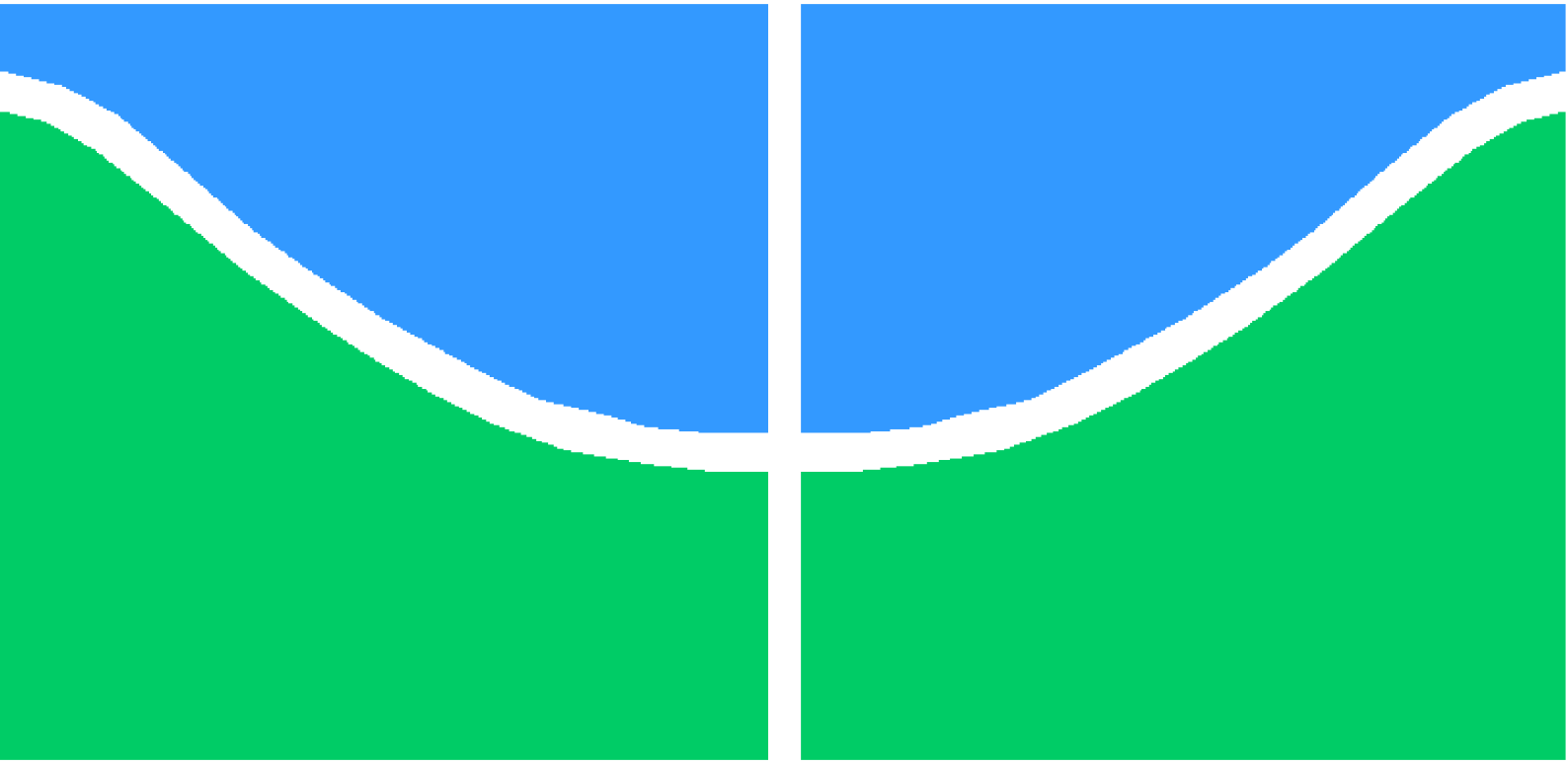

Universidade de Brasília – UnB
Faculdade UnB Gama – FGA
Engenharia de Software

Factbox: Uma Ferramenta para Gerência dos Requisitos

Autor: Phelipe Wener Pereira Mota
Orientador: Prof. Dr. Maurício Serrano

Brasília, DF
2019

Phelipe Wener Pereira Mota

Factbox: Uma Ferramenta para Gerência dos Requisitos

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

Universidade de Brasília – UnB

Faculdade UnB Gama – FGA

Orientador: Prof. Dr. Maurício Serrano

Coorientador: Prof[a] Dra. Milene Serrano

Brasília, DF

2019

Phelipe Wener Pereira Mota

Factbox: Uma Ferramenta para Gerência dos Requisitos/ Phelipe Wener Pereira Mota. – Brasília, DF, 2019-

109 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Dr. Maurício Serrano

Trabalho de Conclusão de Curso – Universidade de Brasília – UnB
Faculdade UnB Gama – FGA , 2019.

1. Palavra-chave01. 2. Palavra-chave02. I. Prof. Dr. Maurício Serrano. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Factbox: Uma Ferramenta para Gerência dos Requisitos

CDU 02:141:005.6

Phelipe Wener Pereira Mota

Factbox: Uma Ferramenta para Gerência dos Requisitos

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

Trabalho aprovado. Brasília, DF, 08 de Julho de 2019 – Data da aprovação do trabalho:

Prof. Dr. Maurício Serrano
Orientador

Profª Dra. Milene Serrano
Coorientador

Prof. Me. Giovanni Almeida Santos
Convidado 1

Brasília, DF
2019

Aos sonhadores, pois sem sonhos não pode haver um amanhã melhor.

Agradecimentos

Agradeço aos meus pais pelo apoio; à minha mãe, que com inefáveis sentimentos me deu forças para essa longa jornada; ao meu pai, por ter me ensinado grandes valores, mostrando os estudos como estrada para uma vida de verdadeira riqueza.

Agradeço aos meus orientadores Maurício Serrano e Milene Serrano por terem me ajudado no amadurecimento e na expressividade das minhas ideias, fazendo-me enxergar mais distante, e expandindo, mesmo antes desse trabalho, a minha visão de Engenharia de Software. Agradeço ao professor Hilmer Neri pelas inquietações que me colocaram na trilha de grandes reflexões.

"Os filósofos têm apenas interpretado o mundo de maneiras diferentes; a questão, porém, é transformá-lo." (Karl Marx)

Resumo

A Engenharia de Software é uma área abrangente que envolve investigações em diferentes âmbitos do conhecimento. Dentre suas demandas, há o Gerenciamento, seja de artefatos, equipe, riscos, custos ou outros. O Gerenciamento de Artefatos pode ser realizado em diferentes níveis de abstração (da camada de negócio ao código), e é particularmente relevante o Gerenciamento dos Requisitos, visando maior conformidade entre o que o domínio demanda e a solução computacional proposta. Apesar dessa relevância, há uma carência de suportes tecnológicos capazes de apoiar essa área.

Diante do exposto, esse Trabalho de Conclusão de Curso procura contribuir com esse *gap* tecnológico, oferecendo uma ferramenta *online* para Gerência dos Requisitos. Para melhor condução da Gerência dos Requisitos, há necessidade de atuação em duas vertentes: Rastreabilidade e Versionamento. Portanto, a solução computacional oferecida atua considerando ambas essas vertentes. Visando a avaliação da solução, fez-se o uso de iterações na análise dos resultados obtidos, aplicando ciclos de pesquisa-ação, documentados via uma abordagem híbrida, qualitativa e quantitativa. Ressalta-se, por fim, que a ferramenta é *open-source*.

Palavras-chaves: Engenharia de Requisitos, Gerenciamento dos Requisitos, Rastreabilidade, Versionamento, Ferramenta *online* e *open-source*.

Abstract

Software Engineering is an comprehensive area that makes investigations in different spheres of knowledge. Among its demands, there is the Management, that might be of artifacts, team, risks, costs among others. Artifact Management can be performed in different levels of abstraction (from the business layer to the code) and a particulary relevant demand is the Requirements Management, aiming for bigger conformity between the demands of the domain and the proposed computational solution. Besides this relevance, there is a absence of techonologic support capable of sponsor this area. Against the foregoing, this Final Course Paper intends to contribute to this technological gap, proposing a online tool for Requirementes Management. For a better conduction of Requirements Management, there is a need to actuate in two branches: Traceability and Versioning. Therefore, the computational solution offered acts in both these branches. Aiming for the evaluation of the solution, it is suggested the use of iterations in the analysis of he acquired data, applying cicles of research-action, accompanied by a hybrid approach, qualitative and quantitative. Finally, it should be noted that the tool is open source.

Keywords: Requirements Engineering, Requirements Management, Traceability, Versioning, Online Tool and Open-Source.

Lista de ilustrações

Figura 1 – Representação dos principais focos de atuação e de interesse deste trabalho . . . 29
Figura 2 – Processo de criação do Rich Picture, traduzido a partir de (BERG, 2015) 34
Figura 3 – Rich Picture da solução proposta no trabalho . . . 35
Figura 4 – Seleção metodológica . . . 44
Figura 5 – Fluxo de atividades do trabalho . . . 46
Figura 6 – Processo de desenvolvimento . . . 48
Figura 7 – Itens do backlog da *Sprint* . . . 49
Figura 8 – Etapas da pesquisa-ação. . . . 50
Figura 9 – Cronograma de atividades do trabalho . . . 52
Figura 10 – Exemplo de matriz de Rastreabilidade do Helix . . . 56
Figura 11 – Exemplo de matriz de Rastreabilidade do Orcanos . . . 56
Figura 12 – Exemplo de Rastreabilidade do Jira . . . 57
Figura 13 – Tela da *homepage* de um projeto mantido no Urutau . . . 57
Figura 14 – Diagrama UML de pontos da arquitetura do Urutau . . . 58
Figura 15 – Exemplo de grafo de Rastreabilidade . . . 60
Figura 16 – Kanban físico com o backlog da ferramenta Factbox . . . 61
Figura 17 – Histórias de Usuários mantidas na ferramenta Factbox. . . . 62
Figura 18 – Visão geral da ferramenta Factbox - Nível 1 da Arquitetura - Notação SADT . . . 63
Figura 19 – Visão arquitetural da ferramenta Factbox - Nível 2 - Principais Componentes . . . 63
Figura 20 – Visão arquitetural da ferramenta Factbox - Nível 3 - *Plugins* . . . 64
Figura 21 – Visão arquitetural da ferramenta Factbox - Nível 4 - Diagrama de Classes para os Modelos de Artefatos Suportados . . . 65
Figura 22 – Exemplo de método *show* no controlador de Artefatos, caso utilizasse herança . . . 65
Figura 23 – Método *show* no controlador de Artefatos, utilizando associações polimórficas . . . 66
Figura 24 – Visualização do histórico de versões de um artefato . . . 66
Figura 25 – Código de geração de Versionamento . . . 67
Figura 26 – Código de criação dos elos de Rastreabilidade . . . 68
Figura 27 – Menu para escolha de criação de artefatos. . . . 69
Figura 28 – Primeira pergunta do Questionário do primeiro ciclo de Pesquisa-ação . 73
Figura 29 – Segunda pergunta do Questionário do primeiro ciclo de Pesquisa-ação . 73
Figura 30 – Terceira pergunta do Questionário do primeiro ciclo de Pesquisa-ação . 74

Figura 31 – Terceira pergunta do Questionário do primeiro ciclo de Pesquisa-ação . 74
Figura 32 – Modificações de usabilidade realizadas durante a primeira *Sprint* na página de visualização de projeto 76
Figura 33 – Modificações de usabilidade realizadas durante a primeira *Sprint* no grafo de Rastreabilidade. 77
Figura 34 – Modificações de usabilidade realizadas durante a primeira *Sprint* no grafo de Rastreabilidade 78
Figura 35 – Primeira pergunta do Questionário do segundo ciclo de pesquisa-ação . 78
Figura 36 – Segunda pergunta do Questionário do segundo ciclo de pesquisa-ação . 79
Figura 37 – Terceira pergunta do Questionário do segundo ciclo de pesquisa-ação . 79
Figura 38 – Modificações de usabilidade realizadas durante a segunda *Sprint* na página de Rastreabilidade 82
Figura 39 – Modificações de usabilidade realizadas durante a segunda *Sprint*. Instalação de um *Breadcrumb*. 83
Figura 40 – Primeira pergunta do Questionário do terceiro ciclo de pesquisa-ação . 84
Figura 41 – Segunda pergunta do Questionário do terceiro ciclo de pesquisa-ação . 85
Figura 42 – Terceira pergunta do Questionário do terceiro ciclo de pesquisa-ação . . 85
Figura 43 – Quarta pergunta do Questionário do terceiro ciclo de pesquisa-ação . . 86
Figura 44 – Tela de História de usuário após modificações do terceiro ciclo. 88
Figura 45 – Gráfico resultante do projeto Habitica 93
Figura 46 – Gráfico resultante do projeto Go Deploy 95
Figura 47 – Gráfico resultante do projeto Twitter 96
Figura 48 – Questionário utilizado no primeiro ciclo de Pesquisa-ação 105
Figura 49 – Questionário utilizado no segundo ciclo de Pesquisa-ação 107
Figura 50 – Questionário utilizado no terceiro ciclo de Pesquisa-ação 109

Lista de tabelas

Tabela 1 – Resumo das ferramentas utilizadas no trabalho. 41
Tabela 2 – Tabela de critérios atendidos por ferramentas de proposta similar ao Factbox 56

Lista de abreviaturas e siglas

CASE	*Computer-Aided Software Engineering*
SADT	*Structured analysis and design technique*
XP	*Extreme Programming*
RoR	*Ruby on Rails*
HTTP	*Hypertext Transfer Protocol*
UnB	*Universidade de Brasília*
TCC	Trabalho de Conclusão de Curso

Sumário

1 INTRODUÇÃO 25

Introdução 25
1.1 Contextualização 25
1.2 Justificativa 26
1.3 Questão de Pesquisa 27
1.4 Objetivos 27
1.4.1 Objetivos Gerais 27
1.4.2 Objetivos Específicos 27
1.5 Organização do Trabalho 28

2 REFERENCIAL TEÓRICO 29
2.1 Contextualização da Engenharia de Requisitos 29
2.1.1 Engenharia de Software 30
2.1.2 Engenharia de Requisitos 30
2.2 Gerência dos Requisitos 32
2.2.1 Rastreabilidade 32
2.2.2 Versionamento 33
2.3 Especificações de Requisitos 33
2.3.1 Rich Picture 33
2.3.2 História de Usuário 35
2.4 Ferramentas de Requisitos 36
2.5 Resumo do Capítulo 37

3 SUPORTE TECNOLÓGICO 39
3.1 Desenvolvimento de Software 39
3.1.1 Kubuntu 17 39
3.1.2 Ruby on Rails 5 39
3.1.3 Vim 8 39
3.1.4 Git 2.14 39
3.1.5 Github 40
3.2 Pesquisa 40
3.2.1 LaTeX 3.14 40
3.2.2 Mendeley Desktop 1.17 40
3.2.3 TexMaker 5 40
3.3 Resumo do Capítulo 41

4 METODOLOGIA . . . 43
4.1 Classificação da Pesquisa . . . 43
4.2 Fluxo de Atividades . . . 45
4.3 Metodologia de Desenvolvimento . . . 47
4.4 Metodologia de Análise de Resultados . . . 50
4.5 Cronograma de Atividades . . . 51
4.6 Resumo do Capítulo . . . 51

5 FACTBOX . . . 55
5.1 Contextualização . . . 55
5.1.1 Trabalhos Relacionados . . . 55
5.1.2 Prova de Conceito . . . 57
5.2 Factbox: Uma Ferramenta de Gerência dos Requisitos . . . 59
5.2.1 Visão Geral da Ferramenta Factbox . . . 59
5.3 Detalhamento dos Requisitos da Ferramenta Factbox . . . 60
5.4 Visão Arquitetural da Ferramenta Factbox . . . 60
5.4.1 Nível 1 . . . 60
5.4.2 Nível 2 . . . 61
5.4.3 Nível 3 . . . 64
5.4.4 Nível 4 . . . 64
5.5 Versionamento e Rastreabilidade na Ferramenta Factbox . . . 66
5.5.1 Versionamento na Ferramenta Factbox . . . 66
5.5.2 Rastreabilidade na Ferramenta Factbox . . . 67
5.5.3 Manter Artefatos . . . 69
5.6 Resumo do Capítulo . . . 69

6 RESULTADOS OBTIDOS . . . 71
6.1 Público-Alvo . . . 71
6.2 Atividades da Pesquisa-Ação . . . 71
6.3 Primeiro Ciclo . . . 72
6.3.1 Coleta de dados, Análise e interpretação . . . 72
6.3.1.1 Questionário . . . 72
6.3.1.2 Observação Participante . . . 73
6.3.2 Plano de Ação . . . 75
6.3.3 Divulgação dos Resultados . . . 75
6.4 Segundo Ciclo . . . 78
6.4.1 Coleta de dados, Análise e interpretação . . . 78
6.4.1.1 Questionário . . . 78
6.4.1.2 Observação Participante . . . 80
6.4.2 Plano de Ação . . . 80

6.4.3 Divulgação dos Resultados 80
6.5 Terceiro Ciclo 84
6.5.1 Coleta de dados, Análise e interpretação 84
6.5.1.1 Questionário 84
6.5.1.2 Observação Participante 85
6.5.2 Plano de ação 86
6.5.3 Divulgação dos Resultados 87
6.6 Resumo do Capítulo 89

7 CONCLUSÃO 91
7.1 Objetivos Alcançados 91
7.2 Resultados dos Casos de Uso 91
7.3 Competências do Factbox 97
7.4 Fragilidades do Factbox 97
7.5 Trabalhos futuros 97

REFERÊNCIAS 99

APÊNDICES 103

APÊNDICE A – PRIMEIRO QUESTIONÁRIO 105

APÊNDICE B – SEGUNDO QUESTIONÁRIO 107

APÊNDICE C – TERCEIRO QUESTIONÁRIO 109

1 Introdução

Neste capítulo, serão descritos a contextualização, apresentando brevemente o tema foco desse trabalho; a justificativa, para elaboração da ferramenta *online*; a questão de pesquisa, na qual são concentrados os esforços e as contribuições do trabalho, e os objetivos geral e específicos. Por fim, é apresentada a organização dessa monografia.

1.1 Contextualização

De acordo com (BOURQUE; FAIRLEY, 2014), um requisito de software pode ser entendido como uma informação documentada, visando a identificação de um problema no mundo real, o qual pretende-se solucionar via software. Em (SAYÃO; CESAR; LEITE, 2005), têm-se atividades consideradas relevantes no que tange o conhecimento dos Requisitos de Software: processos de elicitação, modelagem, verificação e validação. Corroborando com esses autores, e alertando sobre a criticidade das tarefas associadas aos Requisitos de Software, em (NUSEIBEH; EASTERBROOK, 2000), faz-se menção à denominação Engenharia de Requisitos. Essa vista como âncora entre uma solução computacional e as atividades de desenvolvimento.

Objetivamente, para (SOMMERVILLE, 2007), os processos de Engenharia de Requisitos visam a manutenção e a criação de documentos de requisitos, cuja gestão desses documentos proporciona um acordo entre desenvolvedores e clientes, tal como justifica (SILVEIRA, 2006). Para lidar com os artefatos de requisitos, os desenvolvedores podem trabalhar com ferramentas disponíveis no mercado. Conforme apresentado em (CAETANO, 2008), algumas organizações produzem suas próprias ferramentas para suporte ao seu processo. Em (SILVEIRA, 2006), é referido que as ferramentas de gestão são úteis para elicitação, Versionamento, acompanhamento do estado e Rastreabilidade de requisitos.

De acordo com (NUSEIBEH; EASTERBROOK, 2000), é esperado manter a Rastreabilidade ao longo do projeto de um Software. Ainda segundo o autor, a Rastreabilidade contribui em termos de facilidade de leitura, navegabilidade, investigação e mudança na documentação dos requisitos. Reforçando essa ideia, em (SAYÃO; CESAR; LEITE, 2005), tem-se que a Rastreabilidade pode ser tratada em duas vertentes: (i) pré-rastreabilidade, a qual se preocupa em manter o rastro quanto às documentações/especificações de domínio que detalham as necessidades de negócio, as expectativas dos interessados e outros levantamentos inerentes ao contexto em investigação, estabelecendo a *baseline* de requisitos; e (ii) pós-rastreabilidade, a qual cria um elo entre os requisitos e os artefatos de projeto, implementação e testes.

O Versionamento também é um aspecto relevante. Segundo (CAETANO, 2008) e (SILVEIRA, 2006), cabe ao Versionamento: apoiar a captura e a atualização das especificações, considerando distintas fontes de informação, e, ainda, controlar o histórico dos requisitos. Ambas as ações podem ajudar na questão da Rastreabilidade, pois tornam possível acompanhar as relações de dependência através de uma base histórica.

Por fim, cabe ressaltar que as atividades de Rastreabilidade e Versionamento de requisitos estão associadas à Gerência dos Requisitos, sendo essa vista como um aspecto relevante para qualidade de software (SAYÃO; CESAR; LEITE, 2005). Diante do exposto, tornam-se relevantes esforços e contribuições que permitam auxiliar a Engenharia de Requisitos, mais especificamente em suas atividades fins. No caso do presente trabalho, procurou-se contribuir com as atividades de Rastreabilidade e Versionamento, focando em Gerência dos Requisitos. Justificativas, visando especificar/posicionar a área de atuação desse trabalho, são conferidas na próxima seção.

1.2 Justificativa

Com base em (SAYÃO; CESAR; LEITE, 2005), os processos de gerência e desenvolvimento de requisitos recomendados pelos modelos de maturidade são as primeiras etapas no sentido de aumentar a qualidade. Nesse contexto, a Engenharia de Requisitos aparece como uma área importante para organizações buscarem por qualidade.

Entretanto, o que se observa é uma dificuldade em lidar com os requisitos. Assim como ocorre com a Engenharia de Software, tal como colocado em (NUSEIBEH; EASTERBROOK, 2000), a Engenharia de Requisitos é uma área multidisciplinar, a qual, nos diversos estágios de desenvolvimento, demanda distintas técnicas e ferramentas em atendimento aos variados tipos de domínios.

Além disso, como observa (SILVEIRA, 2006), os Requisitos de Software mudam durante o desenvolvimento, e essas mudanças precisam ser documentadas e controladas. Essas dificuldades demandam devido suporte, em especial, ferramentas capazes de apoiá-las. Um dos fatores críticos, conforme exposto por (NUSEIBEH; EASTERBROOK, 2000), é o Gerenciamento de Requisitos ou Gerência dos Requisitos. Esse(a) é visto(a) como não apenas uma habilidade de escrever os requisitos, mas também de torná-los legíveis, rastreáveis, permitindo acompanhá-los ao longo do tempo.

Com base no exposto, nesse Trabalho de Conclusão de Curso, buscou-se contribuir com um suporte computacional focado na gerência do ciclo de vida dos requisitos, atuando, mais especificamente, nas atividades de Rastreabilidade e Versionamento.

1.3 Questão de Pesquisa

O principal intuito desse trabalho foi contribuir com a Gerência dos Requisitos, concentrando esforços nas atividades de Rastreabilidade e Versionamento. Portanto, buscou-se, ao final desse trabalho, responder a seguinte questão de pesquisa: Como proporcionar uma solução computacional capaz de auxiliar a Gerência dos Requisitos, considerando as atividades de Rastreabilidade e Versionamento conjuntamente?

Cabe ressaltar que, para a primeira versão dessa solução computacional, utilizou-se Histórias de Usuário como o tipo base de especificação de Requisitos, dada a relevância e a popularidade das metodologias ágeis. Conforme (RONALD, 2001), as histórias são comumente representadas em cartões, sem a pretenção de conter um conjunto completo de restrições e regras, sendo basicamente uma descrição textual suficiente para relembrar aos interessados os requisitos do sistema. Diante disso, a investigação desse artefato e sua viabilidade como artefato gerenciado, compõe um primeiro esforço da ferramenta *online* desenvolvida.

1.4 Objetivos

Como o escopo da ferramenta é abrangente, seguem os objetivos Geral e Específicos desse Trabalho de Conclusão de Curso. Esses conferem um olhar mais focado nas preocupações chave desse trabalho.

1.4.1 Objetivos Gerais

- Desenvolvimento de uma solução computacional visando apoiar a Gerência dos Requisitos, particularmente, procurando contribuir com as atividades de Rastreabilidade e Versionamento.

1.4.2 Objetivos Específicos

- Investigação dos tópicos de interesse desse trabalho, orientando-se pelas palavras chave: Gerência dos Requisitos, Rastreabilidade (Pré e Pós-rastreabilidade) e Versionamento. Os resgistros dessa investigação foram mantidos, permitindo justificar as escolhas para elaboração da solução computacional;

- Implementação de uma solução computacional condizente com as demandas acordadas no processo investigativo. Essa solução confirma-se como um suporte, disponibilizado de forma *online*, que permite rastrear e versionar Requisitos de Software. Além disso, esse suporte auxilia o Engenheiro de Software no que se refere à Gerência dos Requisitos. Vale ressaltar que essa elaboração envolveu não apenas especificar,

documentar, mais também implementar essa solução, resultando em um produto de software, chamado Factbox, e

- Análise dos resultados obtidos utilizando a metodologia científica de pesquisa-ação.

1.5 Organização do Trabalho

Este documento está dividido da seguinte forma:

Capítulo 2 - Referêncial Teórico: Apresenta os conceitos e modelos relacionados ao tema foco desse trabalho.

Capítulo 3 - Suporte Tecnológico: Descreve os recursos tecnológicos utilizados durante a elaboração deste trabalho.

Capítulo 4 - Metodologia: Detalha a metodologia utilizada para o desenvolvimento deste trabalho.

Capítulo 5 - Factbox: Descreve a ferramenta de Gerência dos Requisitos.

Capítulo 6 - Resultados: Descreve os resultados obtidos durante os ciclos de pesquisa-ação executados.

Capítulo 7 - Conclusão: Apresenta as conclusões do trabalho.

2 Referencial Teórico

Para entendimento do contexto, no qual está inserido esse Trabalho de Conclusão de Curso, faz-se necessária a apresentação do embasamento teórico sobre alguns pontos chave. A Figura 1 contextualiza sobre os tópicos de interesse do trabalho, sendo esses detalhados nas seções adiante. A seção 2.1 exibe os principais tópicos detalhados no decorrer do capítulo. Assim, na seção 2.1, apresenta-se uma contextualização sobre a Engenharia de Software e a sua relação com a Engenharia de Requisitos. Na seção 2.2, faz-se referência à Gerência dos Requisitos e seus aspectos tangentes ao presente trabalho. Na seção 2.3, descreve-se sobre os modelos conceituais relacionados ao trabalho. Na seção 2.4, com base em (FIRESMITH, 2003), faz-se referência às propriedades relevantes para uma ferramenta de requisitos. Por fim, tem-se o resumo do capítulo.

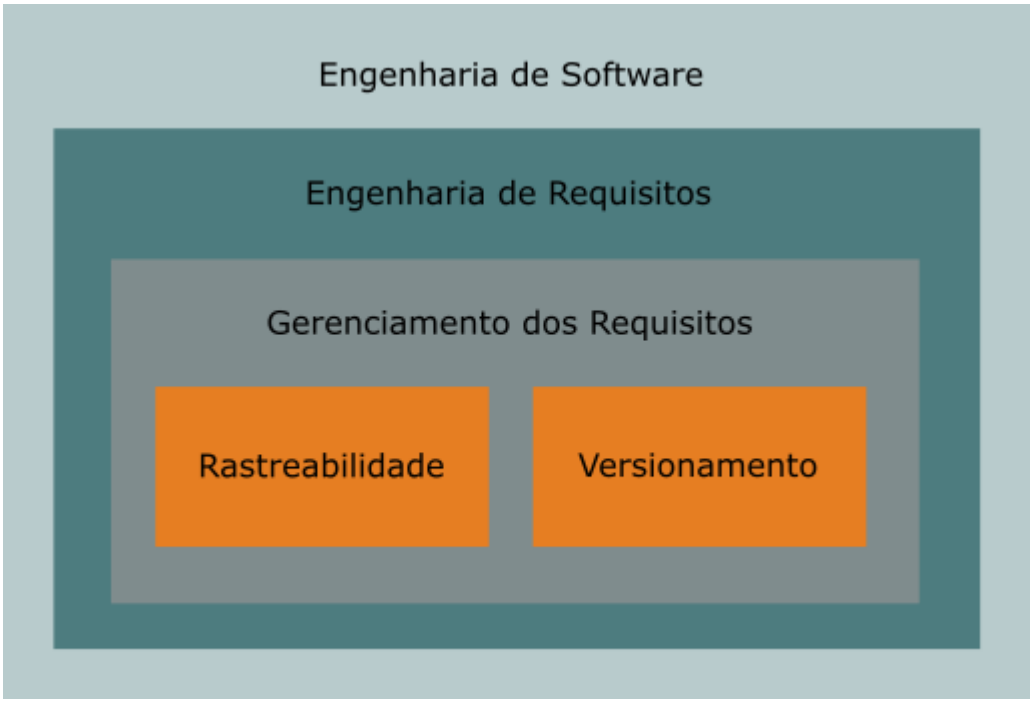

Figura 1 – Representação dos principais focos de atuação e de interesse deste trabalho

2.1 Contextualização da Engenharia de Requisitos

Essa seção irá contextualizar a Engenharia de Requisitos a partir da Engenharia de Software.

2.1.1 Engenharia de Software

É perceptível que software é um produto essencial no cenário mundial, dada a presença deste em diferentes domínios. Como explicitado por (PRESSMAN, 2010), no passado, era complicado prever que o software estaria incorporado em sistemas de todas as áreas. Como software vem se tornando cada dia mais importante, a comunidade de software reforça suas contribuições para investigar tecnologias que tornem mais fácil, rápido e barato desenvolver software de alta qualidade. Pode-se dizer que essa é uma responsabilidade da Engenharia de Software.

Conforme justifica (SOMMERVILLE, 2007), a noção de Engenharia de Software foi proposta em 1968, em uma conferência para discutir o que foi chamado de "crise do software". Segundo (FITZGERALD, 2012), essa crise surgiu com os longos prazos e custos, acima do estimado, além do eventual mau funcionamento quando entregue. A crise do software, ocasionalmente, levou a uma série de reflexões e mudanças na forma como o software era visto e no seu processo de desenvolvimento.

> Foram desenvolvidos métodos eficazes de especificação, *design* e implementação de software. Novas notações e ferramentas reduziram o esforço necessário para produzir sistemas grandes e complexos (SOMMERVILLE, 2007).

De fato, houve uma evolução no desenvolvimento de software. Porém, ainda existem dificuldades e problemas. (PRESSMAN, 2010) discursa sobre alguns fatos relacionados à forma como se deve enfrentar o desafio de fazer software atualmente. Dentre esses, tem-se a necessidade de concentrar esforços para compreender o problema antes de desenvolver uma solução de software. Essa demanda justifica a necessidade de uma das subáreas da Engenharia de Software, tão complexa quanto: a Engenharia de Requisitos.

2.1.2 Engenharia de Requisitos

Conforme (BOURQUE; FAIRLEY, 2014), o termo Engenharia de Requisitos é amplamente utilizado na Engenharia de Software para denotar o tratamento sistemático dos Requisitos de Software. A Engenharia de Requisitos pode ainda ser entendida como o espectro de técnicas e tarefas para compreender os requisitos de um software (PRESSMAN, 2010). Devido à complexidade das atividades da Engenharia de Requisitos no que corresponde à compreensão do domínio, de acordo com (PARNAS; CLEMENTS, 1986), jamais será encontrado um processo de software que permita uma implementação de forma completamente racional e lógica, ainda que se possa chegar muito próximo disso. Dessa forma, é exposto que:

1. Na maioria dos casos, os representantes do domínio de conhecimento no projeto não sabem exatamente o que querem e não conseguem explicar o que sabem;

2. Até mesmo que se saiba os requisitos do software, alguns fatores de relevância para o projeto costumam ser negligenciados, sendo descobertos apenas durante a implementação;

3. Mesmo que se saiba todos os fatores relevantes antes de começar, a experiência mostra que os humanos são incapazes de compreender completamente os detalhes que devem ser entendidos para se construir um software correto;

4. Mesmo que entendido todos os detalhes, os projetos estão predispostos a mudar, invalidando as decisões previamente tomadas para o projeto;

5. O erro humano só pode ser evitado, se for evitado o uso de humanos;

6. É natural que os desenvolvedores estejam com ideias pré-concebidas do projeto, sejam elas inventadas ou adquiridas em relação a outros projetos, e

7. É comum, por questões econômicas, que o software desenvolvido não corresponda aos requisitos, e mesmo assim seja aceito como forma de poupar esforço.

Esses itens referem-se ao desenvolvimento de software como um todo. (PARNAS; CLEMENTS, 1986) faz uso deles para justificar a necessidade de um processo ideal de software, sugerindo, em seguida, a necessidade dos documentos de requisitos, tal que:

1. São o meio para se guardar as expectativas do usuário, podendo ser usados para futuras revisões;

2. Se utilizados como guia de referência, evitam decisões acidentais no planejamento do software;

3. Evitam inconsistência e duplicação, poupando dúvidas para perguntas já acordadas;

4. Podem ser usados para estimativas de quantidade de trabalho ou outros recursos para criação do sistema;

5. Uma garantia contra a rotatividade de pessoas, pois auxilia para que o conhecimento não seja perdido;

6. Fornece uma boa base para o desenvolvimento do plano de testes;

7. Uma forma de definir as restrições do sistema para futuras alterações, e

8. Se estiverem completos e precisos, podem ser usados por um não *expert* de requisitos como forma de argumentação.

De acordo com (TEAM, 2010), é esperado que para um processo de software maduro, as especificações de requisitos cumpram com os itens citados anteriormente, pois esses impactam diretamente na qualidade do software. Para manter esses aspectos no software durante seu ciclo de vida, o (TEAM, 2010) ainda sugere a atividade de Gerência dos Requisitos como uma forma de obter e controlar as mudanças dos requisitos (técnicos e não técnicos).

2.2 Gerência dos Requisitos

Como explica (SOMMERVILLE, 2007), a Gerência dos Requisitos é o processo de entender e controlar as mudanças dos requisitos do sistema, tal que seja mantido o rastro dos requisitos individuais e as ligações entre os requisitos dependentes. Conforme defende (TEAM, 2010), os requisitos podem mudar por diferentes motivos. Portanto, à medida que as necessidades mudam e o trabalho prossegue, os requisitos existentes precisam ser modificados de forma eficiente e efetiva, de modo que, para analisar efetivamente o impacto das mudanças, é necessário que seja conhecida a fonte de cada requisito e documentada a razão das mudanças.

Tal como explica (NUSEIBEH; EASTERBROOK, 2000), minimamente, é esperado o uso de técnicas e ferramentas para o Gerenciamento dos Requisitos, de modo que seja possível fazer o controle de versão e a exploração de referências de Rastreabilidade, monitorando e controlando o impacto das mudanças nas especificações de requisitos.

2.2.1 Rastreabilidade

De acordo com (RAMESH; JARKE, 2001), Rastreabilidade dos Requisitos refere-se à habilidade de detalhar e acompanhar a vida dos requisitos. Portanto, pode ser vista e tratada de dois modos; tanto na direção adiante, conhecendo desde as origens até a implementação; quanto no sentido inverso.(RAMESH; JARKE, 2001) ainda classifica a Rastreabilidade em dois tipos: (i) Pré-rastreabilidade, relacionado ao processo de produção dos requisitos, correspondendo aos aspectos da vida dos requisitos antes que sejam desenvolvidos. (ii) Pós-rastreabilidade, refere-se aos requisitos após implantação como parte da *baseline* de requisitos, correspondendo aos impactos na arquitetura e na solução ao longo do tempo de vida dos requisitos.

Segundo (SERRANO; LEITE, 2011), a Rastreabilidade dos Requisitos apoia os processos de gerenciamento, evolução e validação de software, pois os requisitos expressam as necessidades e restrições do software, ao passo que a Rastreabilidade permite o acompanhamento das alterações no ciclo de vida dos requisitos, sendo assim fundamentais para analisar os impactos das mudanças. Porém, a adoção de Rastreabilidade no desenvolvimento de software não é uma atividade trivial. Como explica (SAYÃO; CESAR; LEITE,

2005), mesmo as organizações que seguem os preceitos dos modelos de maturidade, ainda têm dificuldades em manter os elos de Rastreabilidade.

Para (GOTEL; FINKELSTEIN, 1994), a maioria dos problemas associados à Rastreabilidade fraca tem relação com a falta de uma pré-rastreabilidade, ou com a prática inadequada da mesma. Esse problema propaga-se nas etapas seguintes do desenvolvimento, tal que (RAMESH; JARKE, 2001) classifica a associação das justificativas e fontes aos requisitos, como um grande desafio da Engenharia de Requisitos.

2.2.2 Versionamento

Versionamento, ou Controle de Versão, é a atividade conhecida na comunidade de software como controle ou transparência das mudanças através de uma linha histórica. Tal como descreve (NUSEIBEH; EASTERBROOK, 2000), para que as alterações de requisitos sejam gerenciadas, é esperado, minimamente, o Versionamento através de técnicas e/ou ferramentas.

Para (FIRESMITH, 2003), controle de versão é um tópico importante e esperado de uma ferramenta de requisitos. Mais adiante, será dedicado um tópico específico para discussão sobre os pontos esperados de uma ferramenta de Engenharia de Requisitos. No momento, cabe a reflexão de que o controle de versão de código é um tópico indispensável e de muita atenção dentro de projetos de software. Entretanto, pouco se discute sobre o Versionamento de requisitos e demais artefatos do ciclo de vida dos requisitos.

2.3 Especificações de Requisitos

Existem variadas iniciativas, entre técnicas, métodos e artefatos envolvidos na Engenharia de Requisitos. Entretanto, tal como defende (BROOKS, 1986), nenhuma pode-se provar melhor que a outra em todos os aspectos. Esse autor coloca que, olhando para o passado, não é possível ver "balas de prata"(soluções completamente satisfatórias) para desenvolvimento de Software. Nos subtópicos seguintes, procura-se detalhar as principais especificações utilizadas nesse trabalho.

2.3.1 Rich Picture

De acordo com (BERG, 2015), as informações dos sistemas muitas vezes não conseguem atingir a qualidade esperada. Dentre os motivos, destaca-se a incapacidade, ou a dificuldade, de identificar os requisitos do usuário. Para lidar com essa dificuldade, pode ser utilizado o *Rich Picture* como uma forma visual para entender os diferentes pontos de vista, afim de criar um conhecimento compartilhado da organização.

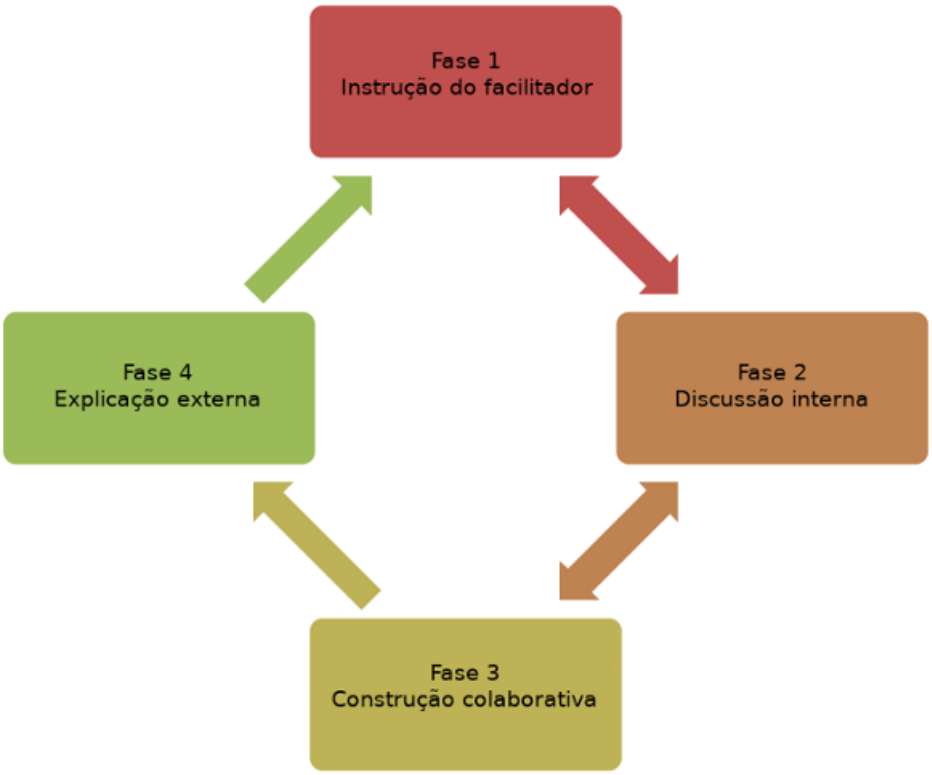

Figura 2 – Processo de criação do Rich Picture, traduzido a partir de (BERG, 2015)

Para (BERG, 2015), conforme expressado de forma generalizada na Figura 2, o processo de criação do *Rich Picture* tem quatro fases:

- **Fase 1.** Instruções sobre o uso do *Rich Picture*.
- **Fase 2.** A discussão interna dos grupos interessados. O autor enfatiza que não necessariamente essa fase precede a fase 3.
- **Fase 3.** A criação do *Rich Picture*, em que os grupos se comunicam e compartilham as percepções do domínio em questão, trazendo o entendimento individual para a atividade de diagramação colaborativa.
- **Fase 4.** Busca comunicar, por expressão verbal, com o facilitador e/ou outros grupos externos. O facilitador pode solicitar mais *Rich Pictures* para expandir questões que surgiram da primeira diagramação, reiniciando o processo.

Conforme descreve (BERG; POOLEY, 2013), o *Rich Picture* serve como uma forma de comunicação global que excede os limites da comunicação textual e verbal. Entretanto, *Rich Picture* são criados com pouco suporte tecnológico, sendo comumente utilizados desenhos a mão livre. Um problema maior é manter a Rastreabilidade desse artefato com outros documentos, uma vez que o mesmo tem que ser digitalizado como imagem. A Figura 3 ilustra um exemplo desse tipo de artefato.

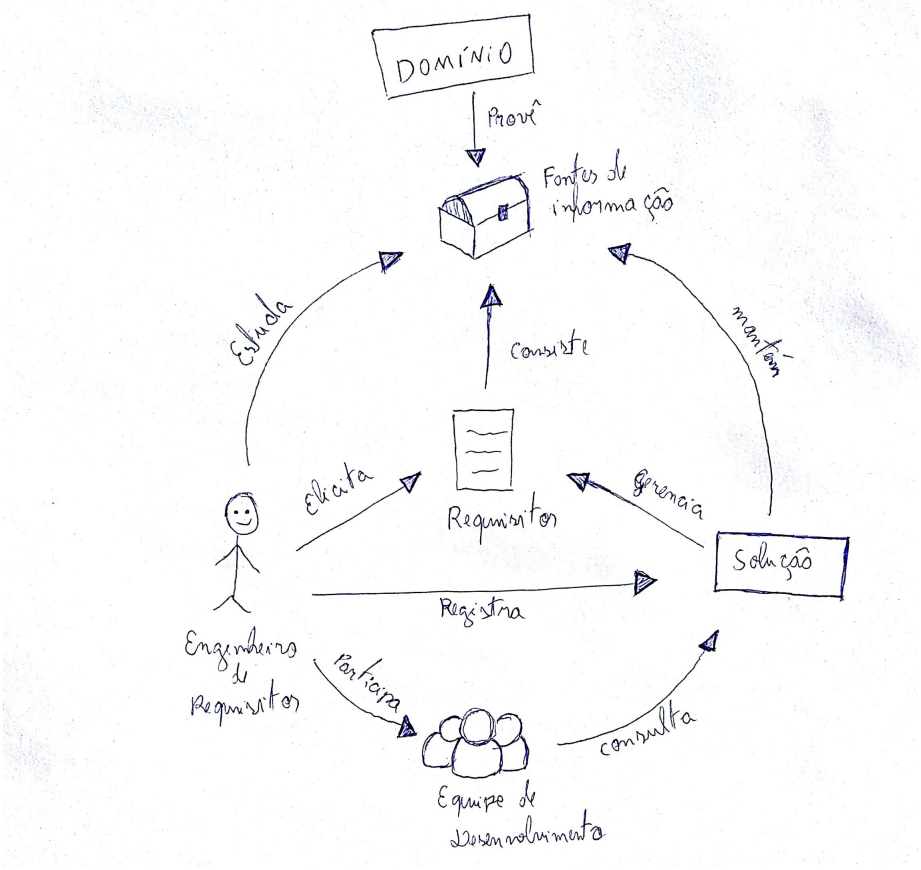

Figura 3 – Rich Picture da solução proposta no trabalho

2.3.2 História de Usuário

Segundo (BECK; FOWLER,), a história de usuário descreve uma funcionalidade que agrega valor aos interessados em um sistema de software. O autor prossegue definindo três aspectos importantes em uma história de usuário:

- Uma descrição por escrito usada para planejar a solução, ou como um lembrete do escopo aos interessados no projeto;
- Conversas relacionadas, que servem para detalhar melhor a história, e
- Descrições documentadas e detalhadas que possibilitam testes e que podem ser utilizadas para determinar quando uma história é concluída.

Conforme explica (WILLIAMS, 2003), as histórias de usuário originaram-se com o método ágil conhecido por *Extreme Programming* (XP). O autor defende que diferentemente de outros modelos de desenvolvimento de software, as equipes XP dispensam os longos ciclos de criação de requisitos e planejamento de projeto, demandando alguns minutos por dia para execução de cada uma dessas atividades. Como resultado, as histórias

de usuário são priorizadas pelos clientes, e os desenvolvedores estimam quanto tempo será necessário para implementá-las.

2.4 Ferramentas de Requisitos

Para (FIRESMITH, 2003), ao longo do tempo, as ferramentas de requisitos aperfeiçoaram alguns critérios como:

1. Ferramentas de processamento de texto e planilhas, como Word e Excel da Microsoft;
2. Bancos de dados com recursos para geração de relatórios sem formatos especificados;
3. Ferramentas de gerenciamento de requisitos que apóiam outras tarefas da Engenharia de Requisitos e não exclusivamente a Gerência dos Requisitos, e
4. Ferramentas devidamente integradas no ambiente de desenvolvimento (IDE).

O autor ressalta que a maioria das ferramentas atuais não apóia todas essas atividades e tarefas, em especial, conjuntamente. Afim de formular uma orientação para escolha de uma ferramenta de Engenharia de Requisitos adequada, (FIRESMITH, 2003) começa por descrever a importância de um repositório de requisitos como forma de oferecer um suporte ao armazenamento de requisitos em um nível mais fino de granularidade, tornando-os mais facilmente tratáveis. Após esse primeiro critério, são defendidos os seguintes aspectos como críticos para uma ferramenta de requisitos:

1. Interface de usuário. Facilitando a interação do usuário para criar e manter os requisitos individuais, seus metadados, modelos e quaisquer outras informações textuais ou não. A interface de usuário deve ainda entender as diferenças entre os tipos de requisitos, seus metadados em comum e os específicos;
2. Apoio à Engenharia de Requisitos. Deve ser provido o apoio a todas as atividades base da Engenharia de Requisitos;
3. Apoio a atividades relacionadas. Deve conferir o apoio às atividades ligadas diretamente à Engenharia de Requisitos, como controle de escopo, configuração de gerenciamento e engenharia da qualidade;
4. Desenvolvimento em equipe. Outros envolvidos precisam ter acesso para fins de aprendizagem, avaliação ou aprovação, além do trabalho cooperativo entre desenvolvedores ser um fator necessário;
5. Segurança. As informações nos requisitos envolvem muitas vezes dados sigilosos que precisam ser guardados;

6. Outros fatores de qualidade. Como qualquer outra ferramenta, é esperado que se cumpra com alguns fatores de qualidade, como completude, internacionalização, desempenho, escalabilidade e usabilidade;

7. Desenvolvimento distribuído. É cada vez mais comum o desenvolvimento de times e organizações separados geograficamente. Assim, é esperado suporte à Engenharia de Requisitos de forma distribuída;

8. Reutilização de requisito. Essa pode poupar retrabalho da equipe. Portanto, deve ser possível a incorporação de requisitos existentes, ainda de forma granular, para aceitar modificações, rejeição ou aceitação conforme revisões, e

9. Não apenas uma ferramenta CASE. Ferramentas CASE (*Computer-Aided Software Engineering*) são ferramentas que auxiliam atividades específicas da Engenharia de Requisitos. Uma ferramenta assim é difícil de se integrar com o ambiente de desenvolvimento.

2.5 Resumo do Capítulo

O capítulo abordou as bases teóricas que sustentam este Trabalho de Conclusão de Curso. Partindo de uma contextualização da Engenharia de Requisitos, comentou-se sobre Gerência dos Requisitos para exploração de dois tópicos: Rastreabilidade e Versionamento. Parte do capítulo foi dedicada à explicação dos artefatos utilizados na primeira versão da ferramenta *online*. O capítulo termina com uma explicação sobre as expectativas esperadas para uma ferramenta de Engenharia de Requisitos, as quais orientaram o desenvolvimento da ferramenta Factbox.

3 Suporte Tecnológico

Este capítulo apresenta as principais tecnologias utilizadas para desenvolvimento da ferramenta de Gerência dos Requisitos. Por questões de organização, o capítulo está dividido em seções, sendo as principais: seção 3.1, na qual serão tratados os suportes que auxiliaram no desenvolvimento da ferramenta *online* Factbox, e seção 3.2, na qual serão apresentados os suportes utilizados no escopo da pesquisa desse trabalho. Por fim, tem-se o resumo do capítulo.

3.1 Desenvolvimento de Software

Para o desenvolvimento da ferramenta Factbox, serão utilizadas as tecnologias apresentadas nas seções 3.1.1 a 3.1.5.

3.1.1 Kubuntu 17

Kubuntu é um sistema operacional, livre e de código aberto, baseado no Ubuntu e licenciado sob os termos *GNU GPL2* (KUBUNTU, 2017). Esse foi o sistema operacional utilizado como ambiente de desenvolvimento do software.

3.1.2 Ruby on Rails 5

Ruby é uma linguagem interpretada, multiparadigma, livre e de código aberto, licenciada sob os termos *BSDL* (RUBY, 2017). *Rails* é um *framework Web MVC (Model View Controller)* da linguagem Ruby, que segue fortemente o modelo de programação orientado à convenção. Também é livre e de código aberto. Esse atende aos termos da licença *MIT*.

O núcleo da ferramenta foi desenvolvido utilizando a tecnologia *Ruby on Rails*.

3.1.3 Vim 8

Vim é um editor de texto simples, porém eficiente, que conta com uma variedade de *plugins* e possibilidades de configuração. Será o principal editor utilizado em desenvolvimento (VIM, 2018).

3.1.4 Git 2.14

Git é um sistema de versionamento robusto e simples, cuja principal forma de interação é via linha de comandos. Foi criado inicialmente por Linus Torvalds e a comuni-

dade do *Kernel* do Linux (GIT, 2018). Popularizou-se mais tarde entre a comunidade de software. Esse suporte foi utilizado para controle de versão do código fonte do Software bem como da monografia desse trabalho.

3.1.5 Github

Github é um software *online* que tem como principal funcionalidade o repositório remoto de projetos versionados no Git. Além disso, o Github conta com gerenciamento de tarefas, suporte a *Wikis* por projeto, revisão de código e gerenciamento de *releases*.

Todas as funcionalidades expostas anteriormente foram utilizadas para apoiar o desenvolvimento do software. Na Wiki, por exemplo, foram mantidos artefatos de arquitetura, tutoriais e outros possíveis documentos técnicos. Os requisitos não foram mantidos nessa plataforma. Os mesmos foram mantidos na própria ferramenta Factbox.

3.2 Pesquisa

Para realização do projeto de pesquisa, as ferramentas descritas na seções 3.2.1 a 3.2.3 foram utilizadas.

3.2.1 LaTeX 3.14

(LATEX, 2018) é um sistema de composição textual, que tem por objetivo facilitar a formatação textual, permitindo um maior enfoque na produção de conteúdo. É utilizado para escrita deste trabalho.

3.2.2 Mendeley Desktop 1.17

Mendeley Desktop é uma ferramenta de gerenciamento de material de pesquisa (ELSEVIER, 2018). É um software proprietário que possui versão comercial e gratuita. Para esse trabalho de pesquisa, foi utilizada a versão gratuita da ferramenta.

3.2.3 TexMaker 5

TexMaker é um editor gráfico para escrita em LaTex (XM1MATH.NET, 2018). Tem suporte multiplataforma para *Mac OS*, *Windows* e *Linux*. Foi utilizado para escrita deste trabalho, em conjunto com o LaTeX.

3.3 Resumo do Capítulo

Este capítulo concentrou-se em explicar o suporte tecnológico utilizado na elaboração deste Trabalho de Conclusão de Curso. Foram abordadas tanto as tecnologias de apoio à implementação da ferramenta Factbox, quanto as ferramentas de apoio à pesquisa. Na Tabela 1, tem-se um resumo das ferramentas apresentadas.

Tabela 1 – Resumo das ferramentas utilizadas no trabalho.

Ferramenta	**Descrição de Uso**	**Versão**
Kubuntu	Sistema operacional utilizado para desenvolvimento	17
Ruby on Rails	Framework web utilizado para desenvolvimento	5
Vim	Editor de texto utilizado para desenvolvimento	8
Git	Ferramenta de versionamento, utilizada no trabalho escrito e na aplicação	2.14
Github	Repositório *online* para versionamento do trabalho	
LaTeX	Sistema de composição textual para formatação da monografia	3.14
Mendeley Desktop	Ferramenta de gerenciamento do material de pesquisa	1.17
TexMaker	Editor textual utilizado para escrita da monografia	5

4 Metodologia

Este capítulo descreve detalhes sobre as metodologias utilizadas para a elaboração desse trabalho, seja no âmbito da pesquisa (parte investigativa do trabalho), seja para desenvolvimento da ferramenta *online* Factbox, seja para coleta dos resultados obtidos. Na seção 4.1, concentrou-se em classificar a pesquisa com base em diferentes perspectivas. Essa classificação ajudou na especificação das metodologias associadas a esse trabalho. Na seção 4.2, tem-se a descrição das atividades relacionadas à condução do TCC como um todo, da definição do tema à entrega da monografia final. Nas seções 4.3 e 4.4, tem-se, respectivamente, os detalhamentos da metodologia que guiou as atividades de desenvolvimento bem como da metodologia que orientou a análise dos resultados obtidos. Na seção 4.5, procurou-se apresentar um cronograma, apontando as principais tarefas realizadas ao longo desse trabalho. Por fim, o resumo do capítulo.

4.1 Classificação da Pesquisa

A metodologia adotada foi classificada quanto à abordagem, natureza, objetivos e procedimentos técnicos. A Figura 4 expõe, através dos itens de cor, as opções definidas para este trabalho.

Quanto à abordagem do problema, uma pesquisa pode ser categorizada em quantitativa e qualitativa. A primeira refere-se à determinação de resultados quantificáveis, através de técnicas estatísticas, por exemplo. Por outro lado, conforme defende (GIL, 2010), a análise qualitativa envolve uma sequência de atividades, como redução de dados, categorização e interpretação desses dados.

Nesse TCC, a pesquisa pode ser categorizada, em termos de abordagem do problema, como quantitativa e qualitativa. Quantitativa, pois realizou-se pesquisa-ação (TRIPP, 2005) para refinamento da pesquisa, coletando métricas para refinamento da aplicação. A cada ciclo de pesquisa-ação, foram levantados cenários de uso, no qual foram coletados os dados. Esses dados serão exibidos na Capítulo 6. Adicionalmente, a análise dessa coleta de dados envolveu compreender o público alvo da ferramenta bem como perceber se a ferramenta atende de fato ou não ao público alvo em aspectos tipicamente subjetivos. Portanto, pode ser também classificada como qualitativa, uma vez que, de acordo com (GERHARDT, 2009), a pesquisa qualitativa não é focada na representatividade numérica, mas na compreensão de determinado grupo.

Em relação à natureza desta pesquisa, caracteriza-se como aplicada, pois conforme explica (SILVEIRA; CÓRDOVA, 2009), essa teve por objetivo gerar conhecimentos com

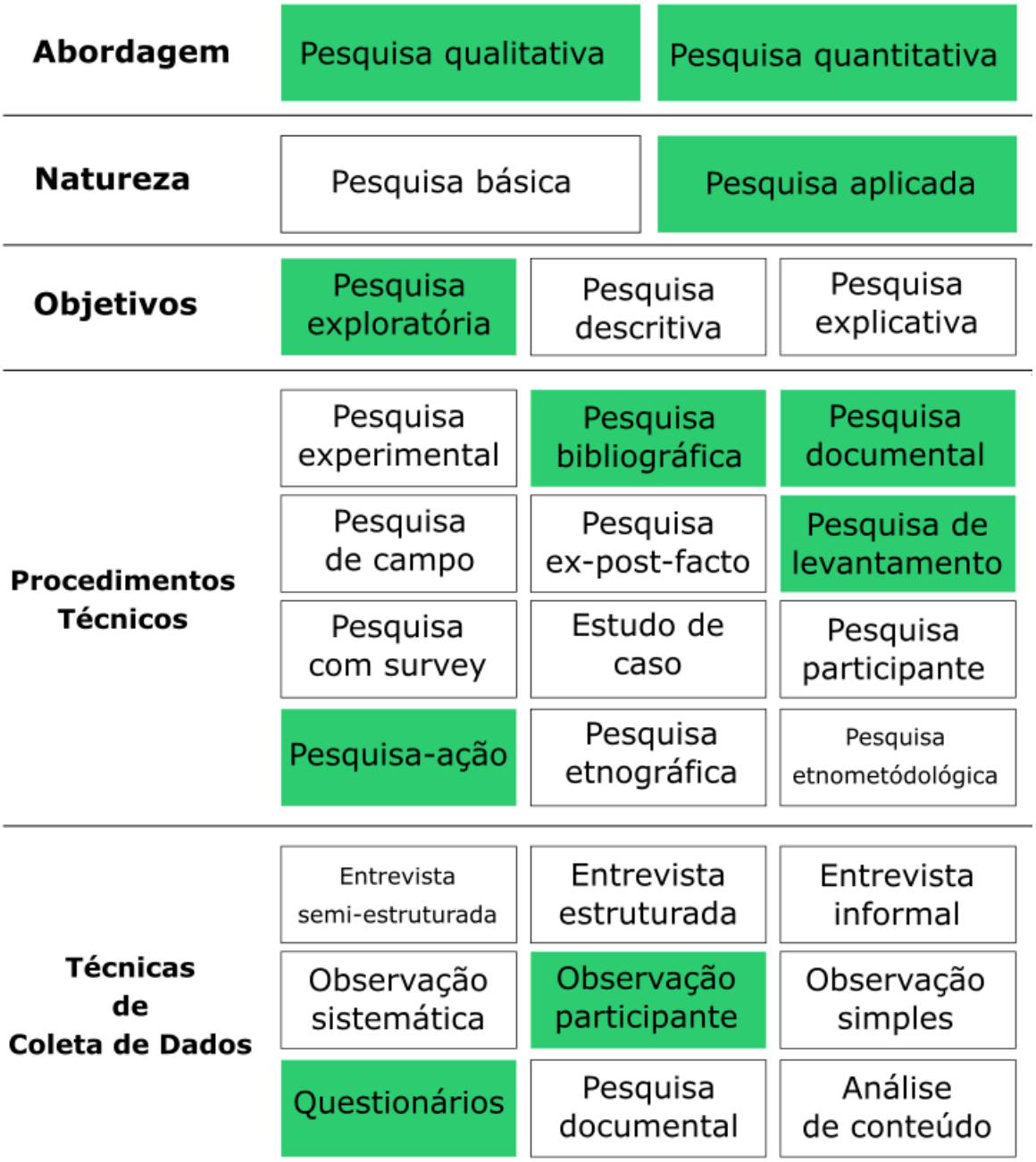

Figura 4 – Seleção metodológica

intuito de encontrar soluções para problemas específicos, no caso, na área de Engenharia de Requisitos, mais especificamente em Gerência dos Requisitos.

Com base nos objetivos, conforme (GIL, 2010), pode-se classificar as pesquisas em três tipos: Exploratória, Descritiva e Explicativa. Esse trabalho enquadra-se na categoria exploratória, pois se tenta entender a necessidade dos Engenheiros de Requisitos, afim de fornecer uma solução de Software. Adicionalmente, a ferramenta é aplicada à uma demanda específica da área de Engenharia de Requisitos, sendo a elaboração desse suporte tecnológico sustentada por pesquisa-ação. Nessa, tem-se investigação junto ao público

alvo, explorando cenários inerentes à Gerência de Requisitos. Então, pesquisa Exploratória é algo bem condizente com os objetivos desse trabalho.

Dentre os possíveis procedimentos técnicos, acordados em (GIL, 2010), procurou-se entender os seguintes:

- Pesquisa bibliográfica: Desenvolvimento de conteúdo a partir de livros e artigos científicos;
- Pesquisa documental: É similar à pesquisa bibliográfica, distinguindo-se apenas da natureza da fonte de informação. As fontes deste trabalho, relacionadas a esse tipo de pesquisa, consistem basicamente em artigos eletrônicos e documentação técnica;
- Pesquisa-ação: Conforme (TRIPP, 2005), essa técnica é uma das várias técnicas de investigação-ação, a qual se planeja, implementa, descreve e avalia com o intuito de continuamente melhorar essa prática investigativa, ao passo que se aprende mais sobre o domínio, e
- Pesquisa de levantamento: É o tipo de pesquisa utilizado quando se deseja conhecer o comportamento das pessoas através de interrogações diretas a esse público alvo. Segundo (GERHARDT, 2009), esse levantamento pode ser de dois tipos: levantamento de uma amostra, ou uma população(também chamado censo).

Para coleta de dados, foram selecionadas as técnicas de observação participante e questionário. A primeira, segundo (VALLADARES, 2007), consiste em um processo longo e complexo, que requer a paciência, postura e disciplina do pesquisador, sendo uma técnica de apoio à análise qualitativa. Já o questionário, ajuda a extrair dados para uma compreensão mais analítica do contexto da pesquisa.

4.2 Fluxo de Atividades

A partir da classificação da pesquisa, considerando as atividades necessárias para realização do trabalho, tem-se o fluxo de atividades apresentado na Figura 5. Esse processo acompanhou a condução do trabalho como um todo. Na presente seção, será apresentada uma breve descrição de cada atividade.

Definir tema: Como primeira atividade, essa teve como objetivo a escolha de um tema. Junto aos orientadores e com base em pesquisas realizadas, decidiu-se trabalhar com Gerenciamento dos Requisitos.

Realizar levantamento bibliográfico: Esta atividade, já realizada, teve por objetivo pesquisar conteúdo de pesquisa relativo ao tema do trabalho.

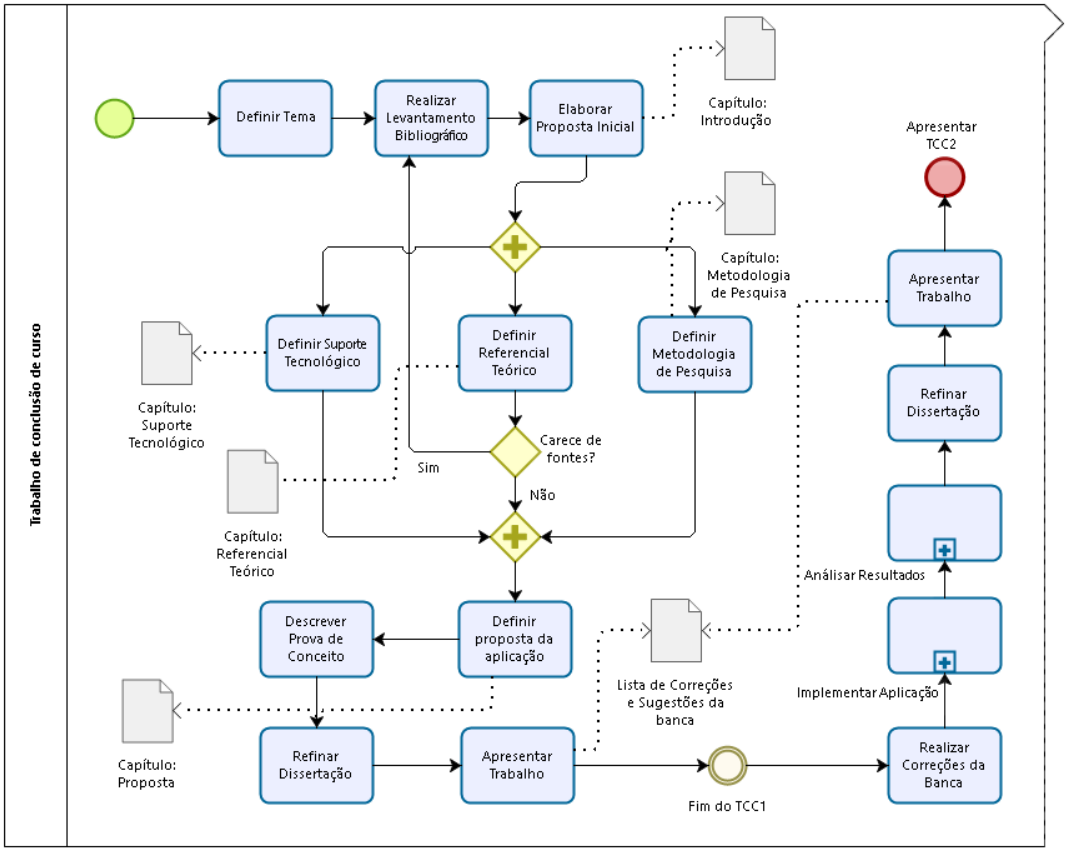

Figura 5 – Fluxo de atividades do trabalho

Elaborar proposta inicial: Esta atividade, já finalizada, teve por objetivo definir uma proposta que contemplasse uma solução para os problemas existentes em relação ao Gerenciamento de Requisitos, em específico, nos tópicos de Rastreabilidade e Versionamento.

Definir referencial teórico: Esta atividade, já realizada, teve por objetivo, acordar, através do levantamento bibliográfico realizado, os principais conceitos que sustentam esse trabalho. Como resultado, esta atividade teve por saída o Capítulo 2.

Definir suporte tecnológico: Esta atividade, já realizada, teve por objetivo descrever o suporte tecnológico utilizado no desenvolvimento do trabalho, tanto em sua parte de pesquisa, como no desenvolvimento da ferramenta Factbox. O Capítulo 3 foi resultado dessa atividade.

Definir metodologia de pesquisa: Esta atividade, já realizada, teve por objetivo definir a metodologia utilizada para guiar esse trabalho de pesquisa, permitindo classificar essa quanto à natureza, abordagem, objetivos e procedimentos. Ao final, como resultado, foi possível definir as atividades necessárias para execução do trabalho.

Definir proposta da aplicação: Essa atividade, já realizada, consistiu em definir

o escopo da solução proposta, detalhando aspectos operacionais, bem como a descrição da arquitetura, em linhas gerais.

Descrever prova de conceito: Esta atividade, já realizada, contempla a descrição de um projeto que serviu como prova de conceito para esse trabalho. Descrito com mais detalhes na subseção 5.1.2.

Refinar dissertação: Esta atividade, já realizada, teve por objetivo conduzir, junto aos orientadores, ciclos de revisões para melhoria contínua da monografia, considerando uma primeira apresentação aos membros da banca.

Apresentar trabalho: Esta atividade visou apresentar os resultados obtidos na primeira etapa do trabalho para os membros da banca. Como resultado, orientações e recomendações foram obtidas, permitindo refinar o trabalho.

Realizar correções da banca: Esta atividade teve por objetivo o refinamento do trabalho com base nas impressões adquiridas junto aos membros da banca.

Implementar aplicação: Esta atividade teve por objetivo o desenvolvimento da ferramenta Factbox. Essa atividade, por envolver desenvolvimento, orientou-se pela metodologia definida na seção 4.3. Como resultado, tem-se a própria ferramenta Factbox, apresentada no Capítulo 5.

Analisar resultados: Esta atividade possibilitou a análise dos resultados obtidos, incluindo: avaliar se a ferramenta Factbox atende de fato às demandas expostas no Capítulo 1 dessa monografia, ou seja, aos objetos geral e específicos estabelecidos a priori. Como essa atividade é mais abrangente, e envolveu a análise dos resultados obtidos, a mesma orientou-se pela metodologia acordada na seção 4.4.

Refinar a dissertação: Novamente, foram realizados ciclos de revisões, junto aos orientadores, visando refinar ainda mais o trabalho em termos de desenvolvimento e documentação.

Apresentar trabalho: Esta atividade visa a apresentação do resultado final do trabalho aos membros da banca. Novamente, espera-se coletar as impressões desses membros, de modo que, se necessário, será realizada mais uma atividade para refinamento da monografia e do projeto.

4.3 Metodologia de Desenvolvimento

Optou-se por conduzir as atividades de desenvolvimento através de um processo baseado em metodológias ágeis. Como práticas adotadas, conforme explica (HIGHSMITH; COCKBURN, 2001), fez-se uso de Histórias de Usuário para criação de um *backlog*. Executando as iterações (também conhecidas por *Sprints*), em períodos constantes de duas semanas. Essas foram executadas paralelamente aos ciclos de pesquisa-ação. A Figura 6,

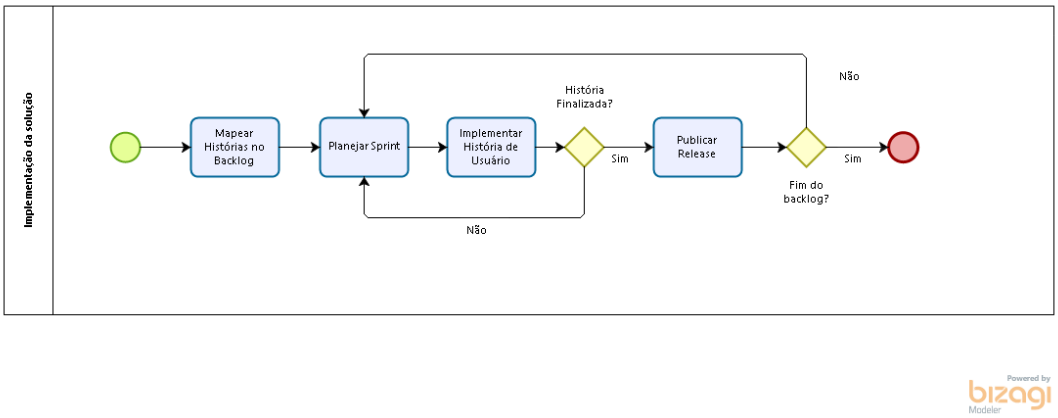

Figura 6 – Processo de desenvolvimento

representa o processo de desenvolvimento levando em conta essas práticas. Segue o detalhamento das demais atividades:

Mapear histórias no *backlog*: Esta atividade teve por objetivo a atualização do *backlog*, de forma a alterar o estado de desenvolvimento das Histórias de Usuário;

Planejar *sprint*: Esta atividade permitiu priorizar as Histórias, obtendo os *backlogs* das *sprints*. Visando ilustrar um desses *backlogs*, tem-se a Figura 7;

Figura 7 – Itens do backlog da *Sprint*

Implementar história de usuário: Trata-se, propriamente, da atividade de implementação das histórias priorizadas no *backlog*, e

Publicar *release*: Atividade referente à publicação das histórias implementadas durante a *Sprint*, e que cumprem com os critérios de aceitação, previamente definidos na escrita da história.

4.4 Metodologia de Análise de Resultados

Pesquisa-ação é uma metodologia científica muito indicada para permitir a evolução de um software de forma iterativa e incremental. Essa metodologia tem como dificuldade, conforme acordado em (GIL, 2010), a ordenação de suas atividades de forma cronológica, uma vez que elas não são, necessariamente, sequencialmente executadas. Ainda segundo o autor, é possível apresentar um conjunto de ações que representam as etapas da pesquisa-ação. Essas atividades estão apresentadas na Figura 8. Entretanto, um ponto alto da pesquisa-ação, e por isso optou-se pelo uso da mesma nesse projeto, é o fato dessa permitir ciclos de refinamento contínuos. Em cada ciclo, tem-se destaque a: um momento dedicado à pesquisa de um ou mais *concerns*; seguido de um momento para coleta de dados, junto, por exemplo, ao público alvo, e, por fim, um momento dedicado às ações, as quais permitem evoluir o tópico investigado orientando-se pelos *feedbacks* coletados.

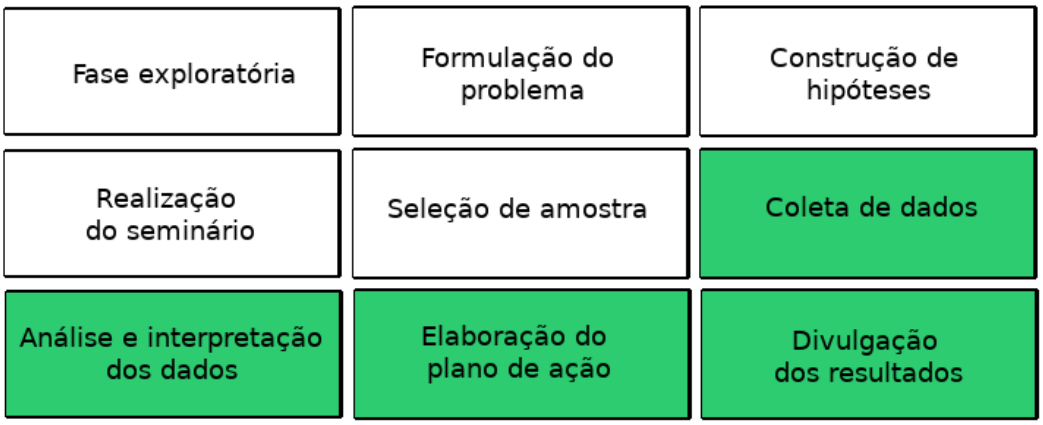

Figura 8 – Etapas da pesquisa-ação.

No presente projeto, a pesquisa-ação permitiu fazer uso desses refinamentos cíclicos, com base na coleta dos dados junto ao público alvo da ferramenta. Detalhes desses refinamentos estão apresentados no Capítulo 6.

Coleta de dados: Avaliou-se a solução, aplicando questionários e fazendo uso de observação participante, conforme discurssado na seção 4.1.

Análise e interpretação de dados: Segundo o autor, essa tarefa pode ser realizada de duas formas: i) de forma clássica, utilizando procedimentos como: categorização,

codificação, tabulação, análise estatística e generalização, e ii) de forma interpretativa, através de discurssões em torno dos dados obtidos. No presente trabalho, fez-se uso de plotagem de gráficos, tanto baseados em métricas, visando uma análise quantitativa; quanto baseados nos dados coletados junto ao público alvo da ferramenta, visando uma análise qualitativa. Novamente, maiores detalhes quanto a esses resultados constam no Capítulo 6.

Elaboração do plano de ação: Essa atividade corresponde ao planejamento de uma ação, cujo propósito, é solucionar um problema que foi objeto de investigação.

Divulgação dos resultados: Dadas que as atividades anteriores foram executadas ciclicamente, ao final de cada ciclo, tem-se a elaboração de um relatório, contido nos resultados deste trabalho.

4.5 Cronograma de Atividades

A Figura 9 apresenta todas as atividades previamente descritas na seção 4.2, conferindo um *timeline* para a realização desse projeto.

4.6 Resumo do Capítulo

Neste capítulo, foi detalhada a classificação da pesquisa, especificando sua abordagem, natureza, objetivos, procedimentos técnicos e técnicas de coleta de dados. Em seguida, tem-se um detalhamento do fluxo de atividades, sendo que duas atividades foram expandidas para outras subseções, por serem mais abrangentes e demandarem a definição de subprocessos. No caso, um subprocesso, apresentado na seção 4.3, para detalhamento das atividades que conduzem o desenvolvimento da solução, e outro subprocesso, apresentado na seção 4.4, para detalhamento das atividades que conduzem a análise de resultados. Por fim, tem-se o cronograma das atividades de trabalho descritas na seção 4.2.

Definir Tema	22days?	01/08/2018	30/08/2018
Realizar Levantamento Bibliográfico	22days	01/08/2018	30/08/2018
Elaborar Proposta Inicial	22days	01/08/2018	30/08/2018
Definir Referêncial Teórico	20days	01/08/2018	28/08/2018
Definir Suporte Tecnológico	22days	01/10/2018	30/10/2018
Definir Metodologia de Pesquisa	22days	01/10/2018	30/10/2018
Definir Proposta de Aplicação	22days	01/11/2018	30/11/2018
Descrever Prova de Conceito	22days	01/11/2018	30/11/2018
Refinar Monografia	22days	01/11/2018	30/11/2018
Apresentar TCC1	22days	01/11/2018	30/11/2018
Realizar Correções da Banca	36days	13/03/2019	01/05/2019
Definir Solução Inicial	28days	01/05/2019	07/06/2019
⊟**Definir Resultados**	39days?	**06/05/2019**	**27/06/2019**
⊟**Ciclo 1**	35days?	**06/05/2019**	**21/06/2019**
Criar Questionário	1day?	06/05/2019	06/05/2019
Criar Cenários de uso	1day?	07/05/2019	07/05/2019
Executar Cenários de uso	3days?	10/05/2019	14/05/2019
Relatar Impressões da Observação Participante	1day?	15/05/2019	15/05/2019
Desenvolver Melhorias/Evoluções	10days	06/05/2019	17/05/2019
Descrever Resultados	1day?	21/06/2019	21/06/2019
⊟**Ciclo 2**	11days?	**20/05/2019**	**03/06/2019**
Criar Questionário	1day?	20/05/2019	20/05/2019
Criar Cenários de uso	1day?	22/05/2019	22/05/2019
Executar Cenários de uso	2days	24/05/2019	27/05/2019
Relatar impressões da observação participante	1day?	27/05/2019	27/05/2019
Desenvolver Melhorias/Evoluções	10days	21/05/2019	03/06/2019
Descrever Resultados	1day?	31/05/2019	31/05/2019
⊟**Ciclo 3**	10days?	**03/06/2019**	**14/06/2019**
Criar Questionário	1day?	03/06/2019	03/06/2019
Criar Cenários de uso	1day?	04/06/2019	04/06/2019
Executar Cenários de uso	2days	07/06/2019	10/06/2019
Relatar impressões da observação participante	1day?	11/06/2019	11/06/2019
Desenvolver Melhorias/Evoluções	10days	03/06/2019	14/06/2019
Descrever Resultados	1day?	14/06/2019	14/06/2019
⊟**Ciclo 4**	9days?	**17/06/2019**	**27/06/2019**
Criar Questionário	1day?	17/06/2019	17/06/2019
Criar Cenário de Uso	1day?	18/06/2019	18/06/2019
Executar Cenários de uso	2days	21/06/2019	24/06/2019
Relatar impressões da observação participante	1day?	25/06/2019	25/06/2019
Desenvolver Melhorias/Evoluções	9days	17/06/2019	27/06/2019
Descrever Resultados	1day?	27/06/2019	27/06/2019
Apresentar Trabalho	1day?	08/07/2019	08/07/2019

Figura 9 – Cronograma de atividades do trabalho

5 Factbox

Este capítulo apresenta detalhes sobre a ferramenta *online*, Factbox, fruto desse trabalho. O capítulo está organizado em seções. Na seção 5.1, encontra-se a contextualização, com dois focos de atenção, sendo: apresentação de possíveis concorrentes, e apresentação da prova de conceito. Particularmente, na prova de conceito, feita pelo autor, procurou-se acordar aspectos que se mostraram como os primeiros obstáculos ao sucesso da ferramenta de Gerência de Requisitos. Assim, tem-se uma primeira proposta de gerenciamento de requisitos, chamada de Urutau. A Urutau permitiu levantar várias preocupações, as quais guiaram o desenvolvimento da ferramenta Factbox, sendo essa última uma evolução significativa em relação à primeira. Na seção 5.2, tem-se um maior olhar à ferramenta Factbox, dedicada à Gerência dos Requisitos, orientando-se por dois aspectos chave: Rastreabilidade e Versionamento. Não menos importantes, outros tópicos também são tratados nesse capítulo. Dentre eles, destaca-se uma visão geral da ferramenta Factbox, seguida de um detalhamento arquitetural. Esse detalhamento foi apresentado em diferentes níveis de abstração, visando facilitar a compreensão. Por fim, tem-se o resumo do capítulo.

5.1 Contextualização

Como uma primeira motivação para realização desse projeto pensou-se no desenvolvimento de uma ferramenta de requisitos *online*, extensível a alguns tipos de artefatos (ex. Histórias de Usuário e *Rich Pictures*), com foco em Rastreabilidade e Versionamento. A intenção é permitir aos desenvolvedores trabalhar com requisitos de uma forma distinta das possibilidades existentes, apoiando-se em manter os rastros entre a *baseline* de requisitos e outros artefatos em tempo de *design*, implementação e teste; além de permitir controlar as versões dos artefatos gerados nesse processo.

As próximas colocações constituem uma exposição das soluções encontradas atualmente no mercado, focadas nesses aspectos, bem como levantadas pelo autor. Adicionalmente, é feita uma breve comparação entre essas iniciativas de mercado e a ferramenta Factbox.

5.1.1 Trabalhos Relacionados

A Tabela 2 exibe algumas das ferramentas mais populares para Gerência dos Requisitos, classificadas em relação ao atendimento dos principais aspectos que impactam a Gerência dos Requisitos. Apesar das ferramentas atenderem aos critérios de Rastrea-

bilidade e Versionamento, cabe enfatizar que fazem de uma forma individual, através do uso de referências e etiquetas entre artefatos, sem uma visualização conjunta do projeto, como a proposta pelo Factbox.

Nome da ferramenta	Extensibilidade	Rastreabilidade	Versionamento	Licença
Helix RM	Não atende	Matriz	Não gráfico	Não livre
Orcanos	Não atende	Matriz	Não gráfico	Não livre
Jira	Não atende	Link	Não gráfico	Não livre

Tabela 2 – Tabela de critérios atendidos por ferramentas de proposta similar ao Factbox

Figura 10 – Exemplo de matriz de Rastreabilidade do Helix

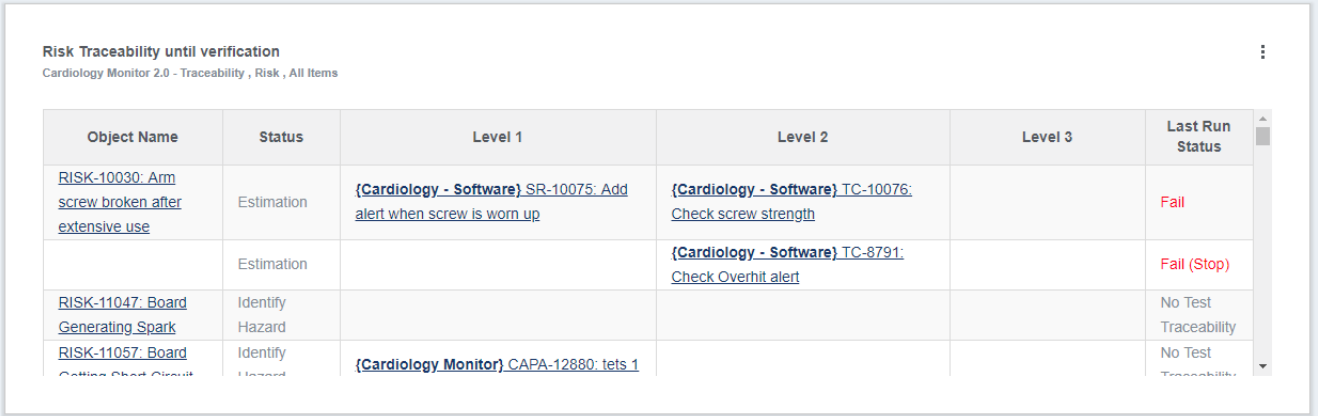

Figura 11 – Exemplo de matriz de Rastreabilidade do Orcanos

Na Figura 10 e Figura 11, é possível ver a matriz de Rastreabilidade dos Requisitos gerenciados nas ferramentas. Já o Jira, ilustrado na Figura 12, não utiliza desse recurso, apenas referenciando via link os tipos suportados.

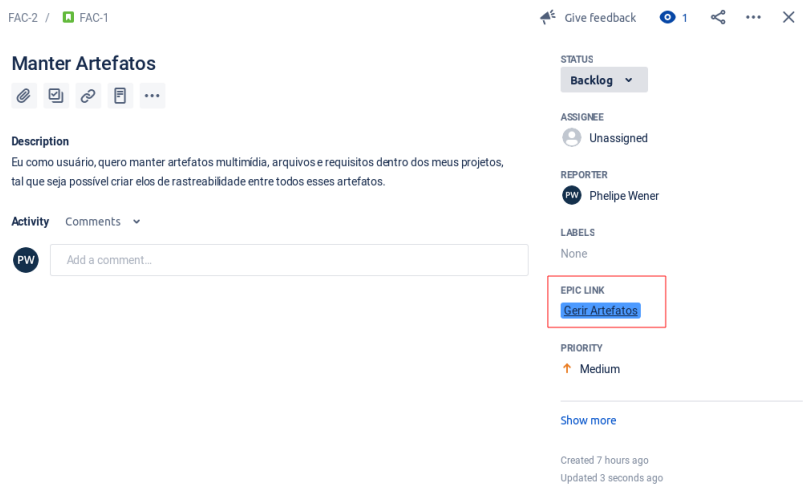

Figura 12 – Exemplo de Rastreabilidade do Jira

Cabe ressaltar que, em um primeiro momento, e visando oferecer uma ferramenta mais aderente às necessidades de Gerência dos Requisitos, com Rastreabilidade, Versionamento e ainda elaborada para ser extensível, optou-se por desenvolver uma prova de conceito. A intenção era explorar os aspectos chave para o sucesso da solução definitiva.

5.1.2 Prova de Conceito

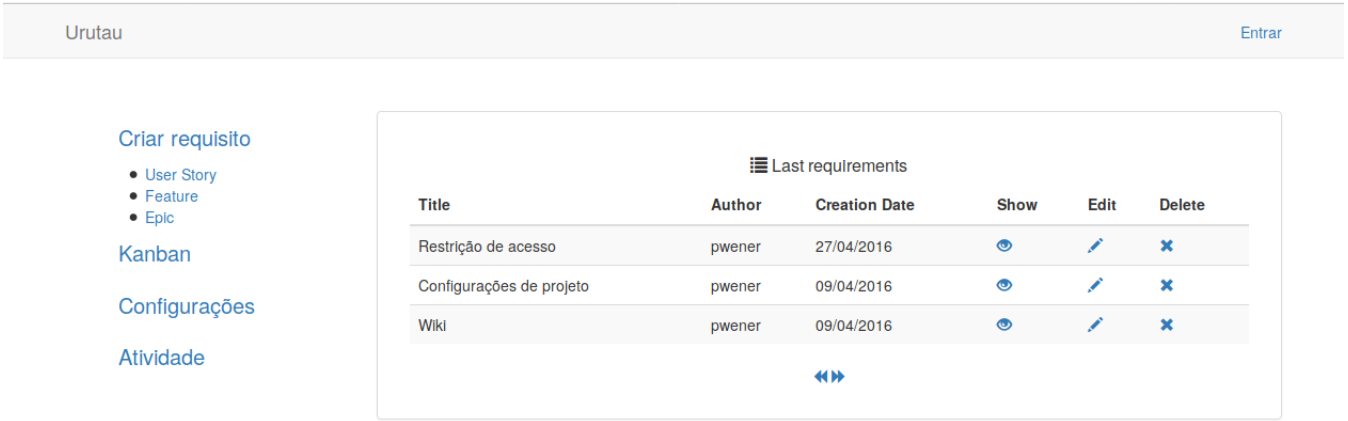

Figura 13 – Tela da *homepage* de um projeto mantido no Urutau

A ideia começou a ser trabalhada em um projeto acadêmico chamado Urutau. Inicialmente, esse projeto envolveu um grupo de alunos de uma disciplina do curso de Engenharia de Software, da Universidade de Brasília, campus Gama-DF. Entretanto, ao término dessa disciplina, apenas um contribuidor proveu continuidade ao projeto, sendo esse o autor do presente projeto. O projeto Urutau era uma solução de software que buscava gerenciar alguns tipos de artefatos, como Histórias de Usuário, Épicos e *Features*. Na Figura 13, é possível visualizar a quantidade de artefatos tratada pelo projeto em sua

última versão. Basicamente, a ferramenta tinha suporte às operações básicas, como criar, apagar e editar os artefatos suportados.

As decisões arquiteturais tomadas na época, e que sustentaram o desenvolvimento dessa primeira versão, dificultaram a extensibilidade dessa ideia. Esse, portanto, foi o principal fator que resultou na descontinuidade do projeto. Visando elucidar essas dificuldades, seguem outras colocações, retomadas na seção 5.4, na qual é acordada a arquitetura proposta para a nova solução computacional.

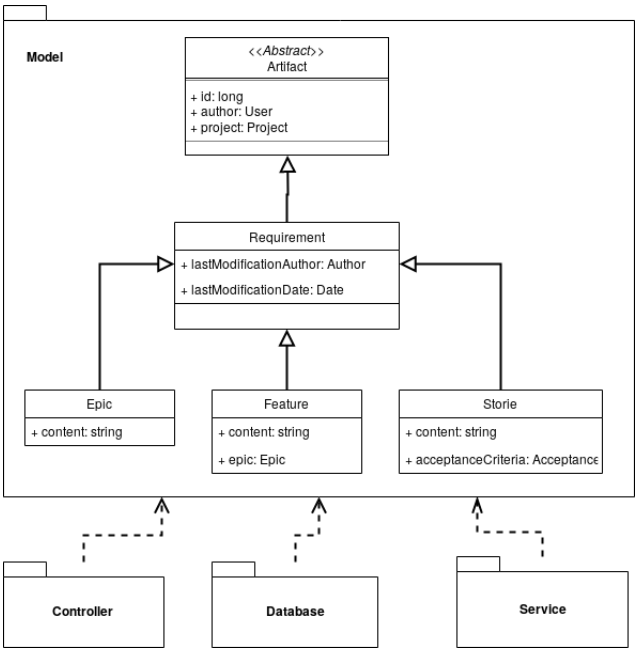

Figura 14 – Diagrama UML de pontos da arquitetura do Urutau

Conferindo maior detalhamento sobre a primeira versão do Urutau, na Figura 14, tem-se, usando a notação da UML (parte diagrama de classes e parte representada em pacotes), as principais relações entre os artefatos, requisitos e demais elementos arquiteturais do sistema projetado na época.

A classe **Artifact** foi criada como a classe mais genérica para representação dos artefatos do sistema. Entende-se por artefatos desde os próprios requisitos - foco da solução - até mesmo tipos de multimídia e outros, a depender da necessidade do usuário. A classe **Requirement** teve seu desenho planejado para artefatos, cujo conteúdo é exclusivamente gerido dentro do sistema. Os demais pacotes podem ser representados com relações de dependência com as classes mais concretas dessa hierarquia. Aqui, tem-se o uso de um princípio da Orientação a Objetos, no caso, polimorfismo, permitindo que as classes concretas *Epic*, *Feature* e *Storie* especializem a parte comportamental definida em **Requirement**, sobrescrevendo métodos. Por fim, os pacotes extras, *Controller*, *Database* e *Service*, fazem uso dessas classes mais especializadas via relações de dependência.

Dentre outros problemas, a ideia de um *plugin* que tratasse um novo tipo de

requisito tornou-se difícil, devido, principalmente, ao alto acoplamento que a herança mais as fortes relações de dependência geraram nessa solução. Portanto, caso o sistema tivesse de prover apoio a um novo artefato, esse demandaria ser implementado diretamente na ferramenta, não conferindo maior flexibilidade aos interessados em estender essa solução. Tal solução acoplada, agravada à dificuldade de reutilização, não permitia extensibilidade, sendo esse um dos princípios desejados na solução. A ferramenta tinha em seu *Roadmap*, um projeto de Rastreabilidade e Versionamento, mas foi descontinuada antes que isso pudesse ser implementado.

Diante do exposto, e procurando uma solução computacional mais aderente às necessidades do projeto e do domínio, buscou-se uma nova solução computacional, cuja visão geral é apresentada na próxima seção.

5.2 Factbox: Uma Ferramenta de Gerência dos Requisitos

As próximas seções procuram abordar a nova solução computacional, sendo essa uma ferramenta de Gerência dos Requisitos, chamada Factbox, com foco em Rastreabilidade e Versionamento dos requisitos. Cabe ressaltar que esses requisitos podem ser funcionais ou não funcionais. Na subseção 5.2.1, confere-se uma visão geral dessa ferramenta. Já na seção seção 5.4, é apresentada a visão arquitetural da mesma, com diferentes níveis de abstração. Por fim, têm-se as considerações finais do capítulo.

5.2.1 Visão Geral da Ferramenta Factbox

Na expectativa de atender diversos artefatos oriundos das atividades da Engenharia de Requisitos, a ferramenta Factbox fornece extensibilidade através de *plugins*, que são, nativamente, parte da arquitetura proporcionada pelo *Ruby on Rails*. Como uma primeira versão para a ferramenta Factbox, esses *plugins* concentram-se em oferecer suporte aos artefatos citados na seção 2.3, sendo esses: *Rich Pictures* e Histórias de Usuários.

Com o intuito de conferir maior detalhamento sobre a nova solução, tem-se, na Figura 15, um grafo mais concreto de como é tratada a questão, por exemplo, de Rastreabilidade de artefatos. A ligação entre os artefatos e suas fontes é conferida através das setas contínuas, sendo possível rastrear a fonte em versões específicas. Nesse sentido, os requisitos como "Controle de Acesso", "Configurações"e "Manter Artefatos"por exemplo, são oriundos do artefato "Rich Picture".

O Versionamento é controlado automaticamente, com base nas modificações realizadas nos artefatos rastreados. Cada alteração gera um novo artefato com referência ao artefato de origem. No grafo da Figura 15, as versões passadas de um artefato são ilustradas na cor cinza e ligadas pela seta pontilhada.

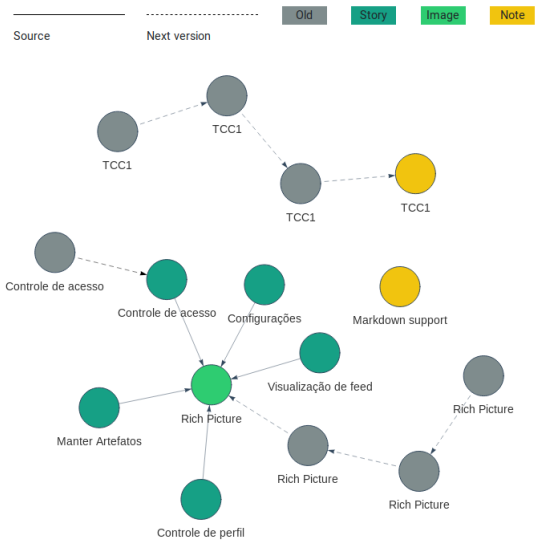

Figura 15 – Exemplo de grafo de Rastreabilidade

5.3 Detalhamento dos Requisitos da Ferramenta Factbox

Inicialmente, o *backlog* do produto, o qual reunia os requisitos da ferramenta Factbox, foi estabelecido via Kanban físico. A Figura 16 ilustra as primeiras Histórias de Usuário especificadas na época. Na primeira versão da ferramenta(Beta) para uso, as histórias de *Feed* e Configurações de Sistema não estavam implementadas, nem priorizadas para as próximas *Sprints*, pois ambas não afetavam a proposta central da ferramenta.

A Figura 17 apresenta as Histórias de Usuário do Kanban físico (vide Figura 16), mas sendo gerenciadas já na Factbox. Para visualizar a última versão dos requisitos e artefatos, acessar: <https://factbox.app/projects/Factbox>

5.4 Visão Arquitetural da Ferramenta Factbox

A seguir, será apresentada a arquitetura de software elaborada para a ferramenta Factbox. Optou-se por apresentar essa arquitetura em 4 níveis, começando por uma visão mais generalizada da aplicação (nível 1), e aprofundando-se nos demais níveis (2 a 4) para um detalhamento cada vez mais específico.

5.4.1 Nível 1

Com base na notação SADT (LAKHOUA; ANNABI, 2006), a Figura 18 procura ilustrar as entradas, os controles, as saídas e os mecanismos, considerando a solução como uma caixa preta.

Como entradas, têm-se os requisitos formais, bem como todo artefato que o usuá-

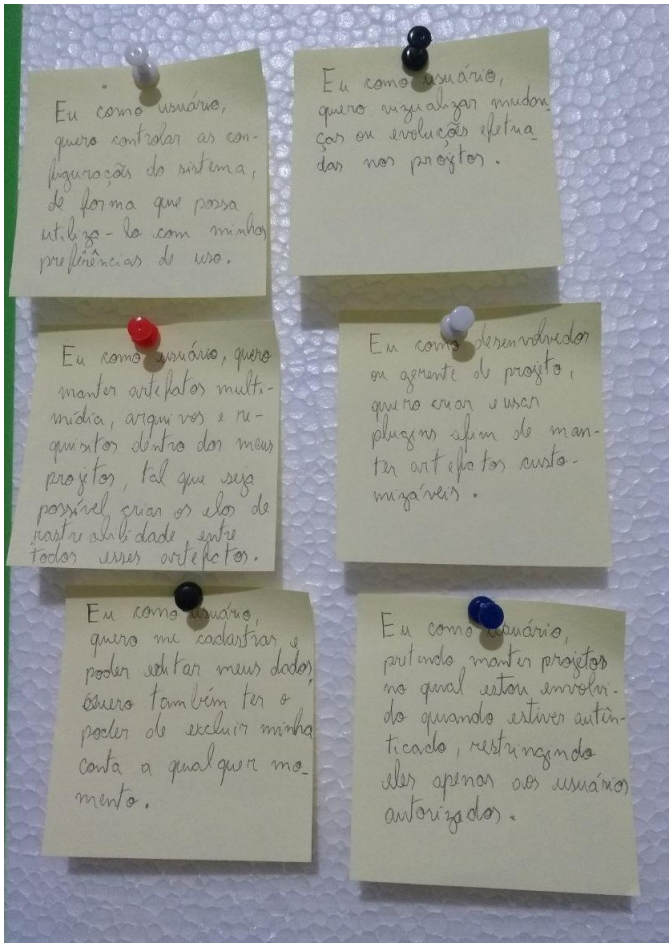

Figura 16 – Kanban físico com o backlog da ferramenta Factbox

rio deseje rastrear como fonte de informação. Como controles, podem ser destacados o domínio, o qual o usuário pretende modelar, bem como o processo de desenvolvimento, o qual orienta o desenvolvimento do projeto naquele dado domínio. Dentre os mecanismos, destacam-se o usuário (Engenheiro de Requisitos) e o Suporte Tecnológico apresentado no Capítulo 3. Por fim, como saídas, têm-se os requisitos gerenciados e demais informações quanto à Rastreabilidade e ao Versionamento dos mesmos.

5.4.2 Nível 2

Revelando um pouco mais as informações encapsuladas na Figura 18, é possível detalhar a solução, considerando um segundo nível de abstração arquitetural. A Figura 19 mostra o fluxo do *core* do Factbox, ilustrando os principais elementos do *framework* Rails, as principais bibliotecas utilizadas e outros componentes como o Banco de Dados e o *Web Server*.

A arquitetura do Rails é baseada no padrão *Model-View-Controller*. A Figura 19 apresenta esses três elementos em conjunto com outros dispositivos que possibilitam o uso da ferramenta através de um navegador.

Manter Artefatos

Story

Eu como usuário, quero manter artefatos multimídia, arquivos e requisitos dentro dos meus projetos, tal que seja possível criar elos de rastreabilidade entre todos esses artefatos.

Acceptance criteria

- Suporte a imagem, áudio e links.
- Deve ser possível manter os artefatos fornecidos via plugin
- O sistema deve prover suporte a plugins que darão suporte a um ou mais tipos de Requisitos de Software
- Cada artefato deve poder citar um ou mais artefatos como fonte de informação.
- Um gráfico de rastreabilidade deve ser gerado ao final entre todos os artefatos de um projeto.

Controle de perfil

Story

Eu como usuário, pretendo manter meus dados através do cadastro e depois em poder edita-los, quero também ter o poder de excluir minha conta a qualquer momento.

Acceptance criteria

- O usuário deve ter login e email únicos.
- O usuário deve ter suporte a mudança de qualquer um de seus dados, tal que para mudança de email e senha, sejam tomadas devidas cautelas.
- Uma página para gerencia de conta é necessária para alteração de dados e exclusão de contas.

Controle de acesso

Story

Eu como usuário, pretendo manter projetos no qual estou envolvido quando estiver autenticado, restringindo eles apenas aos usuários autorizados.

Acceptance criteria

- A senha do login deve ter no mínimo 6 caracteres.
- O usuário deve poder convidar outros usuários para seu projeto.
- O usuário deve conseguir autenticar com login ou email escolhido no cadastro.

Configurações

Story

Eu como usuário, quero controlar as configurações do sistema, de forma que possa utiliza-lo com minhas preferências de uso.

Acceptance criteria

- O usuário deve conseguir alterar o tema do sistema, a principio normal e dark.
- Internacionalização, a principio inglês e pt-br
- Deve ser possivel alterar os dados de usuário criados no cadastro.
- Controle de que tipo de notificações o usuário quer receber.

Visualização de feed

Story

Eu como usuário, quero visualizar mudanças ou evoluções efetuadas nos projetos.

Acceptance criteria

- Mudanças a ser visualizadas: adição, atualização e deleção de artefatos, inclusão de membros, criação de elos de rastreabilidade.
- O feed tem que ser paginado.

Plugins

Story

Eu como usuário quero criar plugins afim de manter artefatos customizáveis.

Acceptance criteria

- Um guia de como criar plugins é necessário

Figura 17 – Histórias de Usuários mantidas na ferramenta Factbox.

O fluxo pode ser entendido começando pelo *Web Server*, que se trata de uma aplicação de terceiros, responsável por servir via protocolo *HTTP* a solução Factbox. Em seguida, o *framework* Rails tem um mecanismo de redirecionamento de rotas. Cada rota deve levar a uma funcionalidade diferente, mediante a lógica da camada controladora responsável. Algumas podem conter funções complexas de visualização, como é o caso da Figura 15. Para tal, foi utilizado um recurso de integração do Rails chamado *Sprockets*,

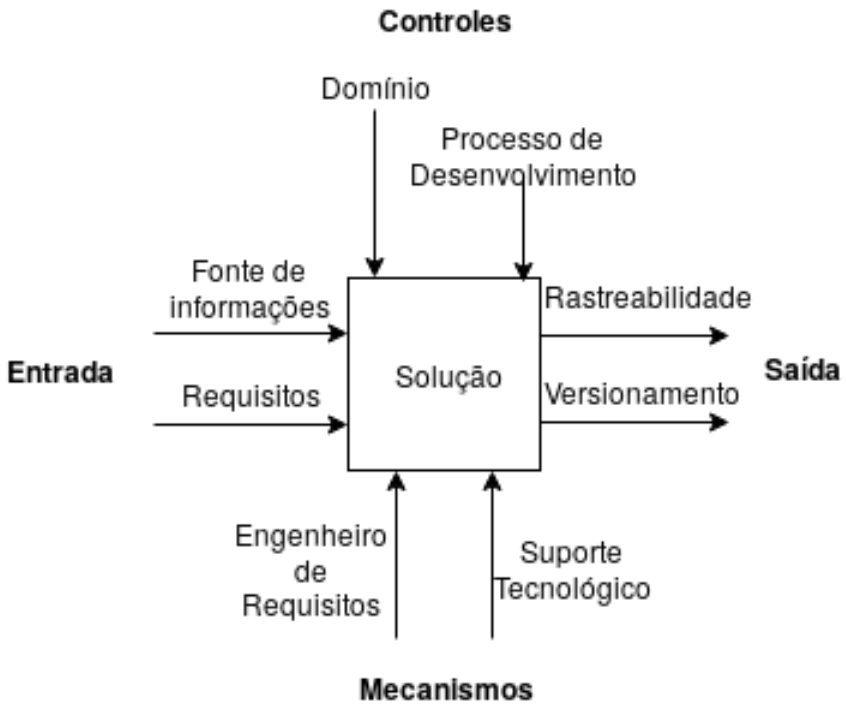

Figura 18 – Visão geral da ferramenta Factbox - Nível 1 da Arquitetura - Notação SADT

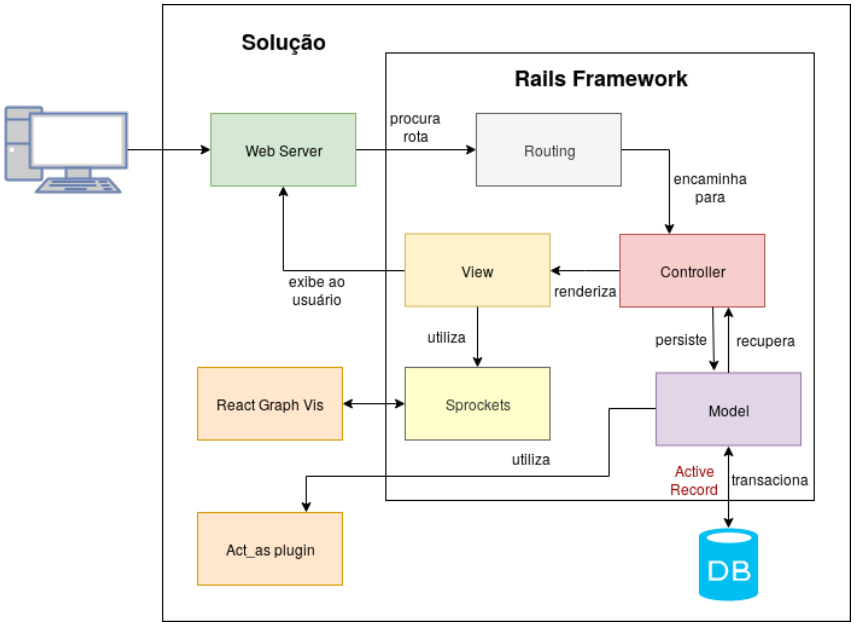

Figura 19 – Visão arquitetural da ferramenta Factbox - Nível 2 - Principais Componentes

que permite o uso de outras tecnologias na camada de *Views*.

Por último, e não menos importante, existem as *Models*, abstrações do contexto que servem de interface para operações no banco de dados. Por uma demanda específica na modelagem da solução, explicado com mais detalhes no Nível 4, um *plugin act-as* é utilizado.

5.4.3 Nível 3

Ainda em termos de arquitetura, têm-se o detalhamento da ferramenta Factbox, em um Nível 3, sendo esse focado na arquitetura dos *plugins*. Segundo (GUIDE, 2018a), o *Framework* Rails tem nativamente suporte a *plugins*, sendo estes extensões ou modificações do *core* da solução. Na solução proposta, essa dinâmica dos *plugins* será utilizada para apoiar novos tipos de artefatos a serem gerenciados na ferramenta. A Figura 20 ilustra os principais componentes que compõem um *plugin*.

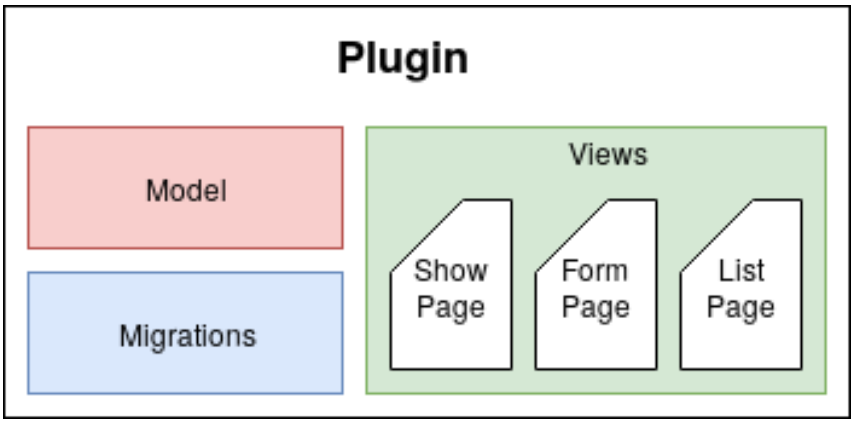

Figura 20 – Visão arquitetural da ferramenta Factbox - Nível 3 - *Plugins*

Basicamente, a *model* refere-se à classe modelo do artefato a ser gerenciado; as *migrations* são classes com comandos para criar ou modificar o sistema em termos de banco de dados e, por fim, as *views* são as páginas utilizadas para o usuário interagir com o novo tipo de artefato.

5.4.4 Nível 4

Visando apresentar a visão lógica da arquitetura da Ferramenta Factbox, de modo a esclarecer o funcionamento dos *plugins* de extensão da solução, tem-se a elaboração do diagrama de classes, com base nas principais entidades envolvidas, considerando um olhar mais focado nos artefatos aceitos na primeira versão da ferramenta. No caso, têm-se: notas, áudio, imagens, histórias de usuário e critérios de aceitação. Optou-se por apresentar as entidades em inglês, pois a ferramenta está sendo implementada no mesmo idioma da linguagem de programação usada para essa codificação. Trata-se de uma boa prática, segundo autores como (MARTIN, 2008).

O diagrama de classes da Figura 21 ilustra os diferentes tipos de artefatos atendidos pela Ferramenta Factbox. A classe **Artifact** é a base para o Versionamento e a Rastreabilidade dentro do sistema. Diferentemente do que é intuitivamente esperado (i.e. relações de herança ou ainda composições), têm-se relações de associação bidirecionais.

Tal prática permitiu lidar com problemas acordados desde o desenvolvimento da prova de conceito (no caso, Urutau), a qual foi discutida na 5.1.2.

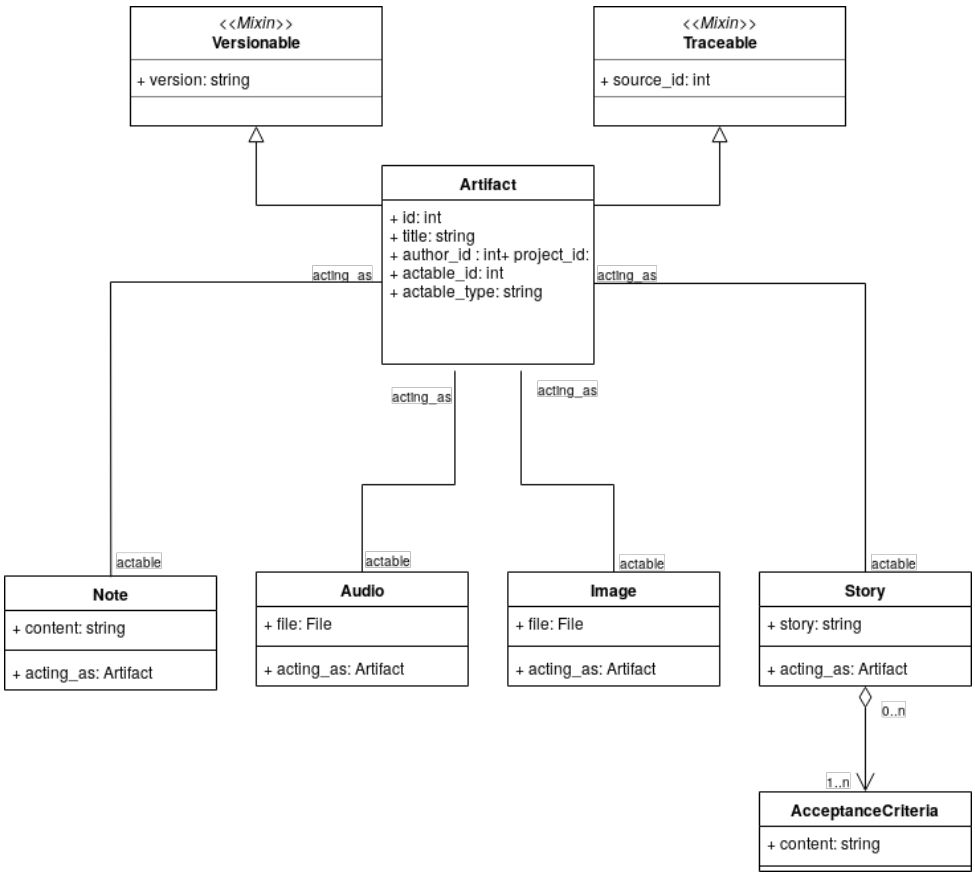

Figura 21 – Visão arquitetural da ferramenta Factbox - Nível 4 - Diagrama de Classes para os Modelos de Artefatos Suportados

Consultando a literatura, em especial (LARMAN, 2007), percebe-se que essa forma de relacionar as entidades, conforme acordado na Figura 21, permite mais facilmente a criação de *plugins*. Isso ocorre porque a associação com **Artifact** permite uma dependência de baixo acoplamento, a qual confere fácil extensibilidade.

```
1   # Shows artifact with specific id
2   # GET /artifacts/:id
3   def show
4     # Search through activerecord one specific artifact
5     @artifact = Artifact.find(params[:id])
6
7     # Rails automatically finds one view named show to artifacts
8     render :show
9   end
```

Figura 22 – Exemplo de método *show* no controlador de Artefatos, caso utilizasse herança

Utilizando herança, com algumas linhas de código, seria possível generalizar o tratamento dos artefatos. O código da Figura 22 mostra a praticidade do método *show* no controlador de artefatos, uma vez que usando o padrão de projeto *Active Record*, nativo

no *Ruby on Rails*, bastaria invocar um registro do banco, enviando o objeto resultante para uma *view* que estivesse preparada para receber os seus atributos.

```
1   # Shows artifact with specific id
2   # GET /artifacts/:id
3   def show
4     @artifact = Artifact.find(params[:id]).specific
5
6     # Render specific artifact
7     render '#{@artifact.type}/show'
8   end
```

Figura 23 – Método *show* no controlador de Artefatos, utilizando associações polimórficas

Na decisão de usar associação, no *Ruby on Rails*, é possível usufruir dessas, utilizando o recurso de associações polimórficas. Segundo o (GUIDE, 2018b), associações polimórficas permitem que uma classe modelo possa pertencer a uma ou mais classes modelos através de uma simples associação. Esse recurso é utilizado na solução através do *plugin* da (COMPUTING, 2010), que encapsula métodos e atributos para lidar com o polimorfismo da associação resultante. A Figura 23 é um exemplo de código utilizado com associação polimórfica.

5.5 Versionamento e Rastreabilidade na Ferramenta Factbox

Como comentado em capítulos anteriores, para se ter uma boa Gerência dos Requisitos, tem-se a necessidade de prover suporte adequado ao Versionamento e à Rastreabilidade. As próximas colocações focam em apresentar como isso é tratado na Ferramenta Factbox.

5.5.1 Versionamento na Ferramenta Factbox

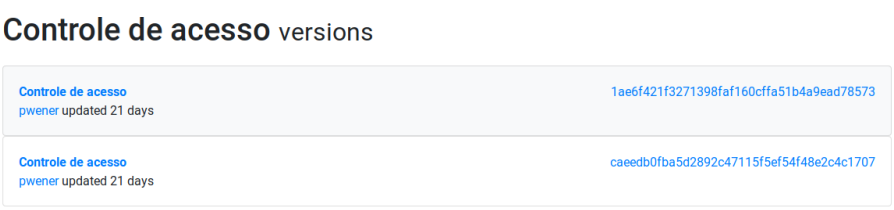

Figura 24 – Visualização do histórico de versões de um artefato

Na Figura 24, é apresentado um histórico de versões. A cada atualização de um artefato, automaticamente, é criada uma nova versão, identificada por uma *hash* única e gerada a partir de todos atributos do objeto. Essa mesma estratégia é utilizada pelo

Git para assegurar integridade (BURGDORF, 2014). Na Ferramenta Factbox, a *hash* de Versionamento é utilizada para acessar um artefato em sua versão específica.

```ruby
require 'digest'

# This module controls version features of artifact
module Versionable
  extend ActiveSupport::Concern

  # Should include this follow properties inside the Artifact
  included do
    validates :version, presence: true
    has_many :votes
  end

  # Method used when artifact is updated
  # Basically, it update the version attribute
  def generate_version
    self.version = Digest::SHA1.hexdigest to_json
  end

  # Method used to discontinue one artifact.
  # When an artifact is created, the old keeps as a separated register.
  # To identifier which is the newest, this property should be true.
  def discontinue
    self.last_version = false
  end
end
```

Figura 25 – Código de geração de Versionamento

É possível ver a implementação do Versionamento na Figura 25, previamente tratada como um *mixin*, no diagrama de classes da Figura 21.

5.5.2 Rastreabilidade na Ferramenta Factbox

Conforme adiantado na subseção 5.2.1, a Ferramenta Factbox lida com Rastreabilidade utilizando elos de Rastreabilidade, os quais resultam no uso do atributo *origin* de *Artifact*, uma vez que ele aponta para a fonte de informação de um novo artefato.

A Figura 26 é referente a parte do código que gera os nós e elos do grafo. Todos os artefatos são iterados, adicionando assim o elos de versionamento, onde cada artefato é referenciado aos seus artefatos filhos, e rastreabilidade, onde cada artefato é referenciado a sua fonte de informação. Por fim, cada artefato é adicionado a uma lista com o retorno da chamada de método *node-options*, que retorna apenas as propriedades necessárias para o grafo, como nome, cor e identificador de cada nó.

```
# GET /traceability/:name
def traceability
  artifacts = Project.find_by_name(CGI.unescape(params[:name])).artifacts

  @nodes = []  Lista com todos artefatos
  @edges = []  Lista com todos os nós entre os artefatos

  artifacts.each do |a|  Itera em todos artefatos do projeto em questão
    # append edge of children to source
    unless a.children.empty?
      a.children.each do |children|
        edge = { from: children.id, to: a.id }
        @edges.push edge          Adiciona o elo de rastreabilidade por versão
      end
    end

    unless a.origin_artifact.nil?
      edge = { from: a.origin_artifact.id, to: a.id, dashes: true }
      @edges.push edge          Adiciona o elo de rastreabilidade da fonte de informação
    end

    # save current artifact as a node
    @nodes.push a.node_options    Adiciona o artefato com as informações necessárias para o nó
  end

  render 'traceability'
end
```

Figura 26 – Código de criação dos elos de Rastreabilidade

5.5.3 Manter Artefatos

Conforme previamente explicado, a ferramenta Factbox tem, por padrão, suporte a notas textuais e imagens. Com a instalação de *plugins*, é possível estender a capacidade da ferramenta Factbox, possibilitando o uso de outros artefatos, por exemplo, Histórias de Usuários. Tal extensibilidade é evidenciada na Figura 27.

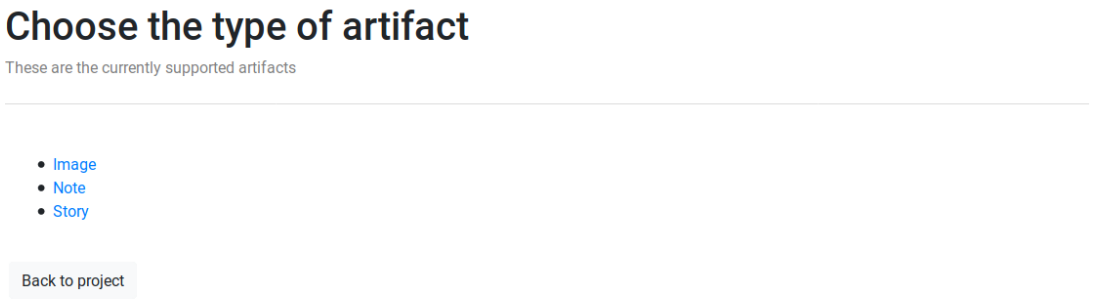

Figura 27 – Menu para escolha de criação de artefatos.

Todos os artefatos são geridos, genericamente, através da controladora da classe *Artifact*, exibida em Figura 21. Conforme detalhado em seção 5.4, o uso de associações polimórficas permite o tratamento de novos artefatos tal que seja possível realizar as operações básicas de criação, edição e exclusão através de um mesmo controlador.

5.6 Resumo do Capítulo

Neste capítulo, foi apresentada a ferramenta Factbox, procurando: (i) acordar as origens dessa solução, apontanto trabalhos relacionados e ainda um primeiro suporte chamado Urutau; (ii) detalhar a nova solução - Ferramenta Factbox - em diferentes níveis arquiteturais e, por fim, (iii) mencionar o tratamento quanto às questões de Versionamento e Rastreabilidade.

Como destaques do capítulo, enfatiza-se o levantamento dos principais aspectos para o sucesso de uma ferramenta de Gerenciamento de Requisitos, no caso: Versionamento, Rastreabilidade e Extensibilidade (via Reutilização de Software). O estudo de trabalhos relacionados (ferramentas: Helix RM, Orcanos e Jira) e o desenvolvimento da Urutau permitiram perceber a relevância desses aspectos, conferindo ao autor maior embasamento na construção da Ferramenta Factbox.

6 Resultados Obtidos

Visando a evolução da ferramenta *online* Factbox, desenvolvida nesse projeto de pesquisa, foi aplicada uma modalidade de análise de resultados conhecida como pesquisa-ação. Esse capítulo está dedicado à apresentação dos ciclos iterativos e incrementais, inerentes ao uso de pesquisa-ação, e que permitiram evoluir a ferramenta Factbox com base nos *feedbacks* do público alvo da ferramenta. Na seção 6.1, é apresentado o público alvo utilizado nos ciclos de pesquisa-ação. Na seção 6.2, as atividades da pesquisa-ação são contextualizadas. Em seguida, os ciclos de pesquisa-ação são detalhados considerando coleta de dados com a análise e interpretação, plano de ação e resultados. Por fim, tem-se o resumo do capítulo.

6.1 Público-Alvo

Para realização da coleta de impressões, foram consideradas amostras do público alvo da ferramenta. No caso, têm-se dois grupos de amostragens: um grupo de cinco pessoas, de uma empresa de software chamada GoLedger, na qual, o autor deste trabalho atua no presente momento; e um grupo de alunos de graduação do curso de Engenharia de Software da Universidade de Brasília.

Cabe enfatizar que o projeto utilizado pela empresa GoLedger, na ferramenta Factbox, é real, em fase de desenvolvimento. Trata-se de uma ferramenta para suporte a gestão de redes *Blockchain*. O projeto tem cinco envolvidos, sendo: um *Product Owner*, três desenvolvedores e um gerente que também atua na implementação do projeto.

Já o projeto trabalhado por cinco alunos da UnB, trata-se de um projeto fictício, que se utiliza de atividades típicas dentro Engenharia de Requisitos. Tal projeto foi elaborado pelo autor, visando explorar situações comuns, com foco no uso em Rastreabilidade e Versionamento.

Outro grupo de alunos, da disciplina de Requisitos de Software da UnB, utilizou a ferramenta. Entretanto, não houve a participação dos mesmos nos ciclos de pesquisa-ação. Os resultados obtidos pelo grupo no uso da ferramenta estão disponíveis em: <https://requisitos-habitica.herokuapp.com/FactBox>

6.2 Atividades da Pesquisa-Ação

Como previamente visto na seção 4.4, a pesquisa-ação utilizada nesse trabalho conta com quatro atividades: Coleta de dados, Análise e Interpretação dos dados, Ela-

boração do plano de ação e Divulgação dos resultados. Na Figura 8, tem-se o uso dessas atividades, já adaptado ao escopo desse projeto.

Na Coleta de dados, são coletadas impressões dos usuários relatados na seção 6.1.Essa coleta garante insumos para a próxima atividade. Em Análise e Interpretação, os dados são plotados em gráficos, utilizando métricas, para prover uma análise quantitativa; e ainda representados também em gráficos, mais com foco nos *feedbacks* dos usuários, visando uma análise qualitativa. Na Elaboração do plano de ação, é realizado um planejamento com o propósito de solucionar os pontos investigados. Por fim, na Divulgação dos resultados, tem-se a versão mais evoluída da ferramenta Factbox, já com os incrementos por iteração, e orientando-se pelo plano de ações.

6.3 Primeiro Ciclo

Para Coleta de Dados, foi aplicado um questionário com quatro perguntas, disponível no Apêndice A. Próximo ao final de uma *Sprint* de desenvolvimento, o público descrito na seção 6.1 foi submetido ao formulário. Cabe ressaltar que o método de observação participante também foi utilizado, gerando uma série de pontos de melhoria observados pelo autor.

6.3.1 Coleta de dados, Análise e interpretação

Como parte das atividades de Coleta de Dados e Análise e Interpretação, tem-se os itens do Questionário, cada uma acordando uma motivação específica, bem como as análises e interpretações realizadas com base nos gráficos plotados. Por fim, e visando a apresentação sumarizada dos resultados obtidos no ciclo, tem-se um relatório de autoria do autor, proveniente de Observação Participante.

6.3.1.1 Questionário

A primeira pergunta, "Você considera a Engenharia de Requisitos importante para qualidade do produto de Software?", permitiu avaliar o perfil do público alvo, entendendo a relevância que o utilizador da ferramenta Factbox creditava às atividades atuantes na área de pesquisa do projeto.

A Figura 28 traz o gráfico resultante das respostas da primeira pergunta. Com a alta relevância conferida pelos participantes à Engenharia de Requisitos, os envolvidos confirmaram-se como boas amostragens do público alvo da ferramenta Factbox.

A segunda pergunta, "Você considera que a ferramenta cumpre com a proposta de manter artefatos e requisitos?", permitiu avaliar a impressão dos participantes quanto as tarefas mais básicas do Factbox, relativas ao requisito "manter um artefato", com foco em:

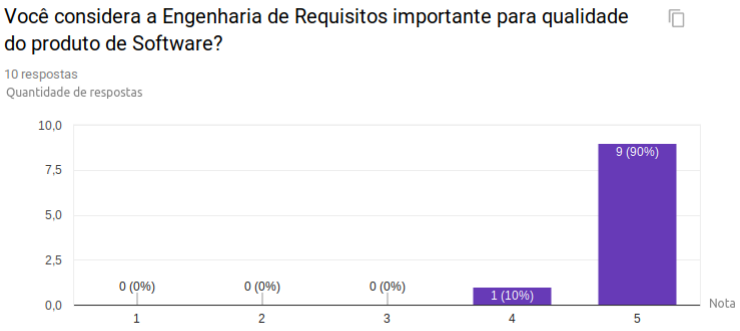

Figura 28 – Primeira pergunta do Questionário do primeiro ciclo de Pesquisa-ação

criar, atualizar e excluir um artefato. Com base na Figura 29, acredita-se que a ferramenta cumpre com as funcionalidades básicas, no que se refere aos artefatos e requisitos.

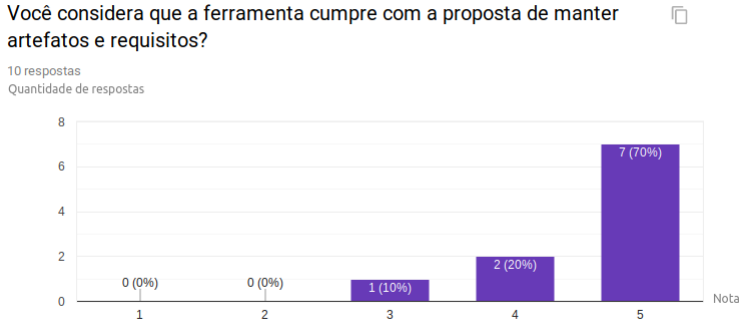

Figura 29 – Segunda pergunta do Questionário do primeiro ciclo de Pesquisa-ação

A terceira pergunta, dessa vez dissertativa, "Você teria uma sugestão de nova funcionalidade para a ferramenta?", permitiu coletar algumas sugestões que poderiam compor o *backlog*. Na Figura 30, tem-se as respostas obtidas.

A quarta pergunta, "De acordo com a sua interação, quais os pontos acha que podem melhorar?", permitiu detectar pontos de melhoria e correção na ferramenta. Na Figura 31, tem-se os pontos levantados, sendo alguns considerados para compor o *backlog* da *sprint*.

6.3.1.2 Observação Participante

Durante a iteração dos usuários, uma série de observações foram realizadas pelo autor, muitas das quais coincidem com as impressões capturadas pelo questionário. Algumas observações foram anotadas de maneira informal, sendo expressas aqui de maneira resumida.

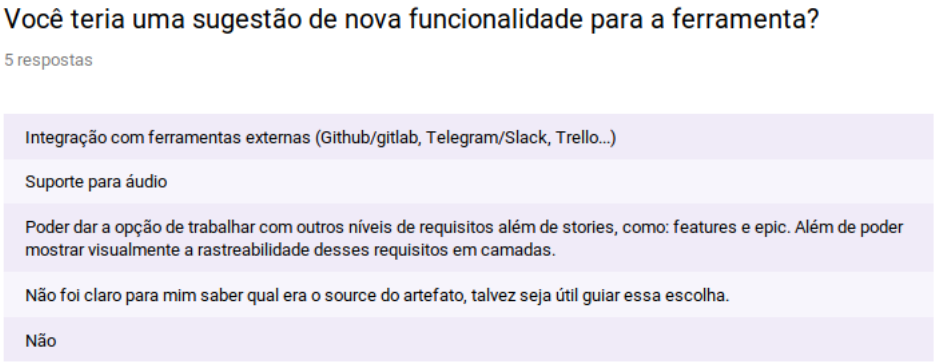

Figura 30 – Terceira pergunta do Questionário do primeiro ciclo de Pesquisa-ação

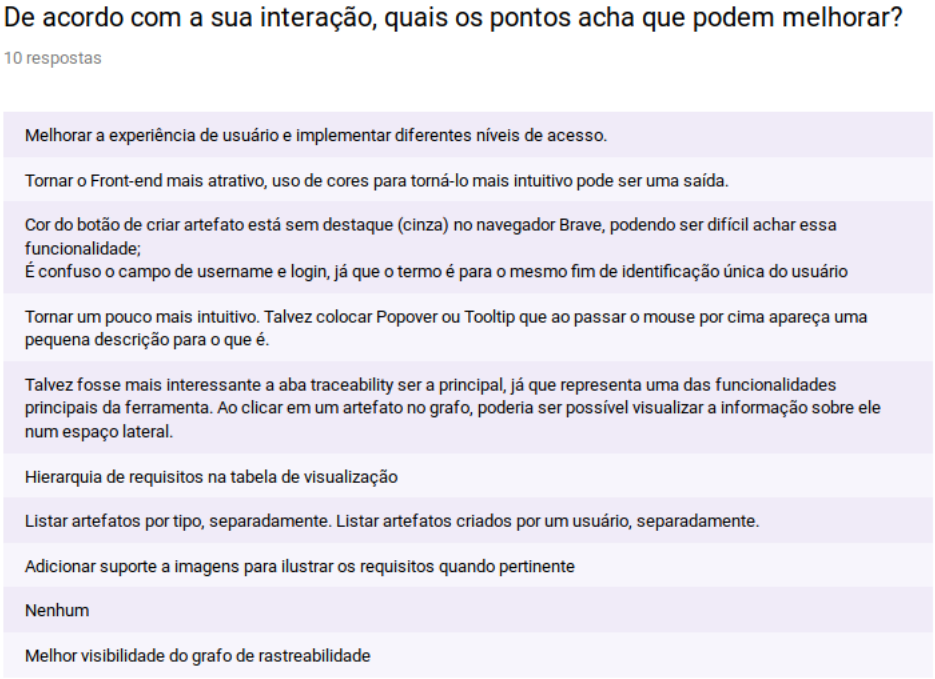

Figura 31 – Terceira pergunta do Questionário do primeiro ciclo de Pesquisa-ação

O problema mais evidente da ferramenta Factbox, durante a primeira iteração, foi quanto às questões de usabilidade. Em especial, na página de visualização do projeto, os usuários ficaram confusos quanto os passos necessários para cumprir seus objetivos. O botão de criar um novo artefato, por exemplo, não foi visto por grande parte dos usuários.

A iteração inicial de ambos os grupos consistiu basicamente de cadastrar novos artefatos, criando imagens, notas e histórias de usuário de acordo com os projetos utilizados. Na criação desses artefatos, o atributo(*source*), que indica a fonte de informação de um artefato para outro, não ficou claro também, precisando de uma intervenção do autor

para explicar sua utilidade.

6.3.2 Plano de Ação

Com a análise disposta na seção anterior, foi possível levantar uma série de pontos de melhoria. Nesse primeiro ciclo, ficou evidente uma série de questões de usabilidade, como:

- Disposição dos items na página de visualização do projeto, incluindo a visualização do grafo de Rastreabilidade, o qual os usuários não conseguiram localizar, e
- Informações sobre o campo *source* na criação de um artefato, uma vez que esse pareceu incompreensível aos usuários.

Esses dois itens foram escolhidos para correção durante a *sprint* de desenvolvimento.

6.3.3 Divulgação dos Resultados

Com a implementação dos items levantados na seção anterior, tem-se as seguintes alterações:

- Mudança do *layout* da página, conforme Figura 32;
- Mudança do botão de "Criar Artefato"como item do menu, conforme Figura 32;
- Renderização do grafo de Rastreabilidade no *layout* da página inicial do projeto, conforme Figura 33;
- Atualização do campo *source* em todos os artefatos. Na Figura 34, o campo aparece com mais informações para facilitar o entendimento do usuário, e
- Correções de *bugs* e outras pequenas melhorias, disponíveis em <https://github.com/factbox/factbox/releases/tag/v1.0.1>

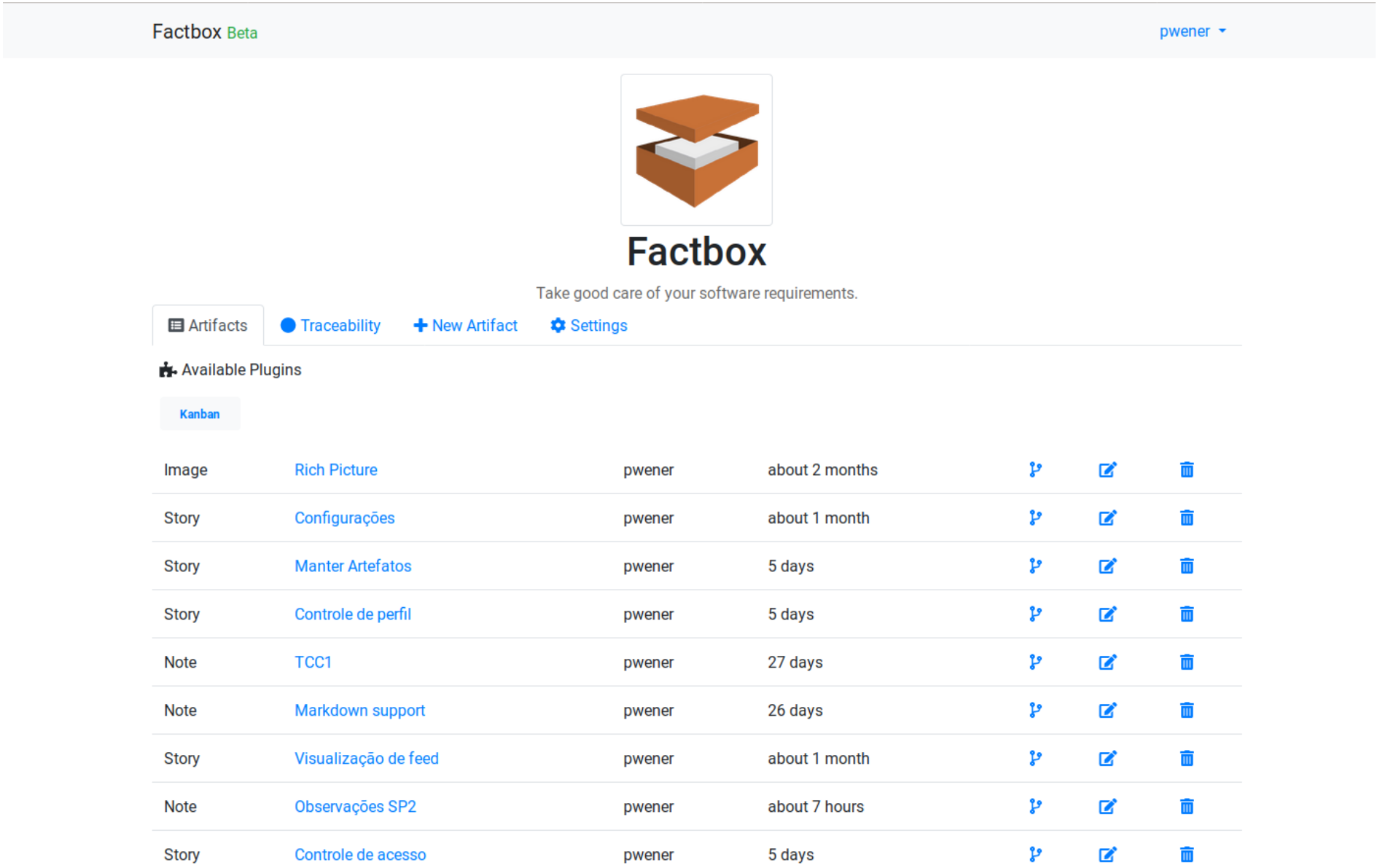

Figura 32 – Modificações de usabilidade realizadas durante a primeira *Sprint* na página de visualização de projeto

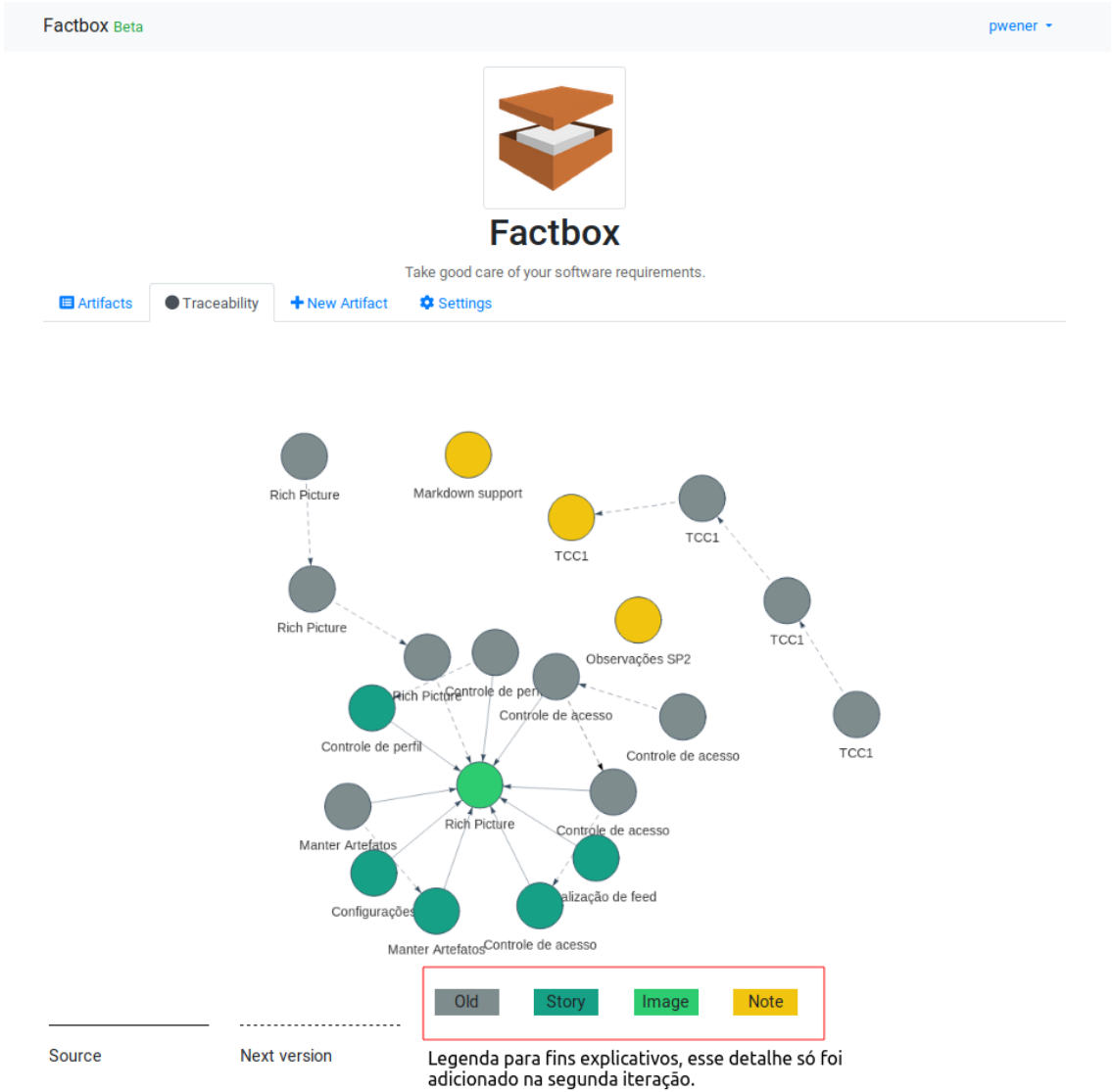

Figura 33 – Modificações de usabilidade realizadas durante a primeira *Sprint* no grafo de Rastreabilidade.

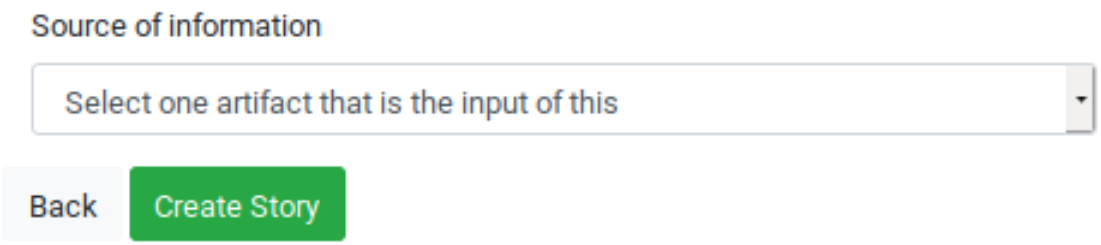

Figura 34 – Modificações de usabilidade realizadas durante a primeira *Sprint* no grafo de Rastreabilidade

6.4 Segundo Ciclo

No segundo ciclo, foi aplicado um questionário com três perguntas disponíveis no Apêndice B. O público da interação passada foi outra vez submetido a responder o formulário. O método de observação participante continuou sendo utilizado, agregando valor à análise qualitativa, reconhecendo pontos de melhoria.

6.4.1 Coleta de dados, Análise e interpretação

Tal como no ciclo passado, os tópicos a seguir são resultados das atividades de Coleta de dados e Análise e interpretação.

6.4.1.1 Questionário

A primeira pergunta, "Você considera que a ferramenta cumpre com a proposta de Rastreabilidade e Versionamento?", permitiu avaliar a qualidade da ferramenta frente ás questões de Rastreabilidade e Versionamento dos artefatos.

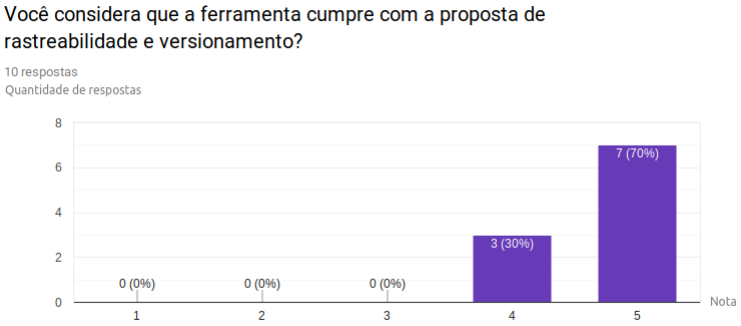

Figura 35 – Primeira pergunta do Questionário do segundo ciclo de pesquisa-ação

A Figura 35 traz o gráfico sobre as avaliações da primeira pergunta. Com alta atribuição dada, entende-se que a ferramenta tem uma impressão positiva frente as questões de Rastreabilidade e Versionamento.

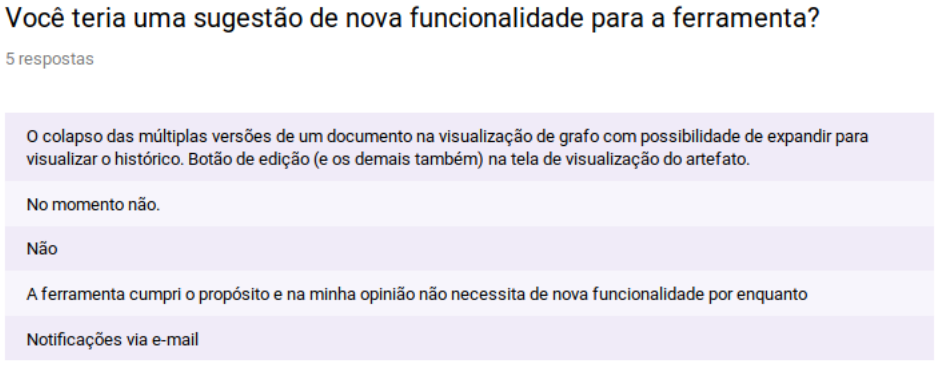

Figura 36 – Segunda pergunta do Questionário do segundo ciclo de pesquisa-ação

A segunda pergunta, "Você teria uma sugestão de nova funcionalidade para a ferramenta?", permitiu coletar sugestões que poderiam compor o *backlog*. Na Figura 36, têm-se as sugestões coletadas, sendo algumas consideradas para compor o *backlog*.

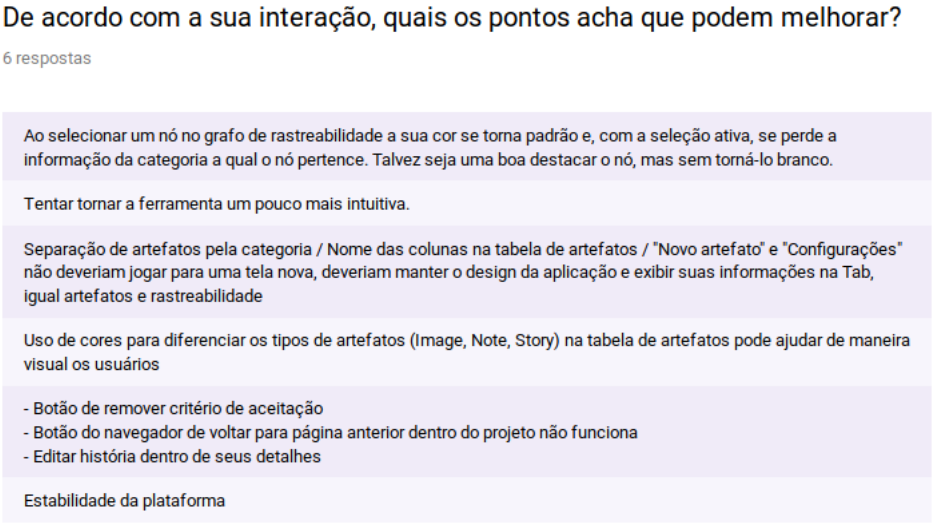

Figura 37 – Terceira pergunta do Questionário do segundo ciclo de pesquisa-ação

A terceira pergunta, "De acordo com a sua interação, quais os pontos acha que podem melhorar?", permitiu detectar pontos de melhoria e correção. Na Figura 37, têm-se os pontos levantados, sendo alguns considerados para compor o *backlog*.

6.4.1.2 Observação Participante

Durante a iteração dos usuários, outra vez foi aplicada a técnica de observação participante, a qual gerou uma série de impressões, muitas das quais captadas de forma mais objetiva nos questionários.

Nessa iteração, os usuários tiveram que explorar mais páginas para cumprir seus objetivos. Além da visualização dos artefatos e criação, que tinham sido as funções mais utilizadas, os usuários iteragiram com mais frequência com o grafo de Rastreabilidade e o recurso de Versionamento. As páginas de edição dos artefatos foram mais utilizada também.

As questões de usabilidade ainda tiveram destaque na segunda iteração. Dentre elas: (i) no gráfico de usabilidade, os usuários requisitaram informações do autor para compreensão, e (ii) durante a navegação, os usuários relataram sentir falta de mecanismos para voltar ou se localizar dentro do sistema.

Outros *bugs* atrapalharam a iteração dos usuários. Diante da criticidade, foram corrigidos antes de finalizar a Coleta de resultados. Esses serão descritos com mais detalhes nas seções adiante.

6.4.2 Plano de Ação

Diante do exposto na seção anterior, foi possível levantar uma série de pontos de melhoria. Dentre as evoluções sugeridas, foram priorizadas:

- Adição de informações e reorganização do *layout* do grafo de Rastreabilidade;
- Adição de um mecanismo eficiente para facilitar a navegação nas páginas, e
- Correção dos *bugs*.

6.4.3 Divulgação dos Resultados

Com a implementação dos itens levantados na seção anterior, tem-se os seguintes resultados:

- Criação de legendas para identificação do tipo de artefato, conforme Figura 38;
- Reorganização da posição das legendas, conforme Figura 38;
- Adição de um *menu*(conhecido por *Breadcrumb*) em todas telas secundárias do projeto, conforme Figura 39;

- Correção de *bug* de sessão: Alguns usuários relataram erro em páginas de acesso restrito. O usuário perdia a sessão e o sistema não tratava isso, redirecionando-o apropriadamente para o *login*;
- Correção de *bug* de Versionamento, no qual a visualização de uma versão levava sempre a última versão do artefato. Foi resolvido antes dos *feedbacks* por se tratar de uma funcionalidade crítica na *sprint*, e
- *Download* ou visualização de mais detalhes da *release*, disponíveis em <https://github.com/factbox/factbox/releases/tag/v.1.2>.

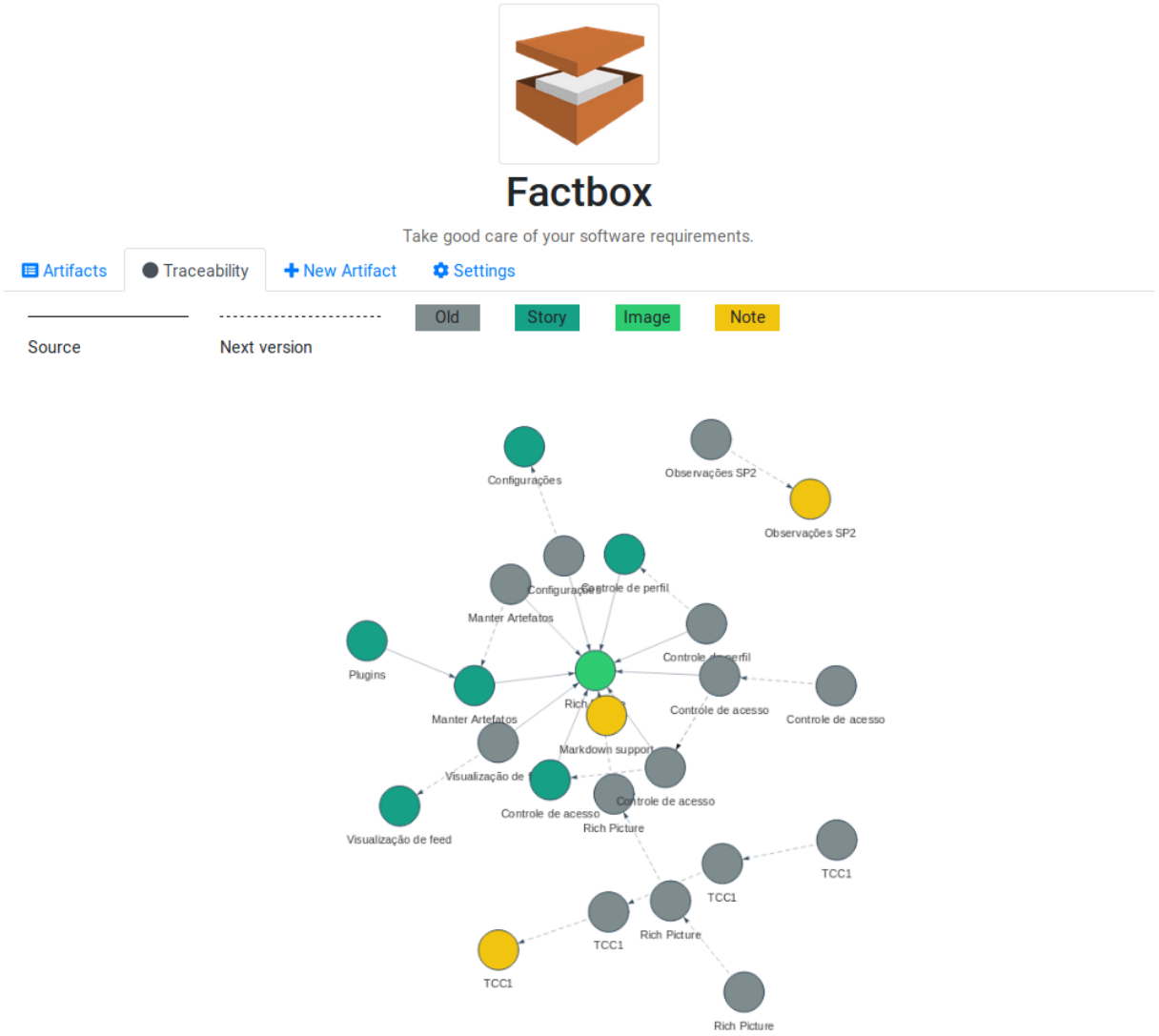

Figura 38 – Modificações de usabilidade realizadas durante a segunda *Sprint* na página de Rastreabilidade

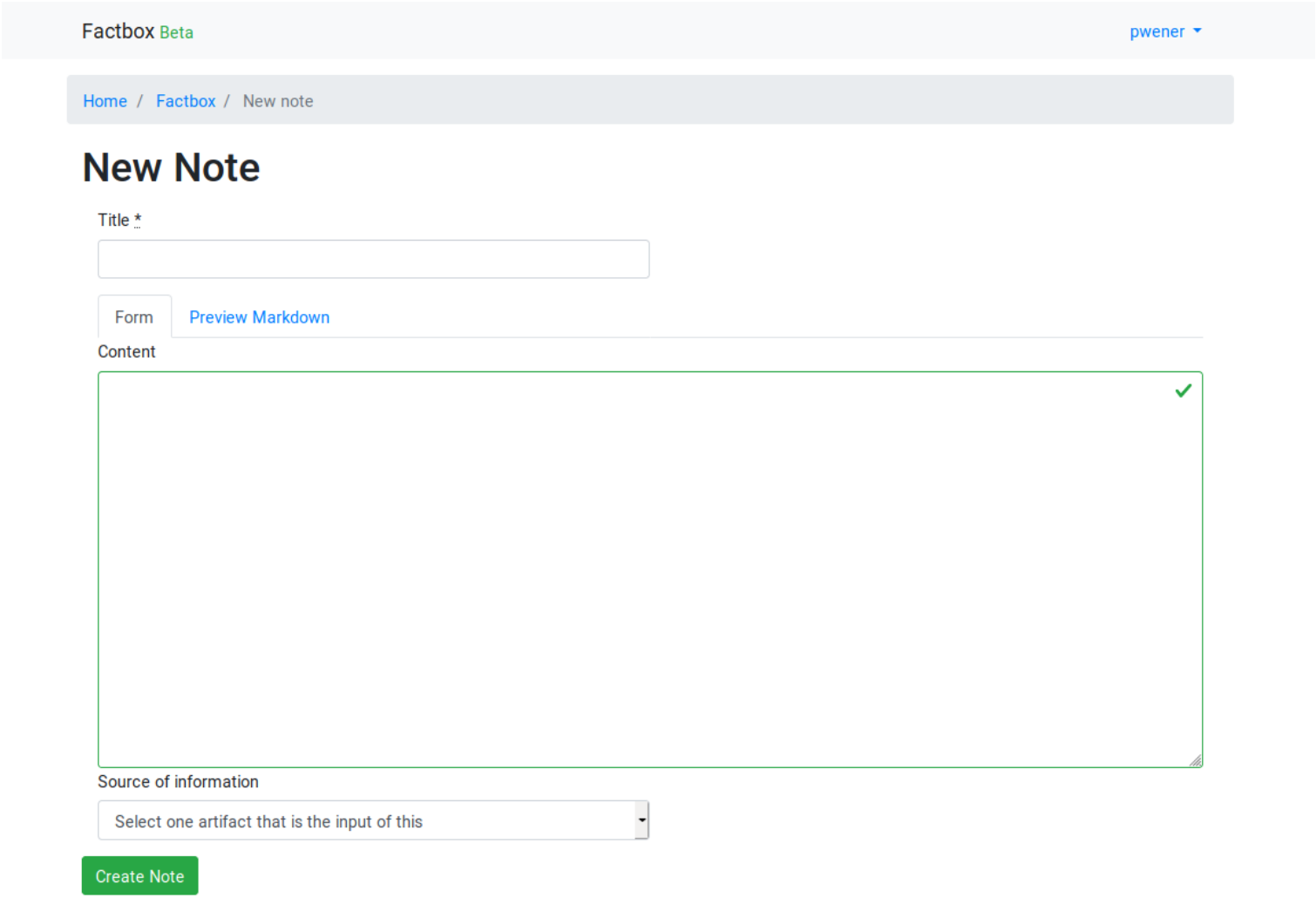

Figura 39 – Modificações de usabilidade realizadas durante a segunda *Sprint*. Instalação de um *Breadcrumb*.

6.5 Terceiro Ciclo

Neste terceiro ciclo, foi aplicado um questionário com quatro perguntas, disponível no Apêndice C. O mesmo público alvo das iterações passadas foi submetido ao formúlário. O método de observação participante continuou sendo utilizado.

6.5.1 Coleta de dados, Análise e interpretação

Os tópicos a seguir são resultado das atividades de Coleta de dados e Análise e interpretação.

6.5.1.1 Questionário

A primeira pergunta, "Você considera que a ferramenta cumpre com a proposta de rastreabilidade e versionamento?", permitiu outra vez, avaliar de um modo geral, a percepção do usuário frente as questões de Rastreabilidade e Versionamento dos artefatos.

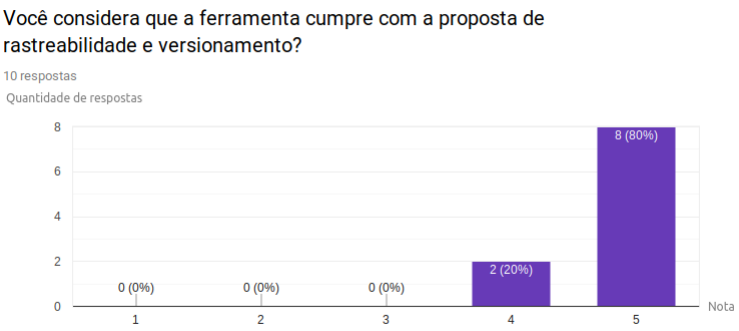

Figura 40 – Primeira pergunta do Questionário do terceiro ciclo de pesquisa-ação

A Figura 40 traz o gráfico sobre as avaliações da primeira pergunta. Com o resultado observado no gráfico, entende-se que os usuários tem uma impressão positiva em relação as questões de Rastreabilidade e Versionamento, as quais a ferramenta Factbox se propõe a resolver. Em relação aos ciclos passados, percebe-se ainda uma evolução na satisfação desse aspecto.

A segunda pergunta, "Você utilizaria a ferramenta em um outro projeto?", permitiu entender se os usuários conseguem ver aplicação da ferramenta em outros contextos. A Figura 41 exibe o resultado dessa questão. A maioria dos votos foi positivo.

A terceira pergunta, "Você teria uma sugestão de nova funcionalidade para a ferramenta?", permitiu coletar sugestões que poderiam compor novos itens no *backlog*. Na Figura 42, têm-se os pontos levantados. Foram avaliados e filtrados para compor novos itens no *backlog*.

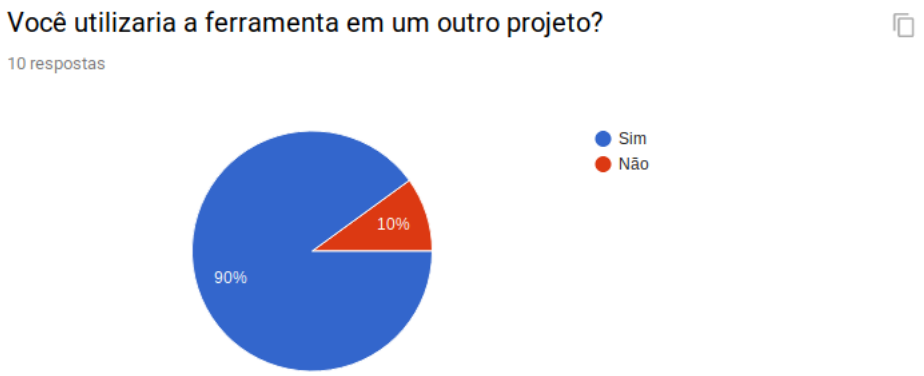

Figura 41 – Segunda pergunta do Questionário do terceiro ciclo de pesquisa-ação

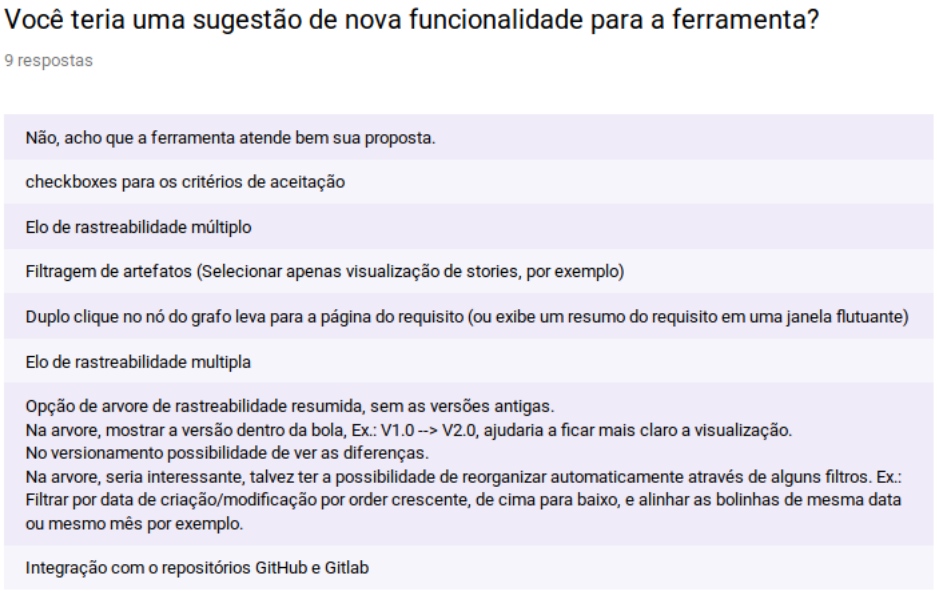

Figura 42 – Terceira pergunta do Questionário do terceiro ciclo de pesquisa-ação

A quarta pergunta, "De acordo com a sua interação, quais os pontos acha que podem melhorar?", permitiu avaliar itens de melhoria dentro do sistema. Na Figura 43, têm-se os pontos levantados, sendo alguns considerados para compor o *backlog*.

6.5.1.2 Observação Participante

Essa última iteração foi a que gerou um maior número de sugestões. Basicamente, o cenário de uso proposto foi o entendimento do escopo de todo projeto e da sua linha histórica. Através desse olhar, várias possibilidades foram identificadas pelos usuários.

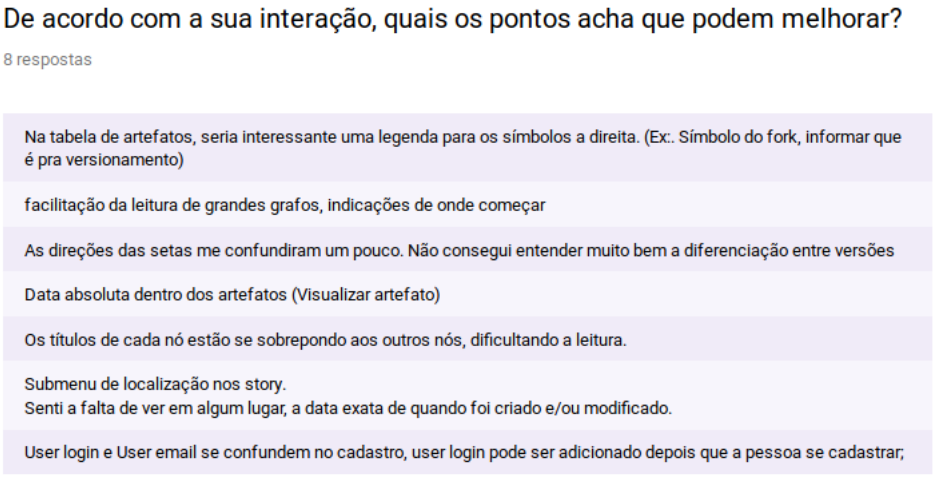

Figura 43 – Quarta pergunta do Questionário do terceiro ciclo de pesquisa-ação

Um dos pontos levantados com maior frequência foi quanto a questão da quantidade de informação no grafo de Rastreabilidade. Algumas soluções foram propostas para resolver o problema, dentre elas, duas foram integradas ao *backlog*, como critério de aceitação do requisito de **Manter Artefatos**(História que lida com o grafo de Rastreabilidade).

Os usuários questionaram outros tópicos de usabilidade, esses foram priorizados e serão descritos com mais detalhes nas seções seguintes.

6.5.2 Plano de ação

De acordo com a seção anterior, foi possível levantar uma série de pontos de melhoria. As priorizadas para a *sprint* de desenvolvimento foram:

- Adição do *breadcrumb* no visualizar e editar das histórias de usuário;
- Visualização de data de criação dentro da página de visualização de cada artefato, e
- Visualização do responsável pela criação ou atualização dentro da página de visualização de cada artefato.

Nessa última *sprint*, apesar de poucos itens, o *backlog* cresceu mais do que nas outras, uma vez que novas Histórias e critérios de aceitação foram criados.

6.5.3 Divulgação dos Resultados

Com a implementação dos itens levantados na seção anterior, têm-se os seguintes resultados:

- Adição do dado da última atualização e o correspondente autor, conforme Figura 44;
- Adição do *Breadcrumb* faltante nas páginas de edição e visualização das histórias de usuário, conforme Figura 44, e
- Correções de outros *bugs* ou mais detalhes da *release*, disponíveis em <https://github.com/factbox/factbox/releases/tag/v.1.0.3>

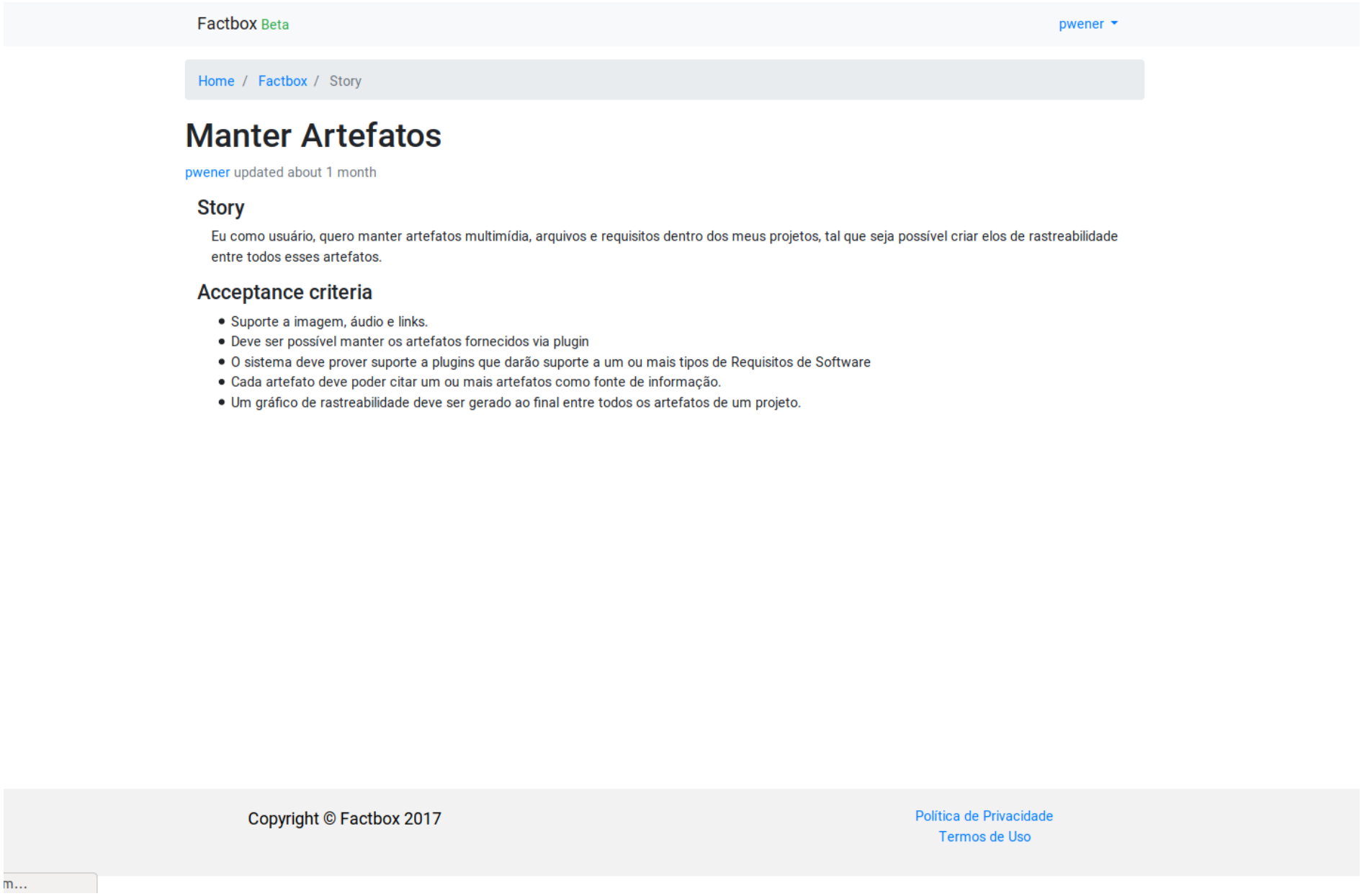

Figura 44 – Tela de História de usuário após modificações do terceiro ciclo.

6.6 Resumo do Capítulo

Neste capítulo, foram apresentados os resultados obtidos ao longo dos ciclos de pesquisa-ação realizados. Ao todo, foram três ciclos de pesquisa-ação. Os resultados encontrados estão vinculados aos objetivos específicos do trabalho. No próximo capítulo, esses objetivos e outros aspectos serão retomados e debatidos com base nos resultados obtidos e apresentados nesse capítulo.

7 Conclusão

Neste capítulo, serão apresentadas as considerações finais sobre esse Trabalho de Conclusão de Curso. Nesse sentido, serão retomados os objetivos específicos acordados no Capítulo 1, evidenciando se os mesmos foram atingidos ou não, com justificativas. Em seguida, uma seção é dedicada a exibir os resultados obtidos em casos de uso da ferramenta. Por fim, têm-se a apresentação das competências, pontos frágeis e trabalhos futuros da ferramenta.

7.1 Objetivos Alcançados

- Investigação dos tópicos de interesse desse trabalho, orientando-se pelas palavras chave: Gerência dos Requisitos, Rastreabilidade (Pré e Pós-rastreabilidade) e Versionamento.

  Tal objetivo foi alcançado ainda na primeira parte desse trabalho.

- Elaboração da solução computacional condizente com as demandas acordadas no processo investigativo. Essa solução confirma-se como um suporte, disponibilizado de forma *online*, que permite rastrear e versionar Requisitos de Software. Além disso, esse suporte auxilia o Engenheiro de Software no que se refere à Gerência dos Requisitos.

  Tal objetivo foi validado nos ciclos de pesquisa-ação, através dos Questionários. Nas questões de pertinência da solução, os resultados indicam que a ferramenta auxiliou na margem do esperado.

- Análise dos resultados obtidos utilizando a metodologia científica de pesquisa-ação. Essa metodologia, conforme pode ser visto no Capítulo de Resultados Obtidos, permitiu realizar ciclos de investigação bem como de ajustes do comportamento da ferramenta, evoluindo-a e adequando-a conforme os *feedbacks* recebidos do público alvo.

  De acordo com as evoluções obtidas, fica evidente que esse objetivo foi alcançado.

7.2 Resultados dos Casos de Uso

Além dos dois projetos utilizados no processo de pesquisa-ação, em paralelo, foi desenvolvido por alunos da disciplina de Requisitos de Software da UnB, um projeto de

Requisitos da ferramenta Habitica: <https://habitica.com>. A Figura 45 ilustra o grafo final alcançado pelo grupo de alunos.

A Figura 46 e Figura 47 apresenta os grafos finais dos dois projetos desenvolvidos nos ciclos de pesquisa-ação.

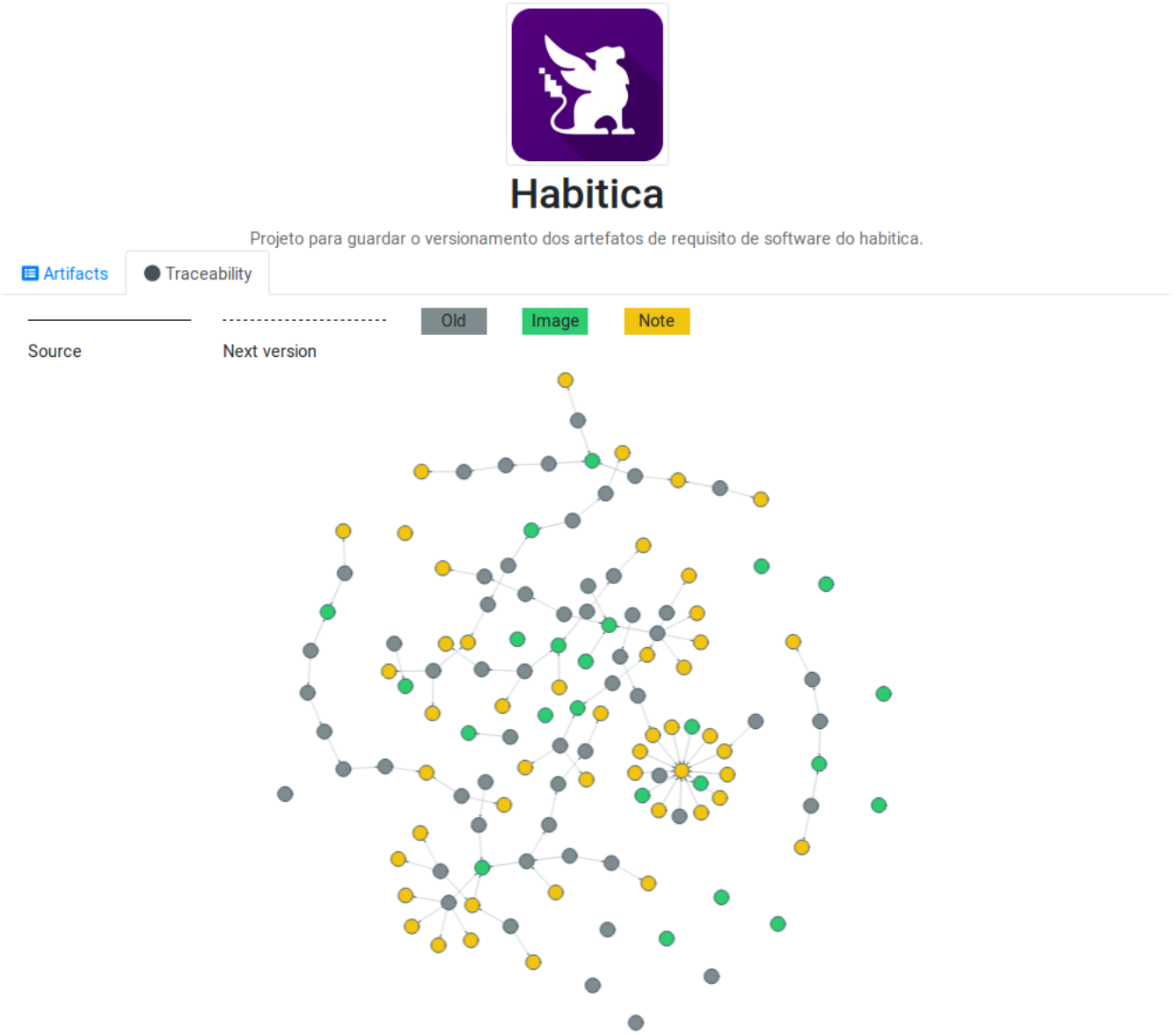

Figura 45 – Gráfico resultante do projeto Habitica

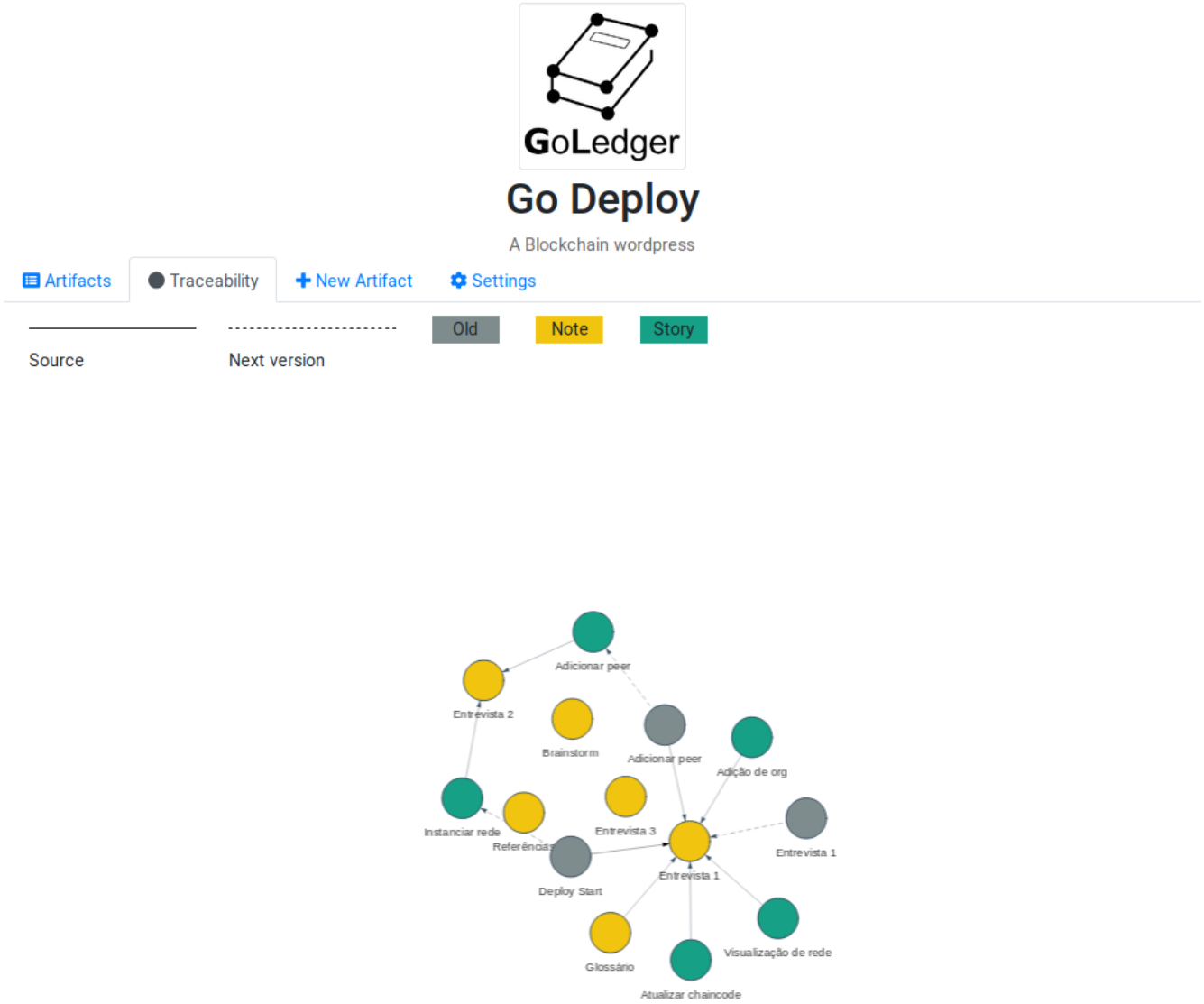

Figura 46 – Gráfico resultante do projeto Go Deploy

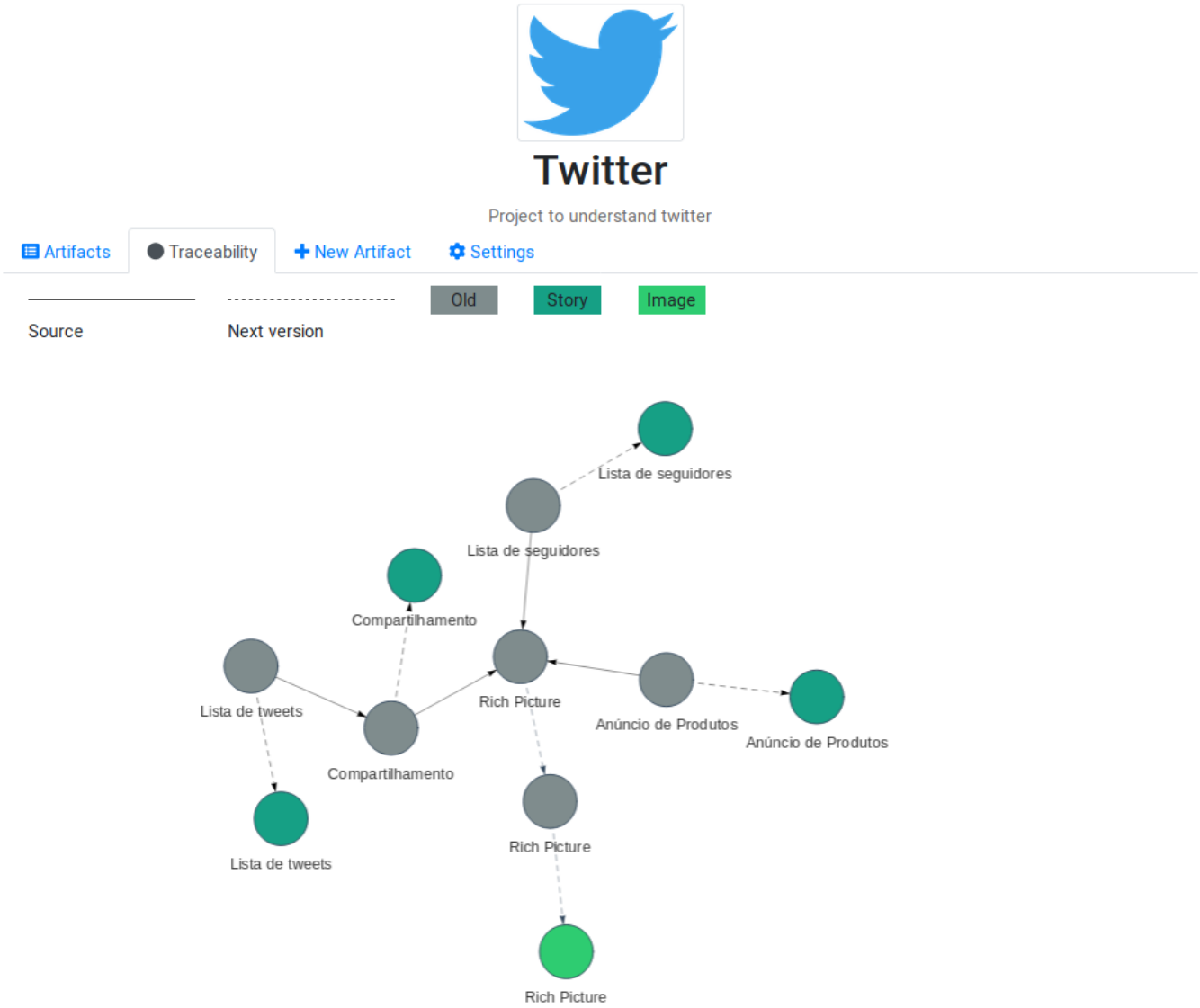

Figura 47 – Gráfico resultante do projeto Twitter

7.3 Competências do Factbox

A ferramenta cumpriu, desde a sua primeira versão, com a missão de Rastreabilidade e Versionamento. Esses dois aspectos foram tratados de forma automática, gerando ao usuário uma compreensão da *baseline* de Requisitos como não se vê em outras ferramentas.

A extensibilidade mostrou-se eficiente, apesar do recurso ter sido pouco explorado. Cabe ressaltar que o *plugin* de *kanban* evoluiu junto da ferramenta, mantendo, durante todas as iterações, sua compatibilidade.

7.4 Fragilidades do Factbox

Apesar de vários resultados positivos, a ferramenta mostrou-se com desafios pela frente. Isso permitirá evoluções em muitos sentidos. Uma das primeiras fragilidades da ferramenta é oferecer suporte a uma quantidade pequena de tipos de artefatos, o que deve ser solucionado com a criação de mais *plugins*. Tal extensibilidade, exige uma comunidade ou grupo maior para manter o projeto com ampla variedade de tipos de requisitos, sendo essa uma situação ideal.

Alguns pontos de usabilidade também foram notados como fragilidades e devem ser repensados como trabalhos futuros. Dentre esses, destaca-se a visualização do grafo, ao qual foram conferidas críticas, em particular, em termos de compreensividade. Novamente, caberiam estudos, com base nesse critério de qualidade, e visando aprimorar as informações reveladas no grafo, as cores utilizadas nos nós, e outros rastros mantidos via grafo. Nesse sentido, seguem algumas iniciativas que agregariam valor à Factbox, e que poderiam ser tratadas em trabalhos futuros.

7.5 Trabalhos futuros

Vários itens levantados nos ciclos de Pesquisa-ação foram adaptados e incluídos nos requisitos da ferramenta, disponíveis em <https://factbox.app/projects/Factbox>.

Podem ser destacados os seguintes trabalhos futuros:

- Integrar com ferramentas de versionamento de código, como *Gitlab* ou *Github*, de modo que *commits* e *Pull Requests* sejam facilmente referenciados dentro do Factbox;
- Aumentar os aspectos de usabilidade do grafo de Rastreabilidade, como facilitar o entendimento dos elos; reduzir a quantidade de nós e criar mecanismos de filtragem;
- Dar suporte a múltiplas referências em termos de fontes de informações, e

- Criar *plugins* alternativos visando conferir suporte a uma variedade maior de tipos de artefatos.

Referências

BECK, K.; FOWLER, M. *User Stories Applied The Addison-Wesley Signature Series The Addison-Wesley Signature Series*. [S.l.: s.n.]. ISSN 0321205685. ISBN 978-0321205681. Citado na página 35.

BERG, T. Rich Picture: The Role of the Facilitator. *Systemic Practice and Action Research*, v. 28, n. 1, p. 67–77, 2015. ISSN 1094429X. Citado 3 vezes nas páginas 15, 33 e 34.

BERG, T.; POOLEY, R. Rich Pictures: Collaborative Communication Through Icons. *Systemic Practice and Action Research*, v. 26, n. 4, p. 361–376, 2013. ISSN 1094429X. Citado na página 34.

BOURQUE, P.; FAIRLEY, R. E. (Ed.). *SWEBOK: Guide to the Software Engineering Body of Knowledge*. Version 3.0. Los Alamitos, CA: IEEE Computer Society, 2014. ISBN 978-0-7695-5166-1. Disponível em: <http://www.swebok.org/>. Citado 2 vezes nas páginas 25 e 30.

BROOKS, F. P. J. No silver bullet-essence and accidents of software engineering. *Proceedings of the IFIP Tenth World Computing Conference*, p. 1069–1076, 1986. ISSN 00189162. Disponível em: <http://www.cgl.ucsf.edu/Outreach/pc204/NoSilverBullet>. Citado na página 33.

BURGDORF, C. 2014. Disponível em: <https://blog.thoughtram.io/git/2014/11/18/the-anatomy-of-a-git-commit.html>. Citado na página 67.

CAETANO, P. R. S. R. Metodologia para Equipas de Desenvolvimento de Requisitos de Sistemas de Informação. p. 152, 2008. Citado 2 vezes nas páginas 25 e 26.

COMPUTING, K. *ActiveRecord::ActsAs*. Berlin: Kraut Computing, 2010. Disponível em: <https://krautcomputing.github.io/active_record-acts_as>. Citado na página 66.

ELSEVIER. 2018. Disponível em: <https://www.elsevier.com/solutions/mendeley>. Citado na página 40.

FIRESMITH, D. Modern requirements specification. *Journal of Object Technology*, v. 2, n. 2, p. 53–64, 2003. ISSN 16601769. Citado 3 vezes nas páginas 29, 33 e 36.

FITZGERALD, B. Software crisis 2.0. *Computer*, v. 45, n. 4, p. 89–91, 2012. ISSN 00189162. Citado na página 30.

GERHARDT, S. Métodos de Pesquisa. p. 31–32, 2009. Citado 2 vezes nas páginas 43 e 45.

GIL, A. *Como elaborar projetos de pesquisa*. Atlas, 2010. ISBN 9788522458233. Disponível em: <https://books.google.com.br/books?id=HSGHRAAACAAJ>. Citado 4 vezes nas páginas 43, 44, 45 e 50.

GIT. 2018. Disponível em: <https://git-scm.com/book/en/v2/Getting-Started-A-Short-History-of-Git>. Citado na página 40.

GOTEL, O. C. Z.; FINKELSTEIN, A. C. W. An Analysis of the Requirements Traceability Problem. *1st International Conference on Requirements Engineering (RE 1994)*, p. 94–101, 1994. ISSN 0425-1644. Citado na página 33.

GUIDE, R. *The Basics of Creating Rails Plugins*. 2018. Disponível em: <https://guides.rubyonrails.org/plugins.html>. Citado na página 64.

GUIDE, R. *Single Table Inheritance*. 2018. Disponível em: <https://guides.rubyonrails.org/association_basics.html#polymorphic-associations>. Citado na página 66.

HIGHSMITH, J.; COCKBURN, A. Agile software development: the business of innovation. *Computer*, v. 34, n. 9, p. 120–127, Sept 2001. ISSN 0018-9162. Citado na página 47.

KUBUNTU. *Termos de Licensa do Kubuntu*. 2017. Disponível em: <https://kubuntu.org/legal/>. Citado na página 39.

LAKHOUA, M. N.; ANNABI, M. Approach of analysis of methods sa, sadt and sart. *2006 2nd International Conference on Information and Communication Technologies*, v. 1, p. 473–477, 2006. Citado na página 60.

LARMAN, C. *Utilizando UML e Padrões*. Bookman, 2007. ISBN 9788560031528. Disponível em: <https://books.google.com.br/books?id=ZHtcynS03DIC>. Citado na página 65.

LATEX. 2018. Disponível em: <https://www.latex-project.org/about>. Citado na página 40.

MARTIN, R. C. *Clean Code: A Handbook of Agile Software Craftsmanship*. 1. ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2008. ISBN 0132350882, 9780132350884. Citado na página 64.

NUSEIBEH, B.; EASTERBROOK, S. Requirements engineering: A roadmap. In: *Proceedings of the Conference on The Future of Software Engineering*. New York, NY, USA: ACM, 2000. (ICSE '00), p. 35–46. ISBN 1-58113-253-0. Disponível em: <http://doi.acm.org/10.1145/336512.336523>. Citado 4 vezes nas páginas 25, 26, 32 e 33.

PARNAS, D. L.; CLEMENTS, P. C. A rational design process: How and why to fake it. *IEEE Trans. Softw. Eng.*, IEEE Press, Piscataway, NJ, USA, v. 12, n. 2, p. 251–257, fev. 1986. ISSN 0098-5589. Disponível em: <http://dl.acm.org/citation.cfm?id=9794.9800>. Citado 2 vezes nas páginas 30 e 31.

PRESSMAN, R. *Software Engineering: A Practitioner's Approach*. 7. ed. New York, NY, USA: McGraw-Hill, Inc., 2010. ISBN 0073375977, 9780073375977. Citado na página 30.

RAMESH, B.; JARKE, M. Toward reference models for requirements traceability. *IEEE Transactions on Software Engineering*, v. 27, n. 1, p. 58–93, 2001. ISSN 00985589. Citado 2 vezes nas páginas 32 e 33.

RONALD, E. *Essential XP: Card, Conversation, Confirmation*. 2001. Disponível em: <https://ronjeffries.com/xprog/articles/expcardconversationconfirmation>. Citado na página 27.

RUBY. *Termos de Licensa do Ruby*. 2017. Disponível em: <https://www.ruby-lang.org/en/about/license.txt>. Citado na página 39.

SAYÃO, M.; CESAR, J.; LEITE, P. Rastreabilidade de Requisitos. *Revista de Informática Teórica e Aplicada*, v. 12, p. 1–30, 2005. ISSN 0103-9741. Citado 3 vezes nas páginas 25, 26 e 33.

SERRANO, M.; LEITE, J. C. S. d. P. A Rich Traceability Model for Social Interactions. *TEFSE 2011 - 6th international workshop on traceability in emerging forms of software engineering*, p. 63–66, 2011. ISSN 02705257. Disponível em: <http://doi.acm.org/10.1145/1987856.1987871>. Citado na página 32.

SILVEIRA, D. T.; CÓRDOVA, F. P. *A pesquisa científica*. [S.l.: s.n.], 2009. 31–42 p. ISSN 1677-5449. ISBN 9788538600718. Citado na página 43.

SILVEIRA, M. C. d. S. P. A reutilização de requisitos no desenvolvimento e adaptação de produtos de software. 2006. Disponível em: <https://repositorio-aberto.up.pt/handle/10216/12020>. Citado 2 vezes nas páginas 25 e 26.

SOMMERVILLE, I. *Software Engineering*. 9th. ed. USA: Addison-Wesley Publishing Company, 2007. ISBN 978-0-321-31379-9, 0-321-31379-8. Citado 3 vezes nas páginas 25, 30 e 32.

TEAM, C. P. *CMMI for Acquisition, Version 1.3*. Pittsburgh, PA, 2010. Disponível em: <http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=9657>. Citado na página 32.

TRIPP, D. Action research : a methodological introduction *. p. 443–466, 2005. Citado 2 vezes nas páginas 43 e 45.

VALLADARES, L. Os dez mandamentos da observação participante. *Revista Brasileira de Ciências Sociais*, v. 22, n. 63, p. 153–155, 2007. ISSN 0102-6909. Citado na página 45.

VIM. 2018. Disponível em: <https://www.vim.org/about.php>. Citado na página 39.

WILLIAMS, L. The xp programmer: the few-minutes programmer. *IEEE Software*, v. 20, n. 3, p. 16–20, May 2003. ISSN 0740-7459. Citado na página 35.

XM1MATH.NET. 2018. Disponível em: <http://www.xm1math.net/texmaker>. Citado na página 40.

Apêndices

APÊNDICE A – Primeiro Questionário

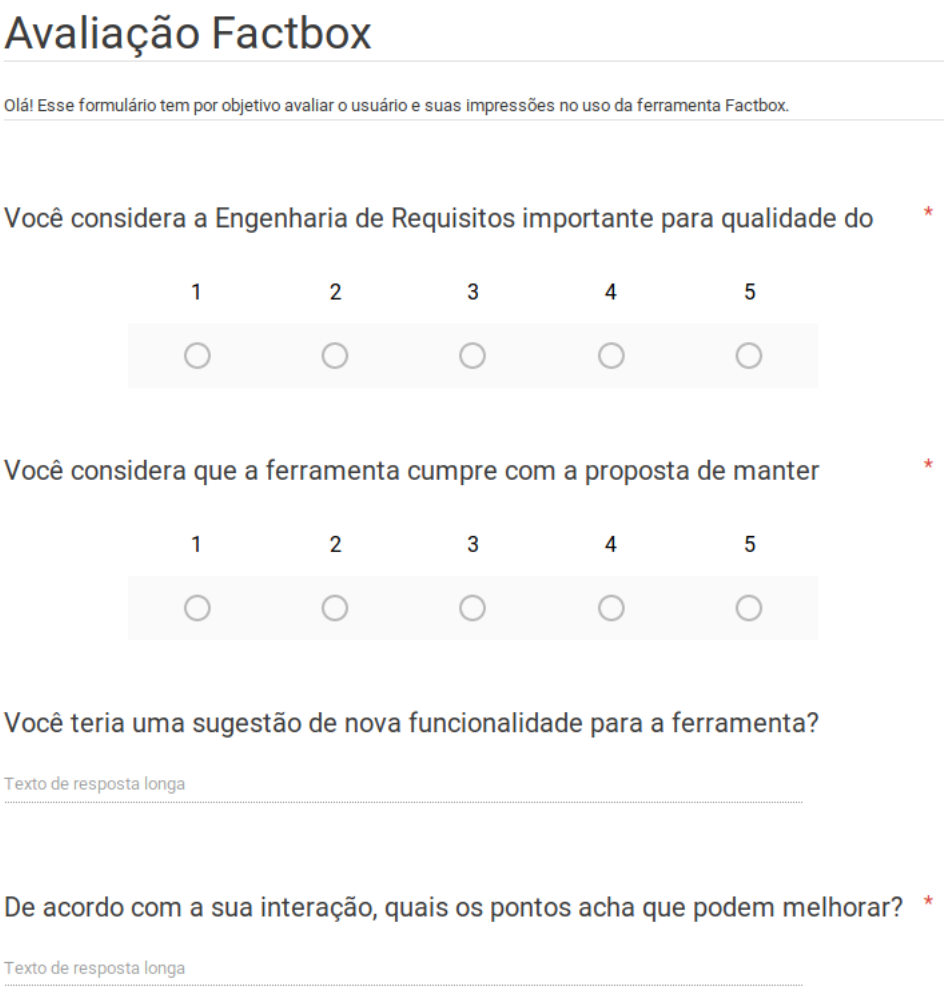

Figura 48 – Questionário utilizado no primeiro ciclo de Pesquisa-ação

APÊNDICE B – Segundo Questionário

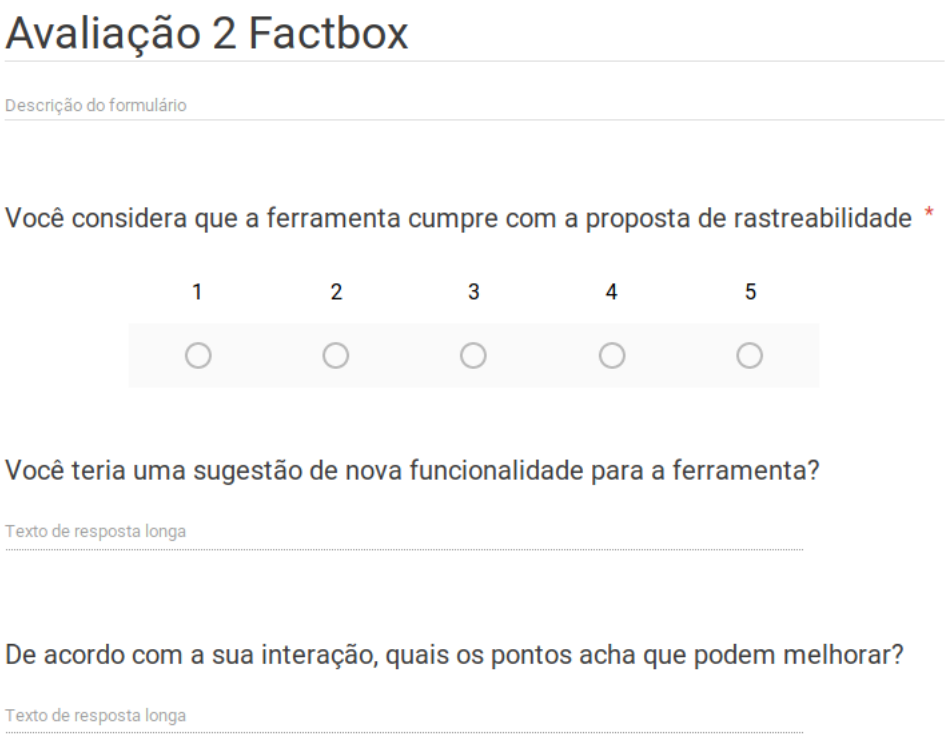

Figura 49 – Questionário utilizado no segundo ciclo de Pesquisa-ação

APÊNDICE C – Terceiro Questionário

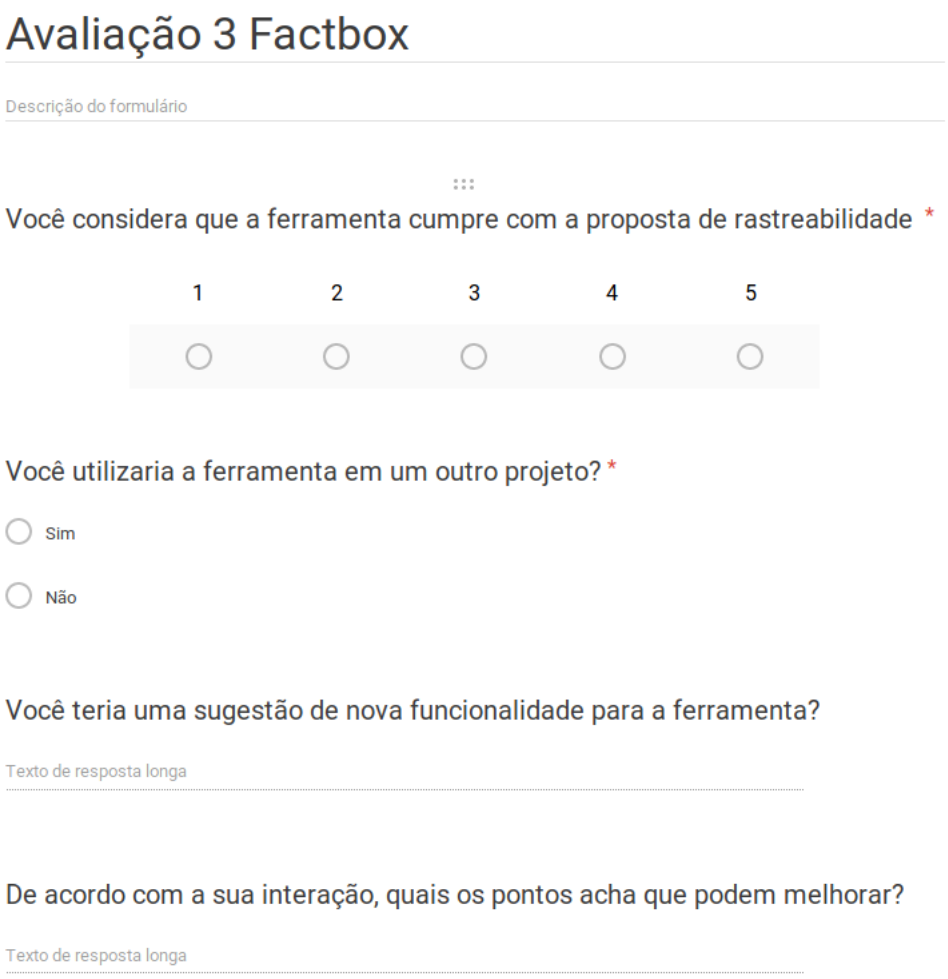

Figura 50 – Questionário utilizado no terceiro ciclo de Pesquisa-ação