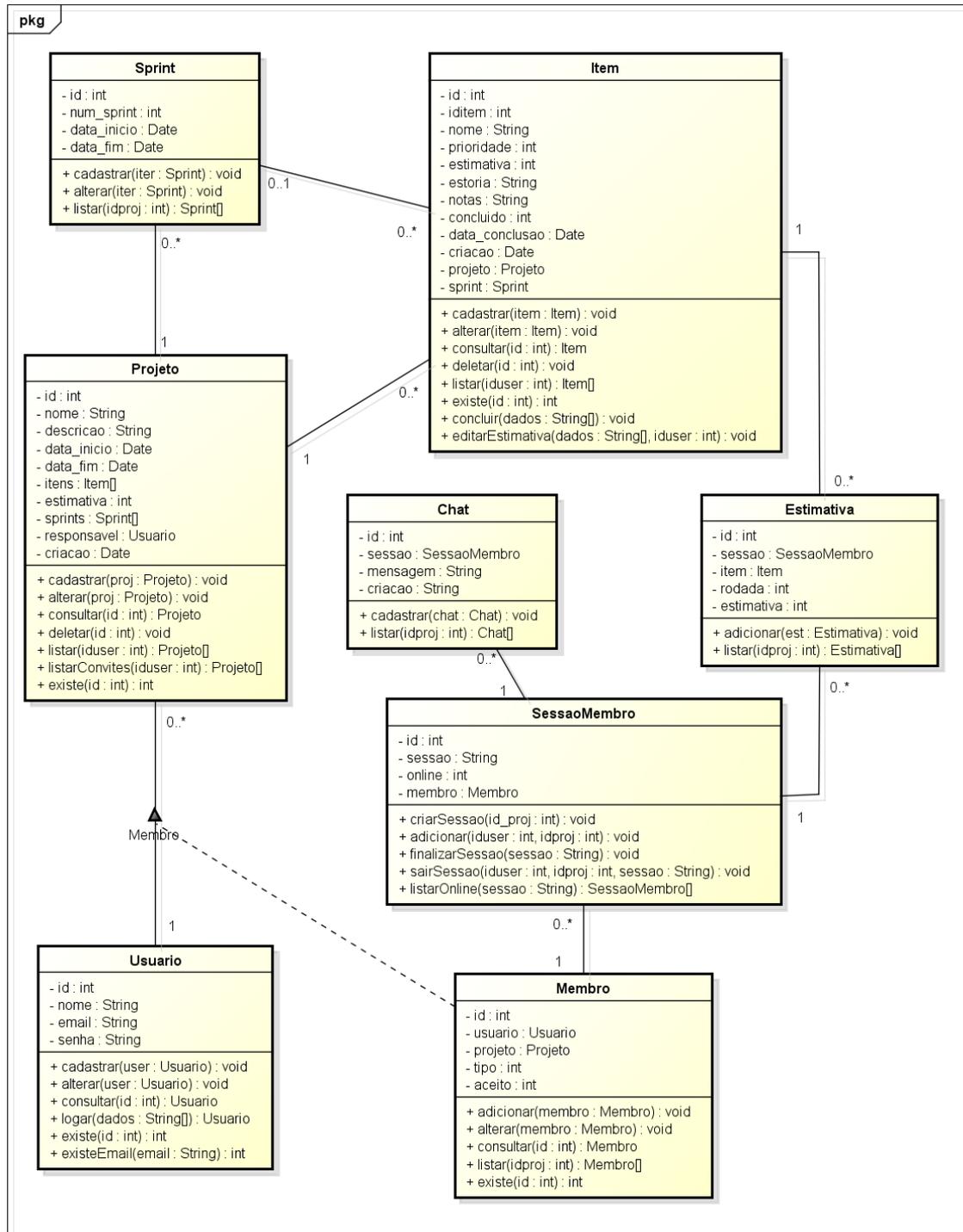


Figura 21. Diagrama de Classes da ferramenta.

A seguir, uma descrição de cada uma das classes do diagrama:

1. **Usuário:** contém informações referentes aos usuários do sistema. Isso inclui os coordenadores e participantes. Pode estar vinculado a vários projetos.
2. **Projeto:** possui informações referentes aos projetos do sistema. Possui um responsável, que é o coordenador (*Scrum Master*), do tipo Usuário e vários membros. Também pode possuir vários itens e *Sprints* associados.
3. **Membro:** classe associativa entre as classes Usuário e Projeto. Ou seja, vincula um usuário a um projeto. Também armazena outros dados, como o tipo do participante do Projeto (se é coordenador ou participante) e também a confirmação da aceitação desses elementos na participação do projeto.
4. **Sprint:** possui informações referentes às *Sprints* (iterações) de um projeto. Cada *Sprint* é vinculada a um projeto e pode estar (ou não) presente em itens do *Product Backlog*.
5. **Item:** possui informações referentes ao item de um *Product Backlog*. Ele deve estar sempre vinculado a um projeto e no máximo a uma *Sprint*. Além disso, o item pode conter várias estimativas realizadas nos jogos de *Planning Poker*.
6. **SessaoMembro:** classe responsável por manter a informação da sessão de um participante num jogo de *Planning Poker*. Verifica o nome da sessão ao qual ele pertence e se está dentro dela ou não. Vale lembrar que numa mesma sessão podem participar vários membros. Isso se todos compartilharem o mesmo nome de sessão, o que ocorre para aqueles que estão no mesmo projeto.
7. **Estimativa:** armazena informações das estimativas, como a rodada dela e o valor informado da carta. Esses dados são vinculados a uma sessão de um membro.
8. **Chat:** armazena informações de uma mensagem enviada por uma sessão de um membro. Ou seja, as informações das mensagens de *chat* são modeladas nessa classe e depois disponibilizadas aos usuários que compartilham da mesma sessão do membro em questão.

6.3. Diagrama Entidade-Relacionamento

O diagrama entidade-relacionamento é uma representação para um modelo de dados de um sistema num alto nível de abstração. Neste trabalho, a estrutura dele foi baseada no diagrama de classes definido acima. A Figura 22 representa o diagrama E-R da ferramenta.

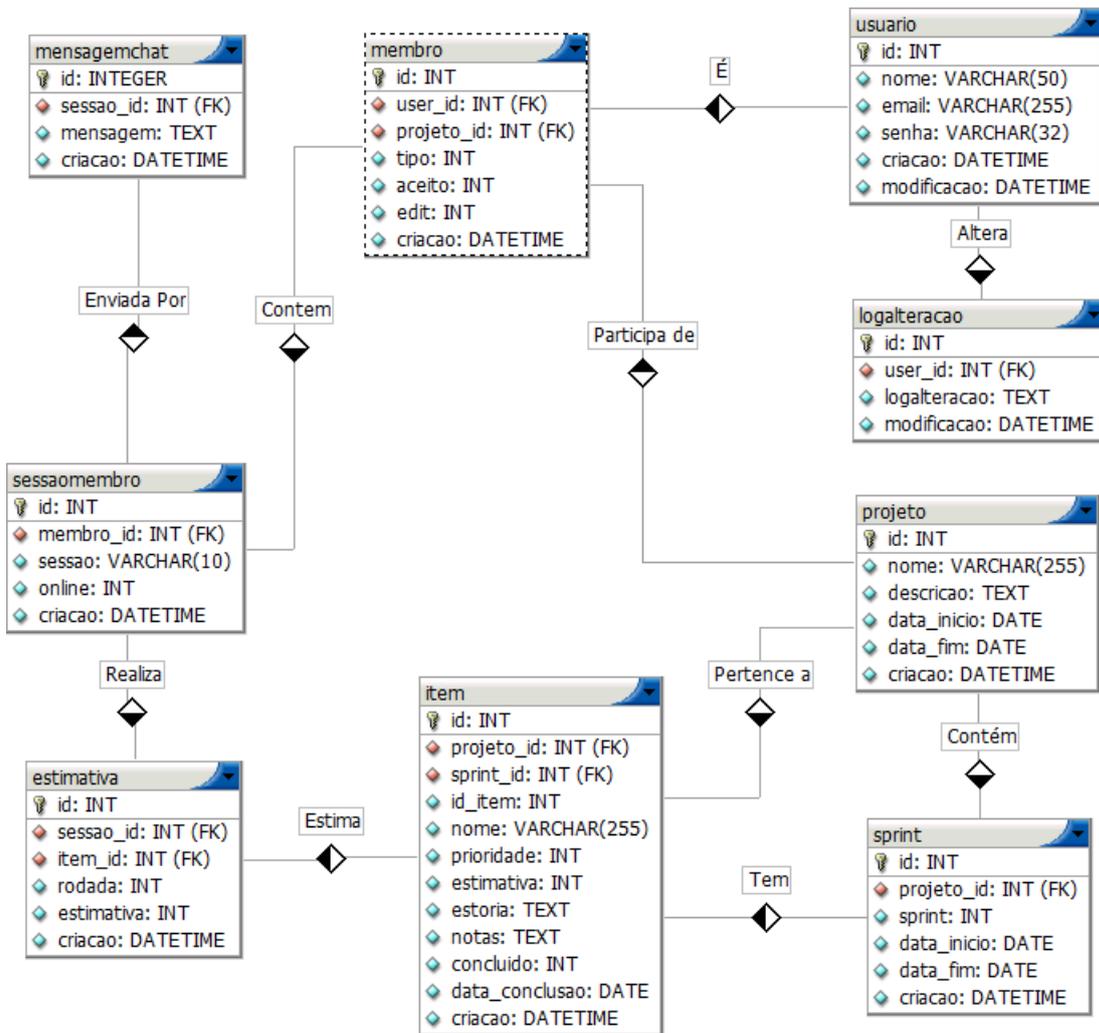


Figura 22. Diagrama E-R da ferramenta.

6.4. Arquitetura do Sistema

A arquitetura de um sistema consiste na definição dos componentes de *software*, suas propriedades externas, e os relacionamentos com outros sistemas. A Figura 23 representa um modelo da arquitetura da ferramenta.

A arquitetura é totalmente dividida em camadas e isso foi necessário ser feito para organizar e separar os diferentes componentes e promovendo a coesão do sistema ao compactar as partes semelhantes. Devido à resolução muito pequena da Figura 23, é realizada a seguir uma análise por cada camada da arquitetura.

A estrutura é baseada no padrão MVC (Model-View-Controller), que se utiliza de três partes. A View é a apresenta a interface visual do sistema que contém a saída de dados; o Controller é responsável pela invocação de objetos ao Model e pelo controle da comunicação

que vem do usuário para o sistema e vice-versa; e o Model que define o contexto da aplicação, regras de negócio e mudança sobre os dados.

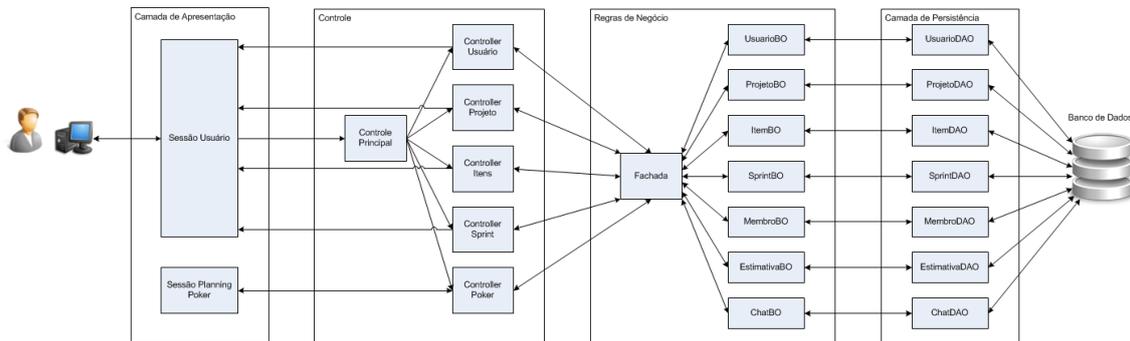


Figura 23. Arquitetura da ferramenta.

A Figura 24 representa a camada de apresentação. Nela são incluídos todos os artefatos relacionados com a interação direta do usuário com a aplicação (páginas PHP; arquivos JavaScript e CSS; imagens). As telas do sistema estão nesta camada, incluindo as telas do *Planning Poker*. Esta camada tem a responsabilidade de receber as informações vindas dos usuários e enviar para a linguagem do negócio.

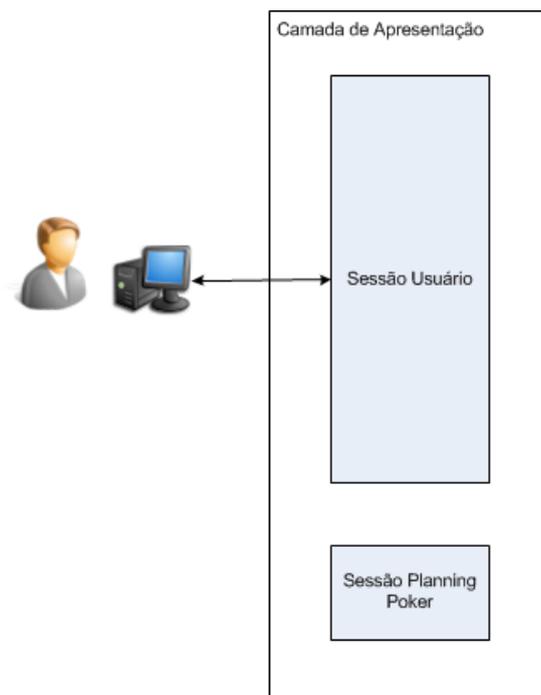


Figura 24. Camada de apresentação.

A Figura 25 representa a camada de controle. Nela são incluídos todos os controles do sistema. Existe um controlador principal, que valida a sessão de um usuário e redireciona para controladores específicos, que por sua vez são responsáveis por receber/enviar dados particulares dependendo da funcionalidade executada pelo usuário.

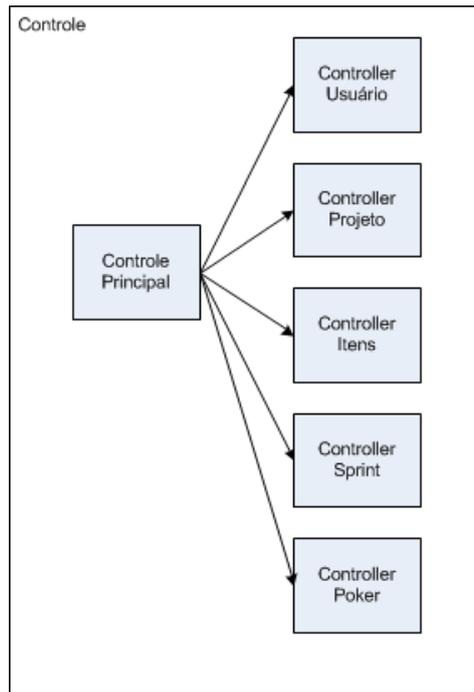


Figura 25. Camada de controle.

A Figura 26 representa a camada de negócios. Nela está incluída toda a lógica, regras, validações do negócio e é responsável por manipular os dados vindos das camadas anteriores, e com o objetivo de expor as informações para a camada de apresentação. Ela é composta de uma classe fachada, que é baseada no padrão de projeto *Façade* e cuja estrutura unifica as interfaces do conjunto de subsistemas, simplificando a utilização dos mesmos. As entidades principais do diagrama de classes possuem em sua estrutura as próprias regras de negócio (com exceção de *SessaoMembro*, cuja lógica está incluída em *MembroBO*).

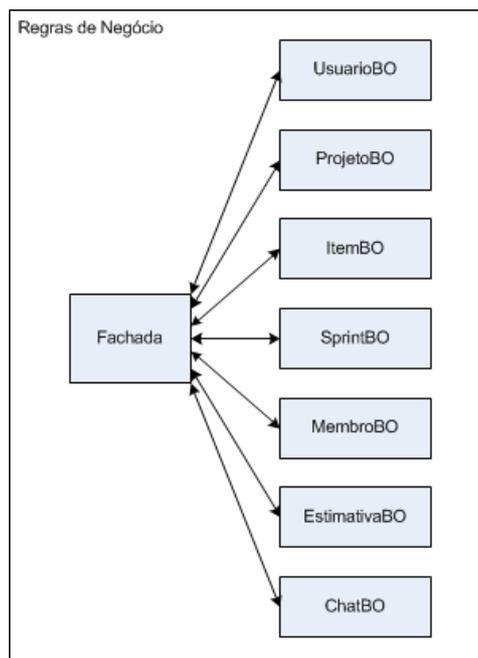


Figura 26. Camada de negócio.

A Figura 27 representa a camada de persistência. Ela tem a responsabilidade de armazenar as informações geradas pela camada de negócio e ao mesmo tempo serve como fonte de dados, pelo fato de ser integrada a um banco de dados.

As entidades principais do diagrama de classes possuem sua própria estrutura para integração com o banco (com exceção de SessaoMembro, cujas funcionalidades de persistência estão incluídas em MembroDAO).

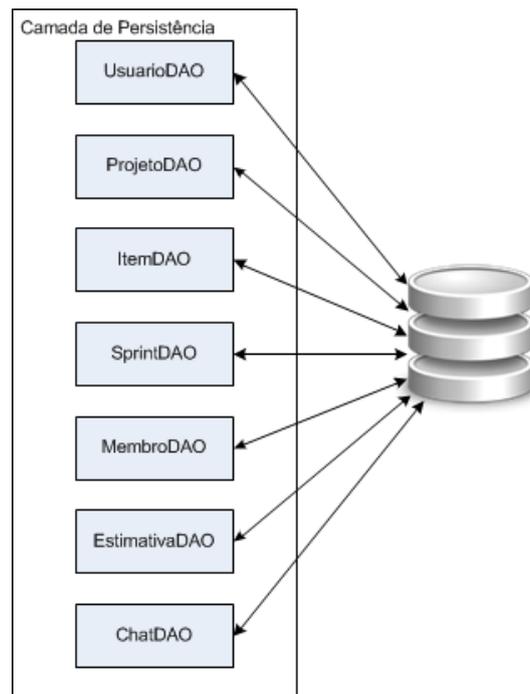


Figura 27. Camada de persistência.

6.5. Protótipo

Para a elaboração desta ferramenta, foi desenvolvido um protótipo para melhor compreensão do entendimento do negócio e para visualizar a navegação das telas. Num primeiro momento, a proposta desse trabalho era criar um protótipo que fosse puramente visual. No entanto, foi concebido o desenvolvimento de algumas funcionalidades, seguindo a arquitetura e a lógica explanada nas seções anteriores deste capítulo.

Para o desenvolvimento do protótipo foi escolhido um ambiente *web*, devido à portabilidade (independência de plataforma) e maior disponibilidade do serviço aos usuários. E embora, tem sido planejado para este ambiente, o desenvolvimento da ferramenta não se limita apenas neste tipo de tecnologias. Ela pode ser aplicável a sistemas *standalone* ou que utilizem *plugins* externos, como é o caso de algumas aplicações Java.

Sobre as tecnologias usadas no protótipo, como servidor web foi escolhido o Apache, devido a sua robustez e segurança. Como linguagem de desenvolvimento, foi utilizado o PHP (*Hypertext Preprocessor*) pela facilidade de implementação, flexibilidade, eficácia e fácil integração com várias tecnologias. E por fim, o banco de dados utilizado foi o MySQL.

Para cada funcionalidade (item do *Product Backlog*) levantada deste trabalho, foi elaborada uma tela correspondente à funcionalidade descrita na estória. É possível visualizar a figura do protótipo das telas de cada uma delas no Apêndice A, separadas por item.

Quanto às estórias funcionais no protótipo, elas seguiram toda a arquitetura descrita no documento, e assim como o diagrama de classes. A organização das tabelas no banco de dados foi baseada no diagrama E-R descrito. Dentre as estórias implementadas, podem ser destacadas aquelas que envolvem cadastro e alteração de usuários; autenticação no sistema (*login* e *logout*); cadastro, alteração, remoção e listagem de projetos; criação de *Sprints*; cadastro, alteração, remoção e listagem de itens do *Product Backlog*; sistemas de convites para participantes do projeto (envio e aceitação); troca de mensagem entre usuários numa sessão de *Planning Poker*; e visualização de um gráfico de acompanhamento da evolução do projeto (*Burndown Chart*).

Embora a implementação do protótipo funcional fosse um requisito extra neste trabalho, ela em geral foi bem realizada dentro do cronograma estabelecido para este trabalho de graduação. O principal contraponto desta atividade é que as estórias relacionadas à sessão e ao jogo do *Planning Poker* não foram desenvolvidas a tempo.

6.6. Considerações Finais

Esse capítulo explicou uma proposta de ferramenta para se estimar com a técnica de *Planning Poker*. Devido à falta de informações apropriadas para o desenvolvimento de um projeto com essa finalidade, a ideia desse trabalho foi realizar elaborar uma documentação das funcionalidades necessárias para atender às características de uma técnica de estimativa de *Planning Poker* e que fosse aplicável a projetos Scrum.

Para isso, foi feito um procedimento para concepção e elaboração de um projeto de software, como elicitação dos requisitos, desenvolvimento dos diagramas fundamentais, arquitetura do sistema, além do desenvolvimento de um protótipo visual e parcialmente funcional, para melhor visualização e entendimento do contexto da ferramenta como um todo.

Capítulo 7

Conclusão

Sabe-se que a estimativa é a atividade que envolve mensurar uma observação sobre determinado ponto. Dentro das práticas de desenvolvimento de *software*, a arte de estimar é fundamental para o gerenciamento de projetos, visto que através dela é possível obter informações relevantes sobre esforço, custos, cronograma, qualidade, riscos e outros. Para isso, necessária alguma forma que possa avaliar do tamanho de um sistema, que é a principal métrica responsável para a elaboração e construção dele.

Dada a importância do Scrum nos últimos tempos e sua consolidação cada vez mais notável como uma metodologia de desenvolvimento de *software*, é essencial utilizar um recurso que se possa trabalhar com técnicas estimativas ágeis que sejam compatíveis com os fundamentos dele.

O objetivo deste trabalho foi apresentar uma proposta de ferramenta para o apoio às estimativas baseadas na técnica de *Planning Poker*, que é bastante comum em projetos ágeis, utilizando os fundamentos da metodologia Scrum. A finalidade é que equipes de projeto possam compreender as funcionalidades por trás da elaboração deste trabalho e assim desenvolver suas próprias ferramentas. Apesar de usar muitas das práticas ágeis, como o levantamento de um *Product Backlog*, também foi utilizado recursos de metodologias tradicionais como diagramas E-R e UML. A ideia era combinar as práticas das duas metodologias para melhor compreensão do domínio, do contexto e da estrutura da ferramenta.

Como abordado, o *Planning Poker* é um modelo que atua de forma interativa e colaborativa, o que exige a presença da equipe num mesmo local para poder realizar as estimativas com sucesso. No entanto, existem muitas dificuldades para se trabalhar com *Planning Poker* em equipes distribuídas geograficamente, devido a falta de maior controle e comunicação entre os membros. Um dos grandes pilares de motivação deste trabalho é justamente reduzir os impactos provocados pela falta de interações presenciais.

A ferramenta apresentada é de natureza multiusuário. A ideia foi elicitar os principais requisitos dela, considerando todas as características abordadas, para que o trabalho possa ser desenvolvido em projetos futuros. Inicialmente, a proposta deste trabalho é que a aplicação dessa ferramenta fosse voltada para a *web*, embora a estrutura planejada possa abranger outras tecnologias.

Por fim, foi desenvolvido um pequeno protótipo visual para melhor entendimento das funcionalidades descritas neste trabalho. A ideia é compreender a navegação das telas e visualizar o conjunto da obra de modo mais próximo de um sistema real em execução. O protótipo utilizado neste trabalho é de código aberto para que outros projetos possam aproveitá-lo e adaptá-lo às próprias necessidades.

7.1. Trabalhos Futuros

Como trabalhos futuros, as funcionalidades da ferramenta proposta podem ser colocadas como referência para o desenvolvimento de projetos que tenham a finalidade de estimar com *Planning Poker*. Mais do que isso, a ferramenta colabora para a aplicação dessa técnica de estimativa num projeto real que utiliza a metodologia Scrum, através do acompanhamento da evolução por gráficos, com base na unidade de pontos de estória.

Além disso, a ferramenta serve como um pequeno sistema de gerenciamento para projetos Scrum, visto que sua elaboração foi pensada para que as equipes pudessem realizar modificações de dados do projeto e dos componentes do *Product Backlog* quando necessários, além de trabalhar com *Sprints*. Portanto, uma proposta futura pode utilizar os requisitos descritos neste trabalho como base para a criação de uma pequena ferramenta de gerenciamento de projetos adaptável para a empresa/organização do interessado.

Por fim, pode ser concebido juntamente a essa ferramenta um pequeno sistema de integração com as ferramentas de gerenciamento de projetos já existentes no mercado, tais como o TargetProcess ou o VersionOne, para facilitar o trabalho de equipes que já utilizam esse tipo de ferramentas e que por ventura tenham a pretensão de migrar os dados de um *Backlog* para o sistema proposto por este trabalho.

Referências Bibliográficas

Abu, Nelson S. R. J. (2010). "*Blog do Abu - Material de Apoio*". Versão 2.1. Disponível: <http://www.tavares.net.br/files/ebooks/SCRUM_BlogDoAbu_ApostilaDeApoio_v2_1.pdf>.

Acesso em 25 mai. 2012.

Agile Manifesto. (2011). "*Manifesto for Agile Software Development*". Agile Alliance. Disponível em: <<http://www.agilemanifesto.org/>>. Acesso em: 05 mai. 2012.

Albrecht, A. J. (1979). "*Measuring Applications Development Productivity*". IBM Applic. Dev. Joint SHARE/GUIDE Symposium. Monterey: [s.n.].

Ambler, Scott W. (2009). "*Examining the Agile Manifesto*". Disponível em: <<http://www.ambysoft.com/essays/agileManifesto.html>>. Acesso em: 05 mai. 2012.

Bavani, Raja. (2009). "*Critical Success Factors in Distributed Agile for Outsourced Product Development*". Computer Society of India, 2009. Disponível em: <<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.167.430&rep=rep1&type=pdf>>.

Acesso em 16. mai. 2012

Cavalcanti, E. de O. (2009). "*FIRESCRUM: Ferramenta de Apoio À Gestão Ágil de Projetos Utilizando SCRUM*". Dissertação (Mestrado) — C.E.S.A.R - Centro de Estudos e Sistemas Avançados do Recife, Recife.

Carvalho, B.V.; Mello, C.H.P. (2009). "*Revisão, Análise e Classificação da Literatura sobre o Método de Desenvolvimento de Produtos Ágil Scrum*". XII Simpósio de Administração da Produção, Logística e Operações Internacionais (SIMPOI), São Paulo.

Clifton, Mark. Dunlap, J. (2003). "*What is SCRUM?*". 18 de Agosto. Disponível em: <<http://www.codeproject.com/Articles/4798/What-is-SCRUM>>. Acesso em: 05 mai. 2012.

Cohn, Mike. (2005) "*Agile Estimating and Planning*", Prentice Hall PTR. 368p.

Cohn, Mike. (2008) "*Non-functional Requirements as User Stories*". Disponível em: <<http://www.mountangoatsoftware.com/blog/non-functional-requirements-as-user-stories>>. Acesso em: 10 jul. 2012.

Conte, S.D.; Shen V.Y. (1985). "*Software Engineering Metrics and Models*". Menlo Park, California: Benjamin/Cummings Publishing, 1985.

CTIC-UFGA. (2011). "*Procedimentos do Planning Poker*", versão 1.3. Centro de Tecnologia da Informação e Comunicação da Universidade Federal do Pará.

CTIS Informática. (2004). "*Manual de Contagem de Ponto de Função*". Setembro. Disponível em: <<http://sergiolaranja.net63.net/Blog/Livros/ManualdeContagemdePontosdeFuncao.pdf>>

Dantas, V. F. (2003). "*Uma Metodologia para o Desenvolvimento de Aplicações Web num Cenário Global*". Dissertação. Universidade Federal de Campina Grande. Campina Grande, Julho.

DeMarco, T. (1991). "*Controle de Projetos de Software*". 9.ed. Rio de Janeiro: Editora Campus.

DIVUS Tecnologia. (2012). "*Gestão de Projetos com Scrum*". Disponível em: <http://www.divus.com.br/bibliotecaVirtual/Apostila_Curso_GP_SCRUM_DIVUS-v1.1.pdf>. Acesso em: 10 jun. 2012.

Florac, William A.; Park, Robert E.; Carleton, Anita D. (1997). "*Practical software measurement: measuring for process management and improvement*" [S.l.], 1997. Disponível em: <<http://www.sei.cmu.edu/pub/documents/97.reports/pdf/97hb003.pdf>>. Acesso em: 20 mai. 2012.

Freire, Herval. (2003). "*Calculando Estimativas: o Método de Pontos de Caso de Uso*". Developer's Magazine, número 78. Fevereiro.

Haugen, N. C. (2006). "*An Empirical Study of Using Planning Poker for User Story Estimation*". Proceedings of the Conference on AGILE 2006 (pp. 23-34). Washington (DC): IEEE Computer Society.

Hazan, Cláudia. (2008) "*Análise de Pontos de Função: Uma aplicação nas estimativas de tamanho de Projetos de Software*". Engenharia de Software, Rio de Janeiro, 01 de junho.

Highsmith, J., Cockburn A. (2001). "*Agile Software Development: The Business of Innovation*". Pgs. 120-122. (2001)

Huzita, Elisa H. M.; da Silva, César A.; Wiese, Igor S.; Tait, Tania F. C., Quinaia, Marcos; Schiavoni, Flávio L. (2008). "*Um Conjunto de Soluções para Apoiar o Desenvolvimento Distribuído de Software*". In: II Workshop de Desenvolvimento Distribuído de Software - WDDS, Campinas, SP. Disponível em: <<http://www.lbd.dcc.ufmg.br:8080/colecoes/wdds/2008/011.pdf>>. Acesso em: 16 mai. 2012.

IFPUG. (2000). "*Function Point Counting Practices Manual: Release 4.3*". International Function Point Users Group. Jan.

Karner, G. (1993). "*Metrics for Objectory*". University of Linköping. Sweden. (LiTH-IDA-Ex-9344:21).

Kniberg, Henrik (2007). "*Scrum and XP from the Trenches*". Disponível em: <<http://www.infoq.com/minibooks/scrum-xp-from-the-trenches>>. Acesso em: 10 mai. 2012.

Lima, Adailton M.; Reis, Rodrigo Q. (2008) "*Compartilhamento de Informações sobre Processos em Ambientes Descentralizados de Desenvolvimento de Software*". In: II Workshop de Desenvolvimento Distribuído de Software - WDDS, Campinas, SP. Disponível em: <<http://www.lbd.dcc.ufmg.br:8080/colecoes/wdds/2008/008.pdf>>. Acesso em: 16 mai. 2012.

Martins, J. C. C. (2009). "*Técnicas para gerenciamento de projetos de software*". São Paulo: Brasport.

Mountain Goat Software. (2012). "*Planning Poker*". Disponível em: <<http://www.planningpoker.com>>. Acesso em: 10 mai. 2012.

Mundim, A. P. F.; Rozenfeld, H.; Amaral, D. C.; Silva, S. L.; Guerreiro, V.; Horta, L. C. (2002). "*Aplicando o cenário de desenvolvimento de produtos em um caso prático de capacitação profissional*". In: *Gestão & Produção*. v. 9, nº. 1, p. 1-16, Abril.

Pereira, Paulo; Torreão, Paula; Marçal, Ana Sofia. (2007). "*Entendendo Scrum para Gerenciar Projetos de Forma Ágil*". Disponível em: <<http://www.siq.com.br/DOCS/EntendendoScrumparaGerenciarProjetosdeFormaAgil.pdf>>. Acesso em: 06 mai. 2012.

Pressman, R. S. (2011). "*Engenharia de Software*". 7ª ed. São Paulo: Ed. McGraw-Hill.

Reis, Helena M. (2010). "*Scrum: Método Ágil para Gestão de Projetos de Software*". 09 out. Disponível: <<http://helenamcd.ueuo.com/images/artigos/gestaodeprojetos/scrum.pdf>>. Acesso em: 17 mai. 2012.

Royce, W. W. (1970). "*Managing the development of large software systems: concepts and techniques*". Proc. IEEE Westcon, Los Angeles, CA.

Santos, Aline et al. (2010). "*Experiência Acadêmica de uma Fábrica de Software utilizando Scrum no Desenvolvimento de Software*". In: IV Workshop Brasileiro de Métodos Ágeis (Agile Brazil), PUC-RS, Porto Alegre, RS. p. 86-98. Disponível em: <<http://www.agilebrazil.com/2010/pt/wbma2010.pdf>>. Acesso em: 16 mai. 2012.

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

- Schwaber, K. (2004). "*Agile Project Management with Scrum*". Microsoft Press.
- Šmite, D., Wohlin, C., Gorschek, T., and Feldt, R. (2010). "*Empirical Evidence In Global Software Engineering: A Systematic Review*". *Empirical Software Engineering*, vol. 15, pp. 91–118.
- Soares, Michel dos Santos. (2004). "*Comparação entre Metodologias Ágeis e Tradicionais para o Desenvolvimento de Software*". Unipac - Universidade Presidente Antônio Carlos, Faculdade de Tecnologia e Ciências de Conselheiro Lafaiete.
- Soares, M. D. S. (2009). "*Metodologias Ágeis Extreme Programming e Scrum para o Desenvolvimento de Software*". *Revista Eletrônica de Sistemas de Informação, Conselheiro Lafaiete*, v. 3, n. 1, 2009. ISSN 1677-3071.
- Sutherland, Jeff et al. (2007) "*Distributed Scrum: Agile Project Management with Outsourced Development Teams*". In: *Proceedings of the 40th Hawaii International Conference on System Sciences*, 40.
- Varaschim, Jacques D. (2009). "*Implantando o SCRUM em um Ambiente de Desenvolvimento de Produtos para Internet*". Departamento de Informática da PUC. Rio de Janeiro. Fev.
- Vazquez, C. E.; Simões, G. S.; Albert, R. M.. (2010). "*Análise de Pontos por Função: Medição, Estimativas e Gerenciamento de Projetos de Software*". 9a. Edição. Érica. São Paulo: 2010.
- Woodward, Elizabeth; Surdek, Steffan; Ganis, Matthew. (2010). "*A Practical Guide to Distributed Scrum*". IBM. Disponível em: <<http://pt.scribd.com/doc/42021760/A-practical-guide-to-Distributed-Scrum>>. Acesso em: 16 mai. 2012.
- Yoshima, Rodrigo. (2007). "*Gerenciamento de Projetos com Scrum*". Aspercom. Abril. Disponível em: <<http://www.aspercom.com.br/ead/mod/resource/view.php?id=245>>. Acesso em 15 mai. 2012.
- Zanatta, Alexandre. (2004) "*xScrum: uma proposta de extensão de um Método Ágil para Gerência e Desenvolvimento de Requisitos visando adequação ao CMMI Florianópolis*". Dissertação (Mestrado em Ciência da Computação) - Curso de Pós-Graduação em Ciência da Computação, Universidade Federal de Santa Catarina. Disponível em: <www.tede.ufsc.br/teses/PGCC0651.pdf>. Acesso em: 12 mai. 2012.

Apêndice A – Product Backlog

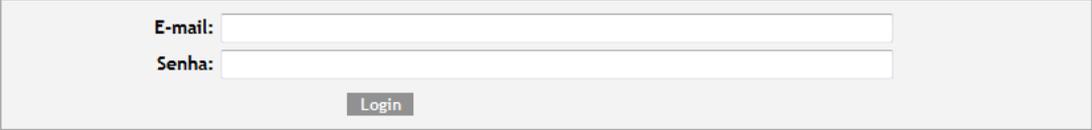
Este apêndice apresenta o *Product Backlog* da ferramenta. Ele contém as histórias que descrevem o funcionamento do sistema e envolvem as tarefas a serem desenvolvidas e posteriormente utilizadas pelos usuários. As histórias foram descritas na forma de cartões, conforme mostrado nas Figuras 3 e 4 do Capítulo 3 deste documento.

Na etapa de levantamento das histórias, as seguintes funcionalidades foram identificadas:

Cadastrar Usuário

ID: 001	Nome: Cadastrar Usuário
Como um usuário, eu gostaria de poder cadastrar no sistema, para que eu possa ser identificado e ter posterior acesso ao sistema.	
 <p>Cadastrar novo usuário</p> <p>Nome: <input type="text"/></p> <p>E-mail: <input type="text"/></p> <p>Senha: <input type="password"/></p> <p>Confirmar Senha: <input type="password"/></p> <p><input type="button" value="Cadastrar"/> <input type="button" value="Voltar"/></p>	
Comportamento de Sucesso: <ol style="list-style-type: none">1. Preencher corretamente todos os campos no formato indicado e clicar em "Cadastrar". Deve retornar a tela principal do sistema e exibir uma mensagem de sucesso.	
Comportamento de Erro: <ol style="list-style-type: none">1. Preencher os campos num formato que não é o indicado ou não preencher pelo menos um deles e clicar em "Cadastrar". Deve exibir uma indicação de erro nos campos incorretos.2. Preencher a senha diferente da confirmação e clicar em "Cadastrar". Deve exibir uma indicação de erro nos campos de senha.3. Preencher um e-mail que já existe no sistema e clicar em "Cadastrar". Deve exibir uma mensagem de erro.	
Notas: <ol style="list-style-type: none">1. Os campos devem ser: nome, e-mail, senha e confirmar senha.2. Utilizar criptografia para a senha.	

Efetuar Login

ID: 002	Nome: Efetuar Login
Como um usuário, eu gostaria de poder realizar o <i>login</i> no sistema, para que eu possa ter acesso ao sistema.	
<p>Acesso ao sistema</p>  <p>The mockup shows a light gray rectangular area containing two input fields. The first field is labeled 'E-mail:' and the second is labeled 'Senha:'. Below these fields is a button labeled 'Login'.</p>	
Comportamento de Sucesso: <ol style="list-style-type: none">1. Preencher corretamente o e-mail e a senha informados no sistema e clicar em “Login”. Deve ter acesso a página principal do usuário.	
Comportamento de Erro: <ol style="list-style-type: none">1. Não preencher pelo menos um dos campos e clicar em “Login”.. Deve exibir uma indicação de erro nos campos não preenchidos.2. Preencher um e-mail ou senha incompatíveis ou que não existam no sistema e clicar em “Login”. Deve exibir uma mensagem de erro.	
Notas: <ol style="list-style-type: none">1. Os campos devem ser: e-mail e senha.2. Utilizar criptografia para a senha.	

Efetuar Logout

ID: 003 **Nome:** Efetuar Logout

Como um usuário, eu gostaria de poder realizar *logout* no sistema, para que eu possa sair do sistema e minhas informações não ficassem mantidas na sessão dele.



Seja bem-vindo, Dennis Silveira.

Projetos que você coordena (como Scrum Master)

Projeto	Data Inicial	Data Final
Arrecadação	13/07/2012	15/07/2012

Comportamento de Sucesso:

1. Clicar no link "Sair" do menu do sistema. Deve sair do sistema e encerrar a sessão do usuário.

Listar Projetos

ID: 004	Nome: Listar Projetos																					
Como um usuário, eu gostaria listar os projetos separados por coordenador e por participante para que eu possa ver em quais deles participo como as coes que eu possa realizar nele.																						
Seja bem-vindo, Dennis Silveira.																						
Projetos que você coordena (como Scrum Master)																						
<table border="1"><thead><tr><th>Projeto</th><th>Data Inicial</th><th>Data Final</th><th>Product Backlog</th><th>Estimativa</th><th>Sprints</th><th>Ações</th></tr></thead><tbody><tr><td>Arrecadação</td><td>01/01/2010</td><td>31/12/2013</td><td>0 Itens</td><td></td><td>0</td><td>    </td></tr><tr><td>Estimativa com Planning Poker</td><td>13/04/2012</td><td>10/07/2012</td><td>0 Itens</td><td></td><td>0</td><td>    </td></tr></tbody></table>		Projeto	Data Inicial	Data Final	Product Backlog	Estimativa	Sprints	Ações	Arrecadação	01/01/2010	31/12/2013	0 Itens		0	    	Estimativa com Planning Poker	13/04/2012	10/07/2012	0 Itens		0	    
Projeto	Data Inicial	Data Final	Product Backlog	Estimativa	Sprints	Ações																
Arrecadação	01/01/2010	31/12/2013	0 Itens		0	    																
Estimativa com Planning Poker	13/04/2012	10/07/2012	0 Itens		0	    																
Projetos que você participa (como membro do Scrum Team)																						
<table border="1"><thead><tr><th>Projeto</th><th>Data Inicial</th><th>Data Final</th><th>Product Backlog</th><th>Estimativa</th><th>Sprints</th><th>Ações</th></tr></thead><tbody><tr><td>YumeYo</td><td>28/02/2012</td><td>30/06/2012</td><td>0 Itens</td><td></td><td>0</td><td>  </td></tr><tr><td>Archimedes</td><td>01/02/2011</td><td>31/12/2012</td><td>0 Itens</td><td></td><td>0</td><td>  </td></tr></tbody></table>		Projeto	Data Inicial	Data Final	Product Backlog	Estimativa	Sprints	Ações	YumeYo	28/02/2012	30/06/2012	0 Itens		0	  	Archimedes	01/02/2011	31/12/2012	0 Itens		0	  
Projeto	Data Inicial	Data Final	Product Backlog	Estimativa	Sprints	Ações																
YumeYo	28/02/2012	30/06/2012	0 Itens		0	  																
Archimedes	01/02/2011	31/12/2012	0 Itens		0	  																
Comportamento de Sucesso:																						
<ol style="list-style-type: none">1. Ao acessar o sistema, serão listados os projetos que o usuário participa como coordenador e como participante.																						
Notas:																						
<ol style="list-style-type: none">1. Deve apresentar informações essenciais como data inicial, data final, número de itens do <i>Product Backlog</i>, estimativa total do projeto e número de <i>Sprints</i>.2. Para o coordenador, as ações que devem ser mostradas por projeto são: “Ver Projeto”, “Alterar Projeto”, “Ver <i>Product Backlog</i>”, “Manter <i>Sprints</i>”, “Ver Gráfico de Acompanhamento”, “Enviar Convite”, “Lista de Membros” e “Remover Projeto”.3. Para o participante, as ações que devem ser mostradas por projeto são: “Ver Projeto”, “Ver <i>Product Backlog</i>”, “Ver Gráfico de Acompanhamento” e “Lista de Membros”.																						

Alterar Usuário

ID: 005 **Nome:** Alterar Usuário

Como um usuário, eu gostaria de poder alterar meu nome ou senha, para que eu possa ser identificado de outra maneira no sistema.

Tela 1:



Seja bem-vindo, Dennis Silveira.

Projetos que você coordena (como Scrum Master)

Projeto	Data Inicial	Data Final
Arrecadação	01/01/2010	31/12/2013
Estimativa com Planning Poker	13/04/2012	10/07/2012

Tela 2:

Alterar usuário

Nome:

E-mail:

Senha:

Confirmar Senha:

Senha Atual:

Comportamento de Sucesso:

1. Clicar no link "Alterar Usuário" no menu principal. Preencher o nome e clicar em "Alterar". Deve retornar a tela de acesso do usuário e exibir uma mensagem de sucesso.
2. Clicar no link "Alterar Usuário" no menu principal. Preencher uma nova senha, a confirmação e a senha atual e clicar em "Alterar". Deve retornar a tela principal do sistema e exibir uma mensagem de sucesso.

Comportamento de Erro:

1. Preencher a senha diferente da confirmação e clicar em "Alterar". Deve exibir uma indicação de erro nos campos de senha.
2. Preencher a senha atual incorretamente e clicar em "Alterar". Deve exibir uma mensagem de erro.
3. Não preencher pelo menos um dos campos e clicar em "Alterar". Deve exibir uma indicação de erro nos campos não preenchidos.

Notas:

1. Os campos devem ser: nome, e-mail (exibição), senha (nova), confirmar senha e senha atual.
2. Utilizar criptografia para a senha.

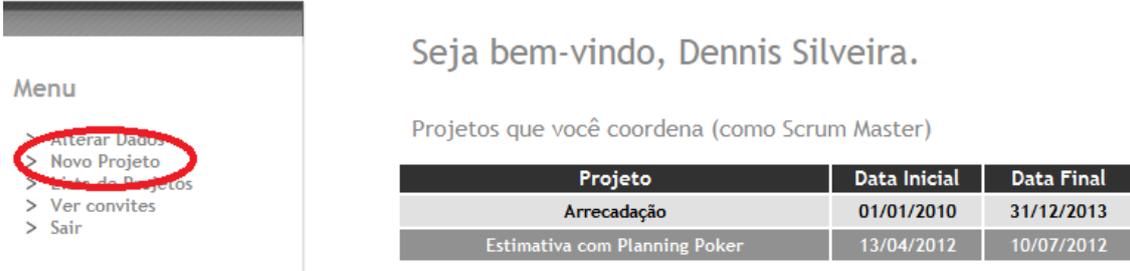
Cadastrar Projeto

ID: 006

Nome: Cadastrar Projeto

Como um usuário, eu gostaria de poder cadastrar projetos do qual faço parte, para que eu possa coordená-los.

Tela 1:



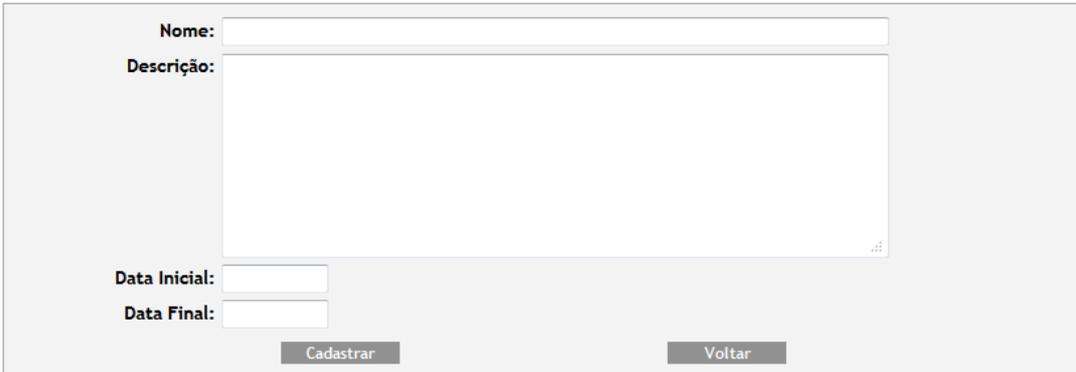
Seja bem-vindo, Dennis Silveira.

Projetos que você coordena (como Scrum Master)

Projeto	Data Inicial	Data Final
Arrecadação	01/01/2010	31/12/2013
Estimativa com Planning Poker	13/04/2012	10/07/2012

Tela 2:

Cadastrar novo projeto



Nome:

Descrição:

Data Inicial:

Data Final:

Comportamento de Sucesso:

1. Clicar no link "Novo Projeto" no menu principal. Preencher corretamente todos os campos no formato indicado e clicar em "Cadastrar". Deve retornar a tela de acesso do usuário e exibir uma mensagem de sucesso.

Comportamento de Erro:

1. Preencher os campos num formato que não é o indicado ou não preencher pelo menos um deles e clicar em "Cadastrar". Deve exibir uma indicação de erro nos campos incorretos.

Notas:

1. Os campos devem ser: nome, descrição, data inicial e data final.

Ver Projeto

ID: 007

Nome: Ver Projeto

Como um usuário, eu gostaria de poder ver detalhes de um projeto do qual faço parte, para que eu possa ter maiores informações sobre ele.

Tela 1:

Seja bem-vindo, Dennis Silveira.

Projetos que você coordena (como Scrum Master)

Projeto	Data Inicial	Data Final	Product Backlog	Estimativa	Sprints	Ações
Arrecadação	01/01/2010	31/12/2013	0 Itens		0	    
Estimativa com Planning Poker	13/04/2012	10/07/2012	0 Itens		0	    

Tela 2:

Visualizar dados do Projeto Estimativa com Planning Poker

Nome:	<input type="text" value="Estimativa com Planning Poker"/>
Responsável:	<input type="text" value="Dennis Silveira"/>
E-mail do Responsável:	<input type="text" value="dwas@cin.ufpe.br"/>
Descrição:	<input type="text" value="Projeto de TG"/>
Data Inicial:	<input type="text" value="13/04/2012"/>
Data Final:	<input type="text" value="10/07/2012"/>
Iterações:	<input type="text" value="0"/>
Estimativa:	<input type="text"/>
Itens:	<input type="text" value="0"/>
<input type="button" value="Voltar"/>	

Comportamento de Sucesso:

1. Ao clicar no link "Ver Informações do Projeto", será possível ver informações mais específicas de um projeto.

Notas:

1. Os campos visualizados devem ser: nome, responsável, e-mail do responsável, descrição, data inicial, data final, número de iterações, estimativa e número de itens no *Product Backlog*.

Alterar Projeto

ID: 008

Nome: Alterar Projeto

Como um coordenador, eu gostaria de poder alterar projetos do qual faço parte, para que eu possa modificar algumas informações sobre ele.

Tela 1:

Seja bem-vindo, Dennis Silveira.

Projetos que você coordena (como Scrum Master)

Projeto	Data Inicial	Data Final	Product Backlog	Estimativa	Sprints	Ações
Arrecadação	01/01/2010	31/12/2013	0 Itens		0	
Estimativa com Planning Poker	13/04/2012	10/07/2012	0 Itens		0	

Tela 2:

Alterar Projeto Estimativa com Planning Poker

Nome:

Descrição:

Data Inicial:

Data Final:

Comportamento de Sucesso:

1. Clicar no link "Alterar Projeto" de um dos projetos. Preencher corretamente todos os campos no formato indicado e clicar em "Alterar". Deve retornar a tela principal do sistema e exibir uma mensagem de sucesso.

Comportamento de Erro:

1. Preencher os campos num formato que não é o indicado ou não preencher pelo menos um deles e clicar em "Alterar". Deve exibir uma indicação de erro nos campos incorretos.

Notas:

1. Os campos devem ser: nome, descrição, data inicial e data final.

Manter Sprints

ID: 009

Nome: Manter Sprints

Como um coordenador, eu gostaria criar manter uma quantidade de *Sprints* definidas para os projetos, cada uma com seus intervalos, para que eu possa definir seus prazos e associar itens a eles. Gostaria de poder editá-las também, se necessário.

Tela 1:

Seja bem-vindo, Dennis Silveira.

Projetos que você coordena (como Scrum Master)

Projeto	Data Inicial	Data Final	Product Backlog	Estimativa	Sprints	Ações
Arrecadação	01/01/2010	31/12/2013	0 Itens		0	
Estimativa com Planning Poker	13/04/2012	10/07/2012	0 Itens		0	

Tela 2:

Manter Sprints

Projeto: Estimativa com Planning Poker

N° de Sprints:

Início Sprint 1:

Final Sprint 1:

Início Sprint 2:

Final Sprint 2:

Início Sprint 3:

Final Sprint 3:

Início Sprint 4:

Final Sprint 4:

Comportamento de Sucesso:

1. Clicar no link "*Sprints*" de um dos projetos. Preencher uma quantidade de *Sprints* desejadas. Depois, devem-se preencher no formato correto todas as datas de início e fim das *Sprints* e clicar em "Alterar". Deve retornar a tela do usuário e exibir uma mensagem de sucesso.

Comportamento de Erro:

1. Não preencher o campo da quantidade de *Sprints* ou preencher num formato que não é o indicado e clicar em "Ok". Deve exibir uma indicação de erro nesse campo.
2. Não preencher pelo menos um dos campos de Sprints ou preencher num formato que não é o indicado e clicar em "Alterar". Deve exibir uma indicação de erro nos campos não preenchidos.

Notas:

1. Os campos devem ser: início e final da *Sprint* na quantidade informada.

Enviar Convites

ID: 010	Nome: Enviar Convites
----------------	------------------------------

Como um coordenador, eu gostaria enviar convites a outros usuários do sistema para que eles possam colaborar no projeto como participantes.

Tela 1:

Seja bem-vindo, Dennis Silveira.

Projetos que você coordena (como Scrum Master)

Projeto	Data Inicial	Data Final	Product Backlog	Estimativa	Sprints	Ações
Arrecadação	01/01/2010	31/12/2013	0 Itens		0	
Estimativa com Planning Poker	13/04/2012	10/07/2012	0 Itens		0	

Tela 2:

Enviar Convites para o Projeto Archimedes

N° de convites:

Email do convidado 1:

Email do convidado 2:

Email do convidado 3:

Email do convidado 4:

Email do convidado 5:

Comportamento de Sucesso:

1. Clicar no link "Enviar Convites" de um dos projetos. Preencher uma quantidade de convites. Depois, devem-se preencher no formato correto os e-mails dos usuários que deseja convidar e que estejam cadastrados no sistema e clicar em "Enviar". Deve retornar a tela de acesso do usuário e exibir uma mensagem de sucesso.

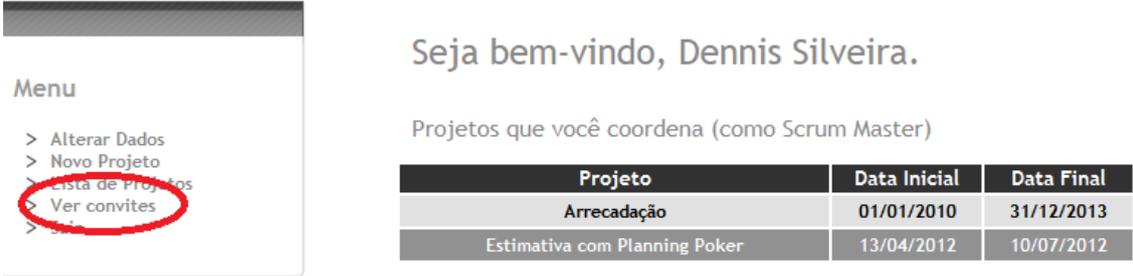
Comportamento de Erro:

1. Não preencher o campo da quantidade de convites ou preencher num formato que não é o indicado e clicar em "Ok". Deve exibir uma indicação de erro nesse campo.
2. Não preencher pelo menos um dos campos ou num formato que não é o indicado e clicar em "Enviar". Deve exibir uma indicação de erro nos campos não preenchidos.
3. Preencher e-mails válidos, mas que não são de usuários cadastrados no sistema. Deve exibir uma mensagem de erro dos e-mails não cadastrados.

Notas:

1. Os campos devem ser os e-mails dos convidados na quantidade informada.

Aceitar Convites

ID: 011	Nome: Aceitar Convites												
Como um usuário, eu gostaria aceitar convites vindos de outros usuários do sistema para que eu possa participar (ou não) do projeto deles.													
Tela 1:  <p>Menu</p> <ul style="list-style-type: none">> Alterar Dados> Novo Projeto> Lista de Projetos> Ver convites> ... <p>Seja bem-vindo, Dennis Silveira.</p> <p>Projetos que você coordena (como Scrum Master)</p> <table border="1"><thead><tr><th>Projeto</th><th>Data Inicial</th><th>Data Final</th></tr></thead><tbody><tr><td>Arrecadação</td><td>01/01/2010</td><td>31/12/2013</td></tr><tr><td>Estimativa com Planning Poker</td><td>13/04/2012</td><td>10/07/2012</td></tr></tbody></table>		Projeto	Data Inicial	Data Final	Arrecadação	01/01/2010	31/12/2013	Estimativa com Planning Poker	13/04/2012	10/07/2012			
Projeto	Data Inicial	Data Final											
Arrecadação	01/01/2010	31/12/2013											
Estimativa com Planning Poker	13/04/2012	10/07/2012											
Tela 2: <p>Sua lista de convites.</p> <table border="1"><thead><tr><th>Projeto</th><th>Responsável</th><th>E-mail</th><th>Ações</th></tr></thead><tbody><tr><td>YumeYo</td><td>Fulano</td><td>fulano@gmail.com</td><td>✓ ✖</td></tr><tr><td>Archimedes</td><td>Cicrano</td><td>cicrano@gmail.com</td><td>✓ ✖</td></tr></tbody></table>		Projeto	Responsável	E-mail	Ações	YumeYo	Fulano	fulano@gmail.com	✓ ✖	Archimedes	Cicrano	cicrano@gmail.com	✓ ✖
Projeto	Responsável	E-mail	Ações										
YumeYo	Fulano	fulano@gmail.com	✓ ✖										
Archimedes	Cicrano	cicrano@gmail.com	✓ ✖										
Comportamento de Sucesso: <ol style="list-style-type: none">1. Clicar no link "Ver Convites" no menu principal. Será exibida uma lista de convites para o usuário, com o nome do projeto a ser convidado. Depois, deve-se escolher a opção para aceitar participar do projeto. Deve retornar a tela de acesso do usuário e exibir uma mensagem de sucesso na operação escolhida.													
Comportamento de Erro: <ol style="list-style-type: none">1. Clicar no link "Ver Convites" no menu principal. Será exibida uma lista de convites para o usuário, com o nome do projeto a ser convidado. Depois, deve-se escolher a opção para rejeitar participar do projeto. Deve retornar a tela de acesso do usuário e exibir uma mensagem de sucesso na operação escolhida.													
Notas: <ol style="list-style-type: none">1. As ações que devem ser mostradas por convite são: "Aceitar Participar" e "Rejeitar Participar".													

Ver Participantes de um Projeto

ID: 012	Nome: Ver Participantes de um Projeto																					
Como um usuário, eu gostaria ver os participantes de um projeto que eu também participo/coordeno para que eu possa ver quem são eles e verificar as funções de cada um.																						
Tela 1: Seja bem-vindo, Dennis Silveira. Projetos que você coordena (como Scrum Master) <table border="1"><thead><tr><th>Projeto</th><th>Data Inicial</th><th>Data Final</th><th>Product Backlog</th><th>Estimativa</th><th>Sprints</th><th>Ações</th></tr></thead><tbody><tr><td>Arrecadação</td><td>01/01/2010</td><td>31/12/2013</td><td>0 Itens</td><td></td><td>0</td><td></td></tr><tr><td>Estimativa com Planning Poker</td><td>13/04/2012</td><td>10/07/2012</td><td>0 Itens</td><td></td><td>0</td><td></td></tr></tbody></table>		Projeto	Data Inicial	Data Final	Product Backlog	Estimativa	Sprints	Ações	Arrecadação	01/01/2010	31/12/2013	0 Itens		0		Estimativa com Planning Poker	13/04/2012	10/07/2012	0 Itens		0	
Projeto	Data Inicial	Data Final	Product Backlog	Estimativa	Sprints	Ações																
Arrecadação	01/01/2010	31/12/2013	0 Itens		0																	
Estimativa com Planning Poker	13/04/2012	10/07/2012	0 Itens		0																	
Tela 2: Lista de Participantes do Projeto YumeYo. <table border="1"><thead><tr><th>Participante</th><th>E-mail</th><th>Tipo</th></tr></thead><tbody><tr><td>Cicrano</td><td>cicrano@gmail.com</td><td>Membro do Scrum Team</td></tr><tr><td>Dennis Silveira</td><td>dwas@cin.ufpe.br</td><td>Membro do Scrum Team</td></tr><tr><td>Fulano</td><td>fulano@gmail.com</td><td>Scrum Master</td></tr></tbody></table> <input type="button" value="Voltar"/>		Participante	E-mail	Tipo	Cicrano	cicrano@gmail.com	Membro do Scrum Team	Dennis Silveira	dwas@cin.ufpe.br	Membro do Scrum Team	Fulano	fulano@gmail.com	Scrum Master									
Participante	E-mail	Tipo																				
Cicrano	cicrano@gmail.com	Membro do Scrum Team																				
Dennis Silveira	dwas@cin.ufpe.br	Membro do Scrum Team																				
Fulano	fulano@gmail.com	Scrum Master																				
Comportamento de Sucesso: 1. Clicar no link "Listar Participantes" de um dos projetos. Será exibida uma lista de participantes do projeto para o usuário.																						
Notas: 1. Listar por nome, e-mail e tipo de participação do responsável.																						

Remover Projeto

ID: 013	Nome: Remover Projeto																					
Como um coordenador, eu gostaria remover um projeto que não tenha mais finalidade em mantê-lo.																						
Seja bem-vindo, Dennis Silveira. Projetos que você coordena (como Scrum Master) <table border="1"><thead><tr><th>Projeto</th><th>Data Inicial</th><th>Data Final</th><th>Product Backlog</th><th>Estimativa</th><th>Sprints</th><th>Ações</th></tr></thead><tbody><tr><td>Arrecadação</td><td>01/01/2010</td><td>31/12/2013</td><td>0 Itens</td><td></td><td>0</td><td></td></tr><tr><td>Estimativa com Planning Poker</td><td>13/04/2012</td><td>10/07/2012</td><td>0 Itens</td><td></td><td>0</td><td></td></tr></tbody></table>		Projeto	Data Inicial	Data Final	Product Backlog	Estimativa	Sprints	Ações	Arrecadação	01/01/2010	31/12/2013	0 Itens		0		Estimativa com Planning Poker	13/04/2012	10/07/2012	0 Itens		0	
Projeto	Data Inicial	Data Final	Product Backlog	Estimativa	Sprints	Ações																
Arrecadação	01/01/2010	31/12/2013	0 Itens		0																	
Estimativa com Planning Poker	13/04/2012	10/07/2012	0 Itens		0																	
Comportamento de Sucesso: 1. Clicar no link "Remover Projeto" de um dos projetos. Será exibida uma tela pedindo confirmação da operação. Será exibida uma mensagem sobre a remoção.																						

Exibir Product Backlog

ID: 014	Nome: Exibir Product Backlog																												
<p>Como um usuário, eu gostaria ver o <i>Product Backlog</i> de um dos projetos dos quais participo, seja como coordenador ou como participante para que eu possa ver informações dos itens/tarefas a serem implementados nele.</p>																													
<p>Tela 1:</p> <p>Seja bem-vindo, Dennis Silveira.</p> <p>Projetos que você coordena (como Scrum Master)</p> <table border="1"><thead><tr><th>Projeto</th><th>Data Inicial</th><th>Data Final</th><th>Product Backlog</th><th>Estimativa</th><th>Sprints</th><th>Ações</th></tr></thead><tbody><tr><td>Arrecadação</td><td>01/01/2010</td><td>31/12/2013</td><td>0 Itens</td><td></td><td>0</td><td></td></tr><tr><td>Estimativa com Planning Poker</td><td>13/04/2012</td><td>10/07/2012</td><td>0 Itens</td><td></td><td>0</td><td></td></tr></tbody></table>		Projeto	Data Inicial	Data Final	Product Backlog	Estimativa	Sprints	Ações	Arrecadação	01/01/2010	31/12/2013	0 Itens		0		Estimativa com Planning Poker	13/04/2012	10/07/2012	0 Itens		0								
Projeto	Data Inicial	Data Final	Product Backlog	Estimativa	Sprints	Ações																							
Arrecadação	01/01/2010	31/12/2013	0 Itens		0																								
Estimativa com Planning Poker	13/04/2012	10/07/2012	0 Itens		0																								
<p>Tela 2:</p> <p>Product Backlog do Projeto Estimativa com Planning Poker</p> <table border="1"><thead><tr><th>ID</th><th>Item</th><th>Prioridade</th><th>Estimativa</th><th>Data de Conclusão</th><th>Sprint</th><th>Ações</th></tr></thead><tbody><tr><td>1</td><td>Cadastrar Usuário</td><td>10</td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td>Efetuar Login</td><td>10</td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td>Efetuar Logout</td><td>10</td><td></td><td></td><td></td><td></td></tr></tbody></table>		ID	Item	Prioridade	Estimativa	Data de Conclusão	Sprint	Ações	1	Cadastrar Usuário	10					2	Efetuar Login	10					3	Efetuar Logout	10				
ID	Item	Prioridade	Estimativa	Data de Conclusão	Sprint	Ações																							
1	Cadastrar Usuário	10																											
2	Efetuar Login	10																											
3	Efetuar Logout	10																											
<p>Comportamento de Sucesso:</p> <ol style="list-style-type: none">1. Clicar no link "Remover Projeto" de um dos projetos. Vão ser listados os itens que fazem parte de um <i>Product Backlog</i> do projeto, com informações básicas como: nome do item, prioridade, estimativa, data de conclusão e iteração da qual o item participa.																													
<p>Notas:</p> <ol style="list-style-type: none">1. As ações que devem ser mostradas por item do <i>Product Backlog</i> são: "Ver Item", "Alterar Item" e "Remover Item".																													

Cadastrar Item

ID: 015

Nome: Cadastrar Item

Como um coordenador, eu gostaria de poder cadastrar itens de um projeto do qual faço parte, para que eu possa acompanhá-los e estimá-los no sistema.

Tela 1:

The screenshot shows two parts of the interface. On the left is a 'Menu' with the following items: '> Alterar Dados', '> Alterar Projeto', '> Novo Item' (circled in red), '> Lista de Projetos', '> Ver convites', and '> Sair'. On the right is a 'Product Backlog' table with the following data:

ID	Item
1	Cadastrar Usuário
2	Efetuar Login
3	Efetuar Logout

Tela 2:

Cadastrar novo item

The screenshot shows a form titled 'Cadastrar novo item' with the following fields: 'ID:' (text input), 'Nome:' (text input), 'Prioridade:' (text input), 'Estória:' (text area), and 'Notas:' (text area). At the bottom of the form are two buttons: 'Cadastrar' and 'Voltar'.

Comportamento de Sucesso:

1. Clicar no link "Novo Item" no menu principal. Preencher corretamente todos os campos no formato indicado e clicar em "Cadastrar". Deve retornar a tela do *Product Backlog* do projeto e exibir uma mensagem de sucesso.

Comportamento de Erro:

1. Não preencher pelo menos um dos campos ou preencher num formato que não é o indicado e clicar em "Cadastrar". Deve exibir uma indicação de erro nos campos não preenchidos.

Notas:

1. Os campos devem ser: ID, nome, prioridade, estória e notas.

Ver Item

ID: 016

Nome: Ver Item

Como um usuário, eu gostaria de poder ver detalhes de um item de um *Product Backlog* de um projeto do qual faço parte, para que eu possa ter maiores informações sobre ele.

Tela 1:

Product Backlog do Projeto Estimativa com Planning Poker

ID	Item	Prioridade	Estimativa	Data de Conclusão	Sprint	Ações
1	Cadastrar Usuário	10				  
2	Efetuar Login	10				  
3	Efetuar Logout	10				  

Tela 2:

Item Cadastrar Usuário do Projeto Estimativa com Planning Poker

ID:

Nome:

Prioridade:

Estimativa:

Estória:

Notas:

Concluído:

Sprint:

Comportamento de Sucesso:

1. Ao clicar no link "Ver Informações do Item" de algum item, será possível ver informações mais específicas de um item.

Notas:

1. Os campos visualizados devem ser: ID, nome, prioridade, estimativa, estória, notas, concluído, data de conclusão (se já estiver concluído) e a *Sprint* (se houver alguma relacionada ao item).

Alterar Item

ID: 017

Nome: Alterar Item

Como um usuário, eu gostaria de poder alterar um item de algum projeto do qual faço parte, para que eu possa modificar algumas informações sobre ele.

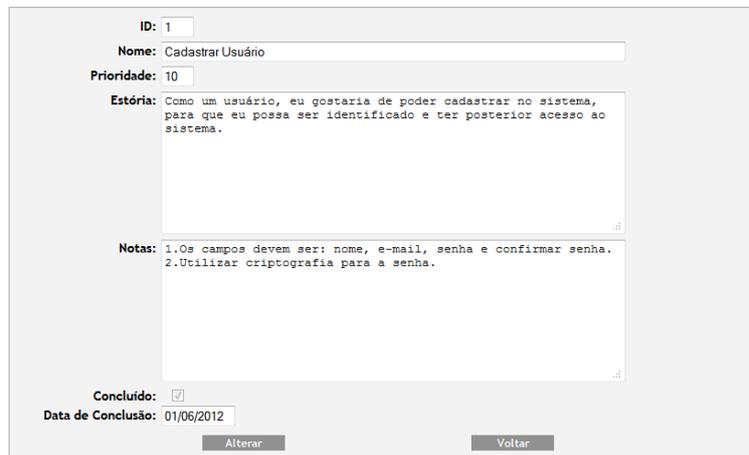
Tela 1:

Product Backlog do Projeto Estimativa com Planning Poker

ID	Item	Prioridade	Estimativa	Data de Conclusão	Sprint	Ações
1	Cadastrar Usuário	10				
2	Efetuar Login	10				
3	Efetuar Logout	10				

Tela 2:

Alterar Item Cadastrar Usuário do Projeto Estimativa com Planning Poker



The screenshot shows a form for editing a user registration item. The fields are: ID (1), Nome (Cadastrar Usuário), Prioridade (10), Estória (Como um usuário, eu gostaria de poder cadastrar no sistema, para que eu possa ser identificado e ter posterior acesso ao sistema.), and Notas (1.Os campos devem ser: nome, e-mail, senha e confirmar senha. 2.Utilizar criptografia para a senha.). There are checkboxes for 'Concluído' and a date field for 'Data de Conclusão' (01/06/2012). At the bottom, there are 'Alterar' and 'Voltar' buttons.

Comportamento de Sucesso:

1. Clicar no link "Alterar Item" de algum item. Preencher corretamente todos os campos no formato indicado e clicar em "Cadastrar". Deve retornar a tela de acesso do usuário e exibir uma mensagem de sucesso.

Comportamento de Erro:

1. Não preencher pelo menos um dos campos (que não seja o "Concluído") ou preencher num formato inadequado e clicar em "Alterar". Deve exibir uma indicação de erro nesses campos.
2. Preencher o campo "Concluído" e não preencher o campo "Data de Conclusão" ou preencher no formato inadequado. Deve exibir uma indicação de erro nesse campo.

Notas:

1. Os campos devem ser: ID, nome, prioridade, estória, notas, concluído, data de conclusão e *Sprint* (se houver alguma cadastrada no projeto).

Remover Item

ID: 018

Nome: Remover Item

Como um coordenador, eu gostaria remover um item que não tenha mais finalidade em mantê-lo no projeto.

Product Backlog do Projeto Estimativa com Planning Poker

ID	Item	Prioridade	Estimativa	Data de Conclusão	Sprint	Ações
1	Cadastrar Usuário	10				  
2	Efetuar Login	10				  
3	Efetuar Logout	10				  

Comportamento de Sucesso:

1. Clicar no link "Remover Item" de um dos itens do *Product Backlog*. Será exibida uma tela pedindo confirmação da operação. Será exibida uma mensagem sobre a remoção.

Ver Gráfico de Acompanhamento

ID: 019

Nome: Ver Gráfico de Acompanhamento

Como um usuário, eu gostaria visualizar um gráfico de acompanhamento (*Burndown Chart*) para que eu verificasse a evolução do projeto ao longo de semanas ou *Sprints*.

Tela 1:

Seja bem-vindo, Dennis Silveira.

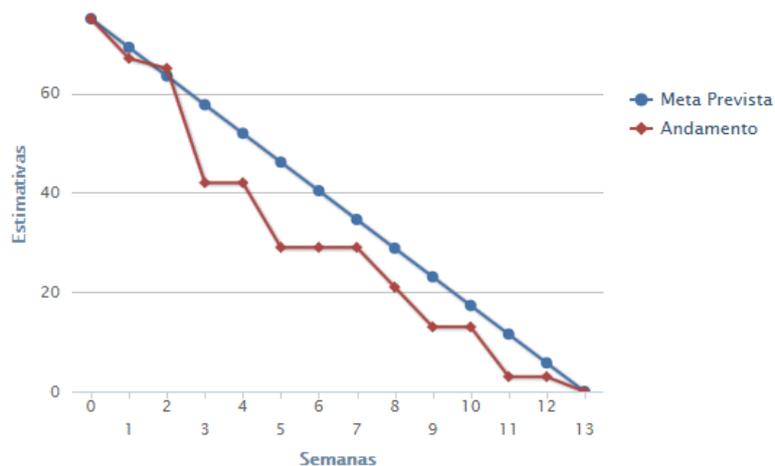
Projetos que você coordena (como Scrum Master)

Projeto	Data Inicial	Data Final	Product Backlog	Estimativa	Sprints	Ações
Arrecadação	01/01/2010	31/12/2013	0 Itens		0	
Estimativa com Planning Poker	13/04/2012	10/07/2012	0 Itens		0	

Tela 2:

Selecione um gráfico:

Gráfico Burndown Chart de Semanas
Evolução do projeto



Comportamento de Sucesso:

1. Caso já os itens do *Product Backlog* do projeto já tenham sido estimados, clicar no link "Visualizar Gráfico de Acompanhamento" de um dos projetos. Será exibida uma tela pedindo para selecionar a evolução em semanas ou *Sprints*.

Comportamento de Erro:

1. É preciso haver estimado os itens do projeto para visualizar o gráfico. Caso contrário, isso não será possível.

Notas:

1. Necessário o uso de uma biblioteca que gera gráficos para poder desenhá-los de acordo com os parâmetros inseridos, correspondentes aos valores das estimativas.

Iniciar Sessão Planning Poker

ID: 020

Nome: Iniciar Sessão Planning Poker

Como um coordenador, eu gostaria de iniciar uma sessão de um projeto para que os participantes possam jogar *Planning Poker*.

Tela 1:

Product Backlog do Projeto Estimativa com Planning Poker

ID	Item	Prioridade	Estimativa	Data de Conclusão	Sprint	Ações
1	Cadastrar Usuário	10				  
2	Efetuar Login	10				  
3	Efetuar Logout	10				  

Iniciar Planning Poker

Tela 2:

Nome: Cadastrar Usuário
Estória: Como um usuário, eu gostaria de poder cadastrar no sistema, para que eu possa ser identificado e ter posterior acesso ao sistema.
Prioridade: 10
Notas: 1.Os campos devem ser: nome, e-mail, senha e confirmar senha. 2.Utilizar criptografia para a senha.

Coordenador: Dennis Silveira
Jogadores:

Rodada 1:

Participante: Dennis Silveira

Estimativa:

Mensagem

Comportamento de Sucesso:

1. Clicar no botão "Iniciar Planning Poker", que aparece no *Product Backlog* de um dos projetos. Será exibida uma tela com o início da sessão, com as informações da estória, a tela de estimativas, a tela de configurações do participante e o menu de *chat* para os participantes trocarem informações.

Notas:

1. É preciso elaborar uma sessão compartilhada, de modo que somente os usuários autorizados vejam as mesmas informações e esses dados sejam frequentemente atualizados.
2. O botão "Iniciar Planning Poker" somente será exibido ao coordenador do projeto.
3. Várias sessões podem ser criadas para o mesmo projeto, porém somente a última é a que vai prevalecer.

Entrar numa Sessão Planning Poker

ID: 021

Nome: Entrar numa Sessão Planning Poker

Como um participante, eu gostaria de entrar numa sessão de um projeto para que eu possa jogar *Planning Poker*.

Tela 1:

Product Backlog do Projeto Estimativa com Planning Poker

ID	Item	Prioridade	Estimativa	Data de Conclusão	Sprint	Ações
1	Cadastrar Usuário	10				  
2	Efetuar Login	10				  
3	Efetuar Logout	10				  

Entrar no Planning Poker

Tela 2:

Nome: Cadastrar Usuário
Estória: Como um usuário, eu gostaria de poder cadastrar no sistema, para que eu possa ser identificado e ter posterior acesso ao sistema.
Prioridade: 10
Notas: 1.Os campos devem ser: nome, e-mail, senha e confirmar senha. 2.Utilizar criptografia para a senha.

Coordenador: Dennis Silveira
Jogadores: Fulano, Cicrano

Rodada 1:

Fulano	Cicrano
	

Participante: Cicrano

0	1/2	1	2	3	5	8	13	20	40	100	?	
---	-----	---	---	---	---	---	----	----	----	-----	---	---

Escolher Carta
Sair da Sessão

Mensagem

[Cicrano] - também cheguei. Vamos começar agora
[Dennis Silveira] - Vamos lá.
[Fulano] - pronto! Cheguei, vamos começar.

Comportamento de Sucesso:

1. Clicar no botão "Entrar no Planning Poker", que aparece no *Product Backlog* de um dos projetos. Esse botão será disponibilizado somente quando o coordenador iniciar uma sessão. Será exibida uma tela com o início da sessão, com as informações da estória, a tela de estimativas, a tela de configurações do participante (baralho e opção pra sair da sessão) e o menu de *chat* para os participantes trocarem informações.

Escolher Carta

ID: 022

Nome: Escolher Carta

Como um participante, eu gostaria escolher uma carta do meu baralho para que eu possa definir minha estimativa para um item avaliado.

Nome: Cadastrar Usuário
Estória: Como um usuário, eu gostaria de poder cadastrar no sistema, para que eu possa ser identificado e ter posterior acesso ao sistema.
Prioridade: 10
Notas: 1. Os campos devem ser: nome, e-mail, senha e confirmar senha. 2. Utilizar criptografia para a senha.

Coordenador: Dennis Silveira
Jogadores: Fulano, Cicrano

Rodada 1:

Fulano Cicrano

PLANNING POKER PLANNING POKER

Participante: Cicrano

0 1/2 1 2 3 5 8 13 20 40 100 ? ☕

Escolher Carta
Sair da Sessão

Mensagem

[Cicrano] - também cheguei. Vamos começar agora
[Dennis Silveira] - Vamos lá.
[Fulano] - pronto! Cheguei, vamos começar.

Comportamento de Sucesso:

1. Dentro da tela de configurações do participante, será possível clicar numa das cartas com a estimativa desejada e confirmar clicando no botão "Escolher Carta". A escolha do participante será mostrada na tela das estimativas quando o coordenador autorizar.

Notas:

1. As cartas do baralho do participante devem ter os 13 valores seguintes: '0', '1/2', '1', '2', '3', '5', '8', '13', '20', '40', '100', '?' e a xícara de café.
2. O sistema deve armazenar um valor numérico para representar as cartas da xícara de café e do '?'.

Mostrar Cartas

ID: 023

Nome: Mostrar Cartas

Como um coordenador, eu gostaria de exibir as cartas dos participantes quando todos eles as confirmarem.

<p>Nome: Cadastrar Usuário Estória: Como um usuário, eu gostaria de poder cadastrar no sistema, para que eu possa ser identificado e ter posterior acesso ao sistema. Prioridade: 10 Notas: 1.Os campos devem ser: nome, e-mail, senha e confirmar senha. 2.Utilizar criptografia para a senha.</p>	<p>[Cicrano] - também cheguei. Vamos começar agora [Dennis Silveira] - Vamos lá. [Fulano] - pronto! Cheguei, vamos começar.</p>				
<p>Coordenador: Dennis Silveira Jogadores: Fulano, Cicrano</p> <p>Rodada 1:</p> <table border="1"><thead><tr><th>Fulano</th><th>Cicrano</th></tr></thead><tbody><tr><td>40</td><td>8</td></tr></tbody></table>	Fulano	Cicrano	40	8	
Fulano	Cicrano				
40	8				
<p>Participante: Dennis Silveira</p> <p>Estimativa: <input type="text"/> Definir Estimativa</p> <p>Mostrar Cartas Realizar Nova Rodada Finalizar Sessão</p>	<p>Mensagem</p> <p><input type="text"/></p>				

Comportamento de Sucesso:

1. Dentro da tela de configurações do coordenador, será possível visualizar o botão “Mostrar Cartas” para exibir as estimativas. Ao clicá-lo, serão viradas as cartas de cada usuário.

Notas:

1. Habilitar o botão somente quando todos os jogadores escolherem suas cartas.

Definir Estimativa para um Item

ID: 024 Nome: Definir Estimativa para um Item

Como um coordenador, eu gostaria de definir uma estimativa para um item quando todos os participantes confirmarem suas estimativas.

<p>Nome: Cadastrar Usuário Estória: Como um usuário, eu gostaria de poder cadastrar no sistema, para que eu possa ser identificado e ter posterior acesso ao sistema. Prioridade: 10 Notas: 1.Os campos devem ser: nome, e-mail, senha e confirmar senha. 2.Utilizar criptografia para a senha.</p>	<p>[Cicrano] - também cheguei. Vamos começar agora [Dennis Silveira] - Vamos lá. [Fulano] - pronto! Cheguei, vamos começar.</p>				
<p>Coordenador: Dennis Silveira Jogadores: Fulano, Cicrano</p> <p>Rodada 1:</p> <table border="1"><tr><td>Fulano</td><td>Cicrano</td></tr><tr><td>8</td><td>5</td></tr></table>	Fulano	Cicrano	8	5	
Fulano	Cicrano				
8	5				
<p>Participante: Dennis Silveira</p> <p>Estimativa: <input type="text"/> Definir Estimativa</p> <p>Mostrar Cartas Realizar Nova Rodada Finalizar Sessão</p>	<p>Mensagem</p>				

Comportamento de Sucesso:

1. Dentro da tela de configurações do coordenador, será possível que ele estabeleça um valor para inserir a estimativa definitiva da estória. Deve preencher corretamente o campo no formato de números e clicar em "Definir Estimativa". Deve exibir uma mensagem de sucesso e passar automaticamente ao próximo item. Aos participantes, será exibida a escolha do coordenador.

Comportamento de Erro:

1. Preencher incorretamente o campo no formato de números e clicar em "Ok". Deve exibir uma mensagem de erro.
2. Não preencher o campo e clicar em "Ok". Deve exibir uma mensagem de erro.

Notas:

1. Habilitar os campos quando todos os jogadores exibirem suas cartas.

Realizar Nova Rodada

ID: 025

Nome: Realizar Nova Rodada

Como um coordenador, eu gostaria realizar uma nova rodada de estimativa para um item para o caso de grande diferença de estimativas dos participantes até que estas tendam a um valor próximo.

Tela 1:

Nome: Cadastrar Usuário
Estória: Como um usuário, eu gostaria de poder cadastrar no sistema, para que eu possa ser identificado e ter posterior acesso ao sistema.
Prioridade: 10
Notas: 1.Os campos devem ser: nome, e-mail, senha e confirmar senha. 2.Utilizar criptografia para a senha.

Coordenador: Dennis Silveira
Jogadores: Fulano, Cicrano

Rodada 1:

Fulano	Cicrano
13	1

Participante: Dennis Silveira

Estimativa:

Mensagem

[Dennis Silveira] - eu sei, vou ter que realizar uma nova rodada para chegar a um acordo nesse impasse

[Cicrano] - houve uma diferença muito grande de valores nessa rodada

[Cicrano] - também cheguei. Vamos começar agora

[Dennis Silveira] - Vamos lá.

[Fulano] - pronto! Cheguei, vamos começar.

Tela 2:

Nome: Cadastrar Usuário
Estória: Como um usuário, eu gostaria de poder cadastrar no sistema, para que eu possa ser identificado e ter posterior acesso ao sistema.
Prioridade: 10
Notas: 1.Os campos devem ser: nome, e-mail, senha e confirmar senha. 2.Utilizar criptografia para a senha.

Jogadores: Fulano, Cicrano

Rodada 1:

Fulano	Cicrano
13	1

Rodada 2:

Fulano	Cicrano
PLANNING POKER	PLANNING POKER

Participante: Dennis Silveira

Estimativa:

Mensagem

[Dennis Silveira] - eu sei, vou ter que realizar uma nova rodada para chegar a um acordo nesse impasse

[Cicrano] - houve uma diferença muito grande de valores nessa rodada

[Cicrano] - também cheguei. Vamos começar agora

[Dennis Silveira] - Vamos lá.

[Fulano] - pronto! Cheguei, vamos começar.

Comportamento de Sucesso:

1. Quando todos os participantes exibirem suas estimativas, será possível visualizar um botão para realizar uma nova rodada dentro da tela de configurações do coordenador. Ao clicá-lo, os participantes vão realizar uma nova rodada de *Planning Poker* sobre o item estimado.

Notas:

1. Habilitar o botão quando todos os jogadores exibirem suas cartas na rodada anterior.

Sair da Sessão

ID: 026

Nome: Sair da Sessão

Como um participante, eu gostaria sair de uma sessão no momento que eu achasse conveniente.

The screenshot displays a poker game interface. At the top left, there is a section for user registration with the following text: 'Nome: Cadastrar Usuário', 'Estória: Como um usuário, eu gostaria de poder cadastrar no sistema, para que eu possa ser identificado e ter posterior acesso ao sistema.', 'Prioridade: 10', and 'Notas: 1. Os campos devem ser: nome, e-mail, senha e confirmar senha. 2. Utilizar criptografia para a senha.' Below this, the coordinator is identified as 'Dennis Silveira' and the players as 'Fulano, Cicrano'. The main area shows 'Rodada 1:' with two players, 'Fulano' and 'Cicrano', each with a 'PLANNING POKER' button. At the bottom, the player 'Cicrano' is active, and a row of buttons shows the game state: '0', '1/2', '1', '2', '3', '5', '8', '13', '20', '40', '100', '?', and a button with a 'C' icon. A red circle highlights two buttons: 'Escalhe Carta' and 'Sair da Sessão'. On the right side, there is a chat window with messages from '[Cicrano] - também cheguei. Vamos começar agora', '[Dennis Silveira] - Vamos lá', and '[Fulano] - pronto! Cheguei, vamos começar.'. At the bottom right, there is a 'Mensagem' input field.

Comportamento de Sucesso:

1. Dentro da tela de configurações do coordenador, será possível visualizar o botão “Sair da Sessão”. Ao clicar nele, o usuário sai da sessão e retorna a tela principal do usuário com uma mensagem de confirmação. Aos outros participantes, será exibida uma mensagem informando a saída do usuário.

Finalizar Sessão

ID: 027

Nome: Finalizar Sessão

Como um coordenador, eu gostaria finalizar uma sessão quando todos os itens do projeto forem estimados.

<p>Nome: Cadastrar Usuário Estória: Como um usuário, eu gostaria de poder cadastrar no sistema, para que eu possa ser identificado e ter posterior acesso ao sistema. Prioridade: 10 Notas: 1.Os campos devem ser: nome, e-mail, senha e confirmar senha. 2.Utilizar criptografia para a senha.</p>	<p>[Cicrano] - também cheguei. Vamos começar agora [Dennis Silveira] - Vamos lá. [Fulano] - pronto! Cheguei, vamos começar.</p>				
<p>Coordenador: Dennis Silveira Jogadores: Fulano, Cicrano</p> <p>Rodada 1:</p> <table border="1"><thead><tr><th>Fulano</th><th>Cicrano</th></tr></thead><tbody><tr><td>8</td><td>5</td></tr></tbody></table>	Fulano	Cicrano	8	5	
Fulano	Cicrano				
8	5				
<p>Participante: Dennis Silveira</p> <p>Estimativa: <input type="text"/> <input type="button" value="Definir Estimativa"/></p> <p><input type="button" value="Mostrar Cartas"/> <input type="button" value="Realizar Nova Rodada"/> <input type="button" value="Finalizar Sessão"/></p>	<p>Mensagem</p> <input type="text"/>				

Comportamento de Sucesso:

- Quando forem definidas as estimativas de todos os itens, será possível visualizar um botão com o nome "Finalizar Sessão" para o coordenador encerrar aquela sessão ao clicá-lo.

Notas:

- Ao finalizar uma sessão, não se deve permitir o uso dela novamente por nenhum usuário.
- Todos os usuários sairão automaticamente dela.

Chat da Sessão

ID: 028

Nome: Chat da Sessão

Como um usuário, eu gostaria ter acesso a um *chat* numa sessão para que eu possa trocar informações com outros participantes do projeto enquanto realizamos as estimativas. Gostaria também de ver o histórico de mensagens enviadas.

Nome: Cadastrar Usuário
Estória: Como um usuário, eu gostaria de poder cadastrar no sistema, para que eu possa ser identificado e ter posterior acesso ao sistema.
Prioridade: 10
Notas: 1.Os campos devem ser: nome, e-mail, senha e confirmar senha. 2.Utilizar criptografia para a senha.

Jogadores: Fulano, Cicrano

Rodada 1:

Fulano	Cicrano
13	1

Rodada 2:

Fulano	Cicrano
PLANNING POKE	PLANNING POKE

Participante: Dennis Silveira

Estimativa:

Chat:

- [Dennis Silveira] - eu sei, vou ter que realizar uma nova rodada para chegar a um acordo nesse impasse
- [Cicrano] - houve uma diferença muito grande de valores nessa rodada
- [Cicrano] - também cheguei. Vamos começar agora
- [Dennis Silveira] - Vamos lá.
- [Fulano] - pronto! Cheguei, vamos começar.

Mensagem:

Comportamento de Sucesso:

1. Preencher o campo mensagem com alguma informação e clicar em "Enter". A mensagem será exibida na lista para todos os outros usuários da sessão visualizar.

Notas:

1. As mensagens só possuem validade para essa sessão. Mesmo se for criada uma nova sessão para o projeto em questão, o histórico das mensagens da sessão anterior não aparece na nova.

Apêndice B – Backlog de Requisitos Não Funcionais

Este apêndice apresenta o *Backlog* de requisitos não funcionais da ferramenta. Ele contém informações sobre as características do sistema. Assim como as funcionalidades do *Product Backlog* foram descritos anteriormente através de estórias, é possível fazer o mesmo para descrever os requisitos funcionais. Assim, as estórias dos requisitos não funcionais foram descritas na forma de cartões, conforme mostrado na Figura 3 do Capítulo 3 deste documento.

Os requisitos não funcionais deste projeto são os seguintes:

Performance

ID: 001	Nome: Performance
Como um usuário, eu gostaria que o sistema apresentasse uma performance aceitável, de modo que o tempo de uma solicitação seja o mínimo possível.	

Usabilidade

ID: 002	Nome: Usabilidade
Como um usuário, eu gostaria que o sistema fosse intuitivo e fácil de usar com interface simples, exibindo as informações corretamente e com mensagens de erro claras e compreensíveis.	

Disponibilidade

ID: 003	Nome: Disponibilidade
Como um usuário, eu gostaria que o sistema permanecesse o maior tempo possível disponível para o acesso e para execução das operações e que evitasse filas de espera pela indisponibilidade.	

Manual de Instalação

ID: 004	Nome: Manual de Instalação
Como um usuário, eu gostaria que o sistema possuísse um manual de instalação, para que eu encontrasse todas as informações necessárias, em caso de dúvidas, no momento de instalá-lo e configurá-lo.	

Manual do Usuário

ID: 005	Nome: Manual do Usuário
Como um usuário, eu gostaria de um manual do usuário que fosse compressível e objetivo, para que eu possa encontrar nele todas as informações necessárias sobre o uso do sistema.	

Escalabilidade

ID: 006	Nome: Escalabilidade
Como um usuário, eu gostaria que o sistema fosse bom o suficiente para comportar uma razoável quantidade de usuários simultaneamente que estejam executando as tarefas dele e ao mesmo tempo fosse capaz de trabalhar de maneira uniforme.	

Integridade

ID: 007	Nome: Integridade
Como um usuário, eu gostaria que as informações que eu inserisse no sistema estejam corretas e que quando eu realizasse consultas, esses dados estejam consistentes com o que eu havia inserido antes.	

Privacidade

ID: 008	Nome: Privacidade
Como um usuário, eu gostaria que as informações que eu inserisse no sistema fossem apenas visualizadas ou alteradas por pessoas autorizadas por mim, como por exemplo, na edição de um dos itens de um <i>Product Backlog</i> , ou no acesso a sessão do <i>Planning Poker</i> .	

Legibilidade do Código

ID: 009	Nome: Legibilidade do Código
Como um desenvolvedor, eu gostaria o que o código do sistema fosse o mais simples e legível possível para que eu e outros desenvolvedores possamos compreendê-lo, pois a proposta é que este sistema possa ser continuado no futuro. Para isso, é importante colocar comentários, dividir o sistema em camadas e elaborar uma boa documentação.	