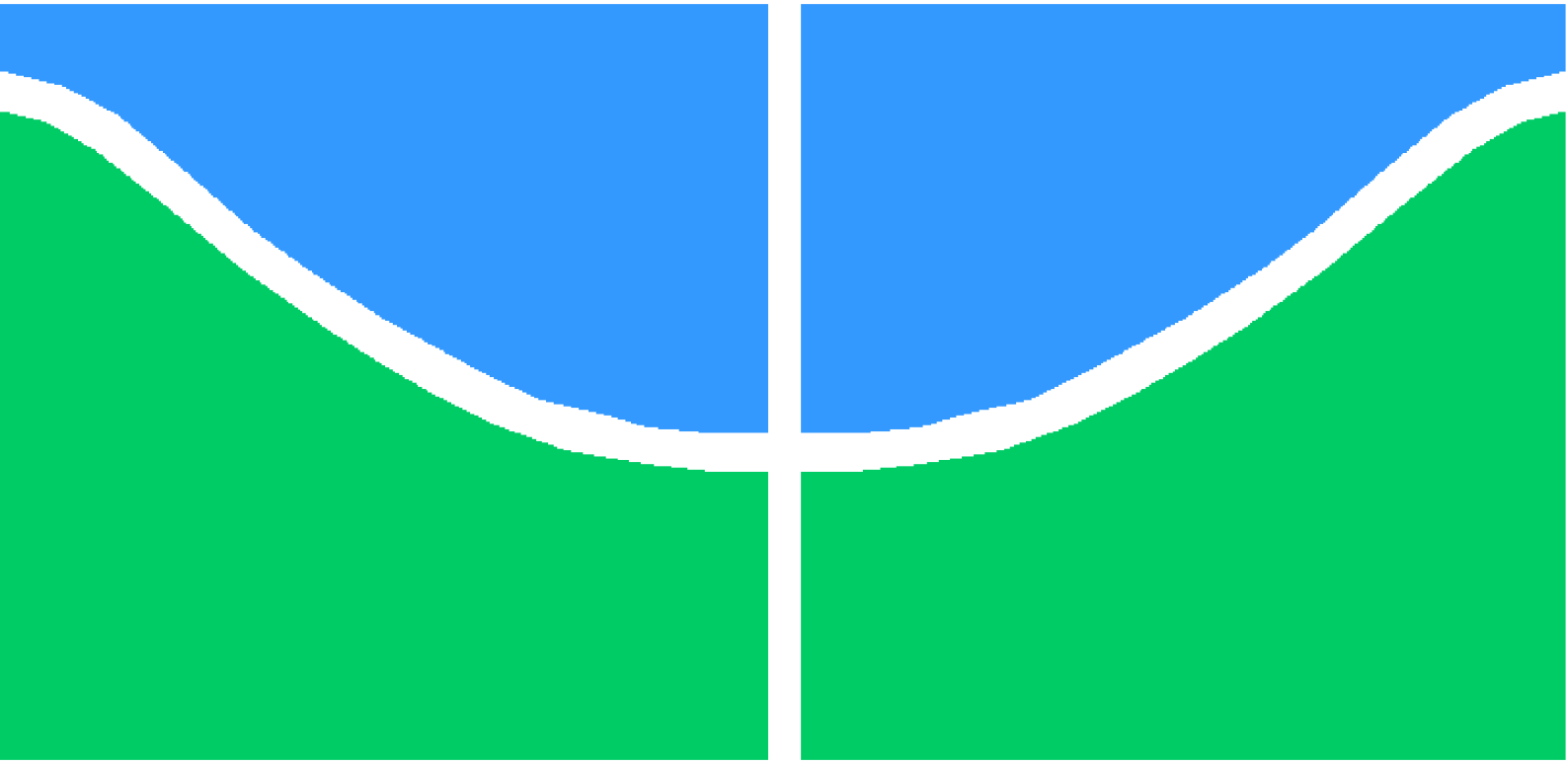

Universidade de Brasília - UnB
Faculdade UnB Gama - FGA
Requisitos de Software

Relatório 1

Autor: Nicácio Arruda, Pedro Sales, Ruan Nawe, Sabryna de Sousa
Orientador: (Prof. Elaine Venson)

Brasília, DF
2016

Nicácio Arruda, Pedro Sales, Ruan Nawe, Sabryna de Sousa

Relatório 1

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: (Prof. Elaine Venson)

Brasília, DF

2016

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

MA	Metodologia Ágil
ER	Engenharia de Requisitos
SAF	Scaled Agile Framework
CMMI	Capability Maturity Model - Integration
MPS.BR	Melhoria do Processo de Software Brasileiro
MR-MPS	Modelo de Referência
MA-MPS	Método de Avaliação
MN- MPS	Modelo de Negócio
GRE	Gerência de Requisitos
DRE	Desenvolvimento de Requisitos
TI	Tecnologia de Informação

Sumário

1 Introdução

"Entender os requisitos de um problema está entre as tarefas mais difíceis enfrentadas por um engenheiro de software".(PRESSMAN, 2006, p.126).

Tendo em vista a afirmação acima, infere-se que é necessário que a engenharia de requisitos forneça mecanismos apropriados para entender as necessidades dos clientes, analisar viabilidade, especificar a solução sem ambiguidades, validar, gerenciar as necessidades conforme suas alterações.

Com base nas observações acima, este relatório tem como objetivo descrever de forma detalhada todos os mecanismos utilizados para levantar, especificar e detalhar todos os requisitos de um software, bem como os mecanismos utilizados nas fases de especificar solução, modelar processo, validar e gerenciar os requisitos.

1.1 Visão Geral do Relatório

Este relatório abrange os seguintes tópicos:

- Contextualização do Cliente: Conhecer a empresa, entender o problema e Descreve o contexto de negócio.
- Abordagem de engenharia de requisitos e Justificativa: Esclarecimento da abordagem escolhida e os motivos que levaram a equipe a escolher a metodologia.
- Processo de engenharia de requisitos: Descreve o processo que será realizado para o levantamento e documentação dos requisitos.
- Técnicas de elicitação de requisitos: Descreve as técnicas de elicitação de requisitos que serão utilizadas no trabalho 2.
- Gerenciamento de requisitos: Descreve os processos que serão realizados para o gerenciamento de requisitos e as ferramentas utilizada para esse objetivo.
- Planejamento do Projeto: Descreve o planejamento das atividades e as ferramentas

1.2 Contexto do Cliente

"Em todos os sistemas novos, o processo de engenharia de requisitos deve começar com um estudo de viabilidade. A entrada para o estudo de viabilidade consiste num

conjunto preliminar de requisitos de negócio, um esboço da descrição do sistema e como o processo pretende apoiar os processos de negócio" (SUMMERVILLE, 2007, p.97).

A empresa Eletrojun possui um sistema de compartilhamento de projetos como ilustrado no apendice " ", esse sistema tem objetivo de reunir diversos estudantes no intuito de compartilhar projetos e ideias.

O software desenvolvido pela Eletrojun está incompleto, há diversas funcionalidades que não estão implementadas, além de erros que devem ser corrigidos. Esse sistema tem um papel muito importante para a empresa, que é o papel de unir todos os estudantes da Universidade de Brasília - Campus FGA, no intuito de desenvolver, criar, melhorar e divulgar projetos. A empresa espera que esse software possa ser o canal de comunicação entre os estudantes, de modo que, projetos possam ser desenvolvidos com o auxílio desse sistema

Esse site tem funcionalidades para gestão de outros projetos e necessita de uma ferramenta para compartilhamento de arquivos, esse sistema deve permitir que o usuário compartilhe ideias com outros usuários.

Atualmente a Eletrojun não possui uma ferramenta para o gerenciamento de ideias, e sofre com diversos problemas administrativos, no que tange ao compartilhamento dessas ideias.

A estrutura da Eletrojun é vertical, e segue uma hierarquia bem definida, esse tipo de estrutura pode ser favorável ao bom andamento do projeto. Segundo Summerville, inevitavelmente os stakeholders irão ter visões diferentes sobre a importância e prioridade dos requisitos, e algumas vezes essas visões entraram em conflito.

Visto que, a equipe de Analista de Requisitos terá que se reunir apenas com a diretora da empresa elimina-se então os problemas das diferentes visões e opiniões por parte dos clientes.

2 Abordagem de ER

A necessidade da busca de métodos que poderiam vir a ajudar na evolução do processo de construção de software, culminaram com o desenvolvimento da Engenharia de Software, disciplina que traz consigo métodos e técnicas que ajudam na administração da complexidade inerente do software. Contudo, apesar de inegáveis avanços, ainda existem pontos obscuros, sendo certamente requisitos de software um destes pontos.

No ano de 1979, o Gouvernment Accouting Office (GAO) nos USA publicou um relatório no qual afirmava que menos de 2% (dois por cento) dos investimentos feitos no desenvolvimento de software resultaram em um software que atendia seus requisitos (2).

Tal ineficiência no alcance da satisfação dos clientes referentes ao produto final de software, resultaram no surgimento, desenvolvimento e definição de processos de desenvolvimento de software, com o objetivo de aumentar a produtividade e diminuir os riscos de um projeto(3).

Dentre estes processos, encontram-se o processo cascata, o interativo, o espiral, o incremental, entre outros, que podem ser aplicados à soluções que melhor se encaixam em seus princípios e contextos. Para o presente projeto, foi proposta uma solução baseada nas Metodologias Ágeis (MAs): Abordagem que tem como objetivo principal garantir a entrega de produtos em um menor prazo, com maior qualidade e satisfazendo às necessidades dos clientes através da aplicação da produção enxuta para desenvolvimento de software, atacando os desafios emergentes de contextos dinâmicos (4). Baseado na análise do processo, foi determinada a utilização do processo SAF para o presente projeto, que será sucintamente descrito subsequentemente.

2.1 Scaled Agile Framework (SAF)

Tendo sua origem nas metodologias e filosofias ágeis, o SAF é um processo criado inicialmente para se adaptar ao contexto empresarial, trazendo a possibilidade de realizar entregas rápidas e com proximidade do cliente, alinhando constantemente com a expectativa do mesmo e adaptando constantemente o projeto às mudanças de requisitos do cliente.

O SAF tem foco no produto em detrimento da documentação e nas pessoas envolvidas no processo, aproximando estas e trazendo o cliente como membro efetivo da equipe. Este tem papel fundamental na tomada de decisões no processo do desenvolvimento do projeto.

Tais práticas corroboram uma maior adaptatividade à mudanças, tendendo a ame-

nizar riscos no processo de execução do projeto, principalmente pela proximidade entre equipe e cliente (BOEHM, 2002) (1)

O SAF é baseado nos princípios Ágeis, que se baseiam nos nove princípios seguintes:

- Adotar uma visão sistêmica
- Entendemos a economia da cadeia de valor
- Desenvolvemos sistemas iterativamente e incrementalmente
- Integramos e testamos frequentemente; adaptamos imediatamente
- Gerenciamos riscos e a eficácia através de ciclos de aprendizado rápido e síncrono
- Facilitamos o fluxo: limitamos o trabalho em execução, reduzimos o tamanho do lote e gerenciamos a duração da fila
- Sincronizamos entre os domínios o planejamento e a colaboração
- Baseamos marcos na avaliação objetiva dos sistemas de trabalho
- Desbloqueamos a motivação intrínseca dos trabalhadores do conhecimento
- Descentralizamos a tomada de decisão

A natureza particular dessa proposta é a ênfase que o framework dá ao gerenciamento de requisitos, trazendo novos conceitos nesta área, como histórias (user stories), temas (themes) e épicos (epics). Conceitos que ampliaram a forma de capturar requisitos, melhor adequados à ênfase em colaboração, flexibilidade e iterações mais curtas no projeto. (5)

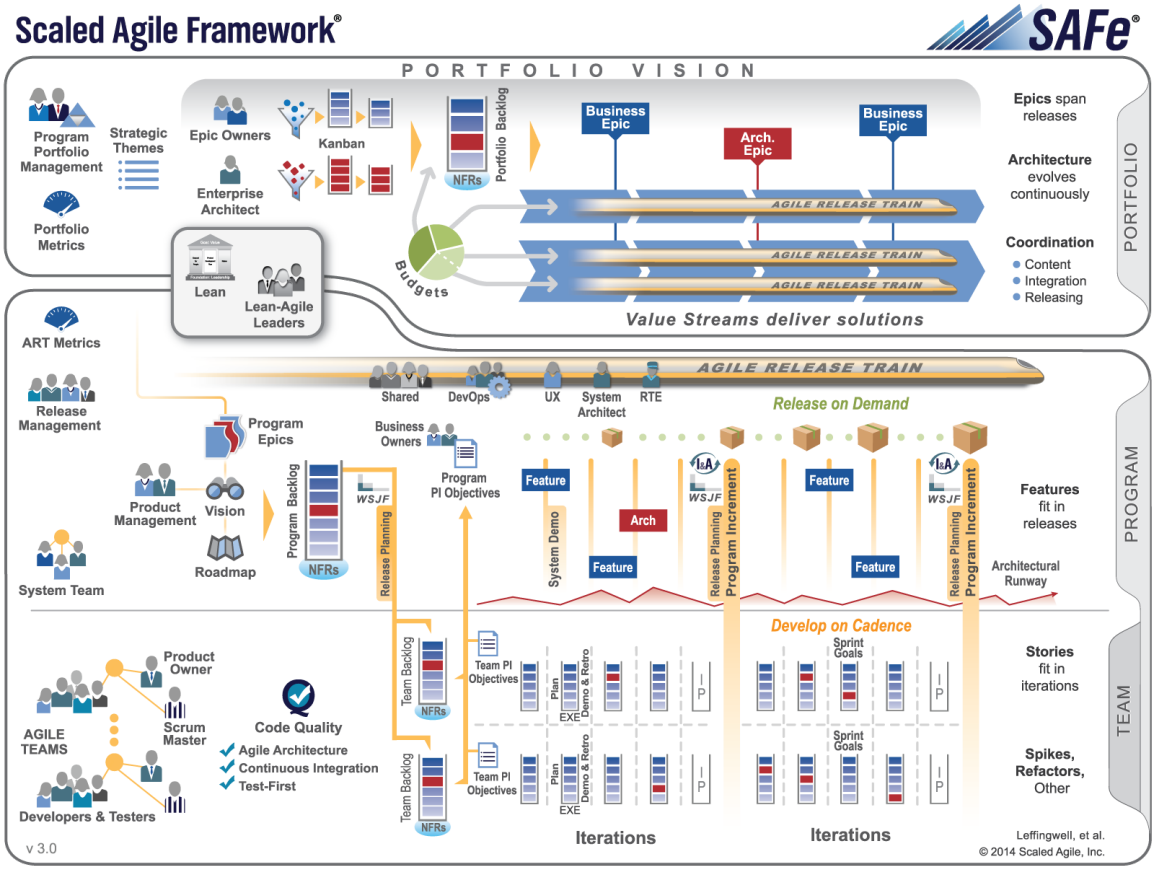

Figura 1 – SAF

2.2 Modelo de Maturidade

O modelo de maturidade foi desenvolvido para auxiliar a execução de atividades que envolvam processos. Ele funciona como um guia em que a organização deve-se espelhar, realizar um plano , a fim de chegar em um nível de excelência. A exigência da contínua melhora dos projetos é o que torna como necessidade este gerenciamento de maturidade.(6)

Pode-se entender que ao utilizar o modelo de maturidade, a organização envolvida terá vários benefícios como: mais produtividade das equipes, redução de custos, riscos do projeto minimizados, aumento do retorno de investimento e qualidade no produto entregue. (7)

2.2.1 MPS.BR

O MPS.BR (Melhoria do Processo de Software Brasileiro) tem o objetivo de melhorar a qualidade dos processos de software, principalmente das micro, pequenas e médias empresas. Ele baseia-se no CMMI (Capability Maturity Model - Integration) e é constituído pelas normas: NBR ISO/IEC 12207 – Processo de Ciclo de Vida de Software, pelas emendas 1 e 2 da norma internacional ISO/IEC 12207 e pela ISO/IEC 15504 – Avaliação

de Processo. (10)

Ele está dividido em três modelos: Modelo de Referência (MR-MPS), Método de Avaliação (MA-MPS), Modelo de Negócio (MN-MPS) como pode ser visto na (Figura 2.2.1).

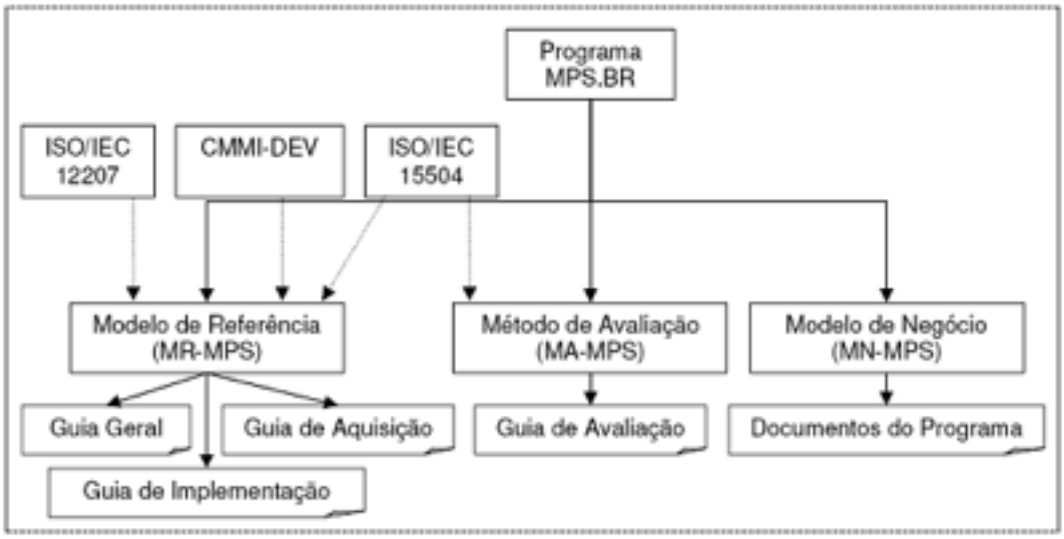

Figura 2 – Programa MPS.BR

- MR-MPS

  O modelo de referência para melhoria do processo de software é dividido em vários níveis de maturidade como pode ser visto na (Figura 2.2.1).

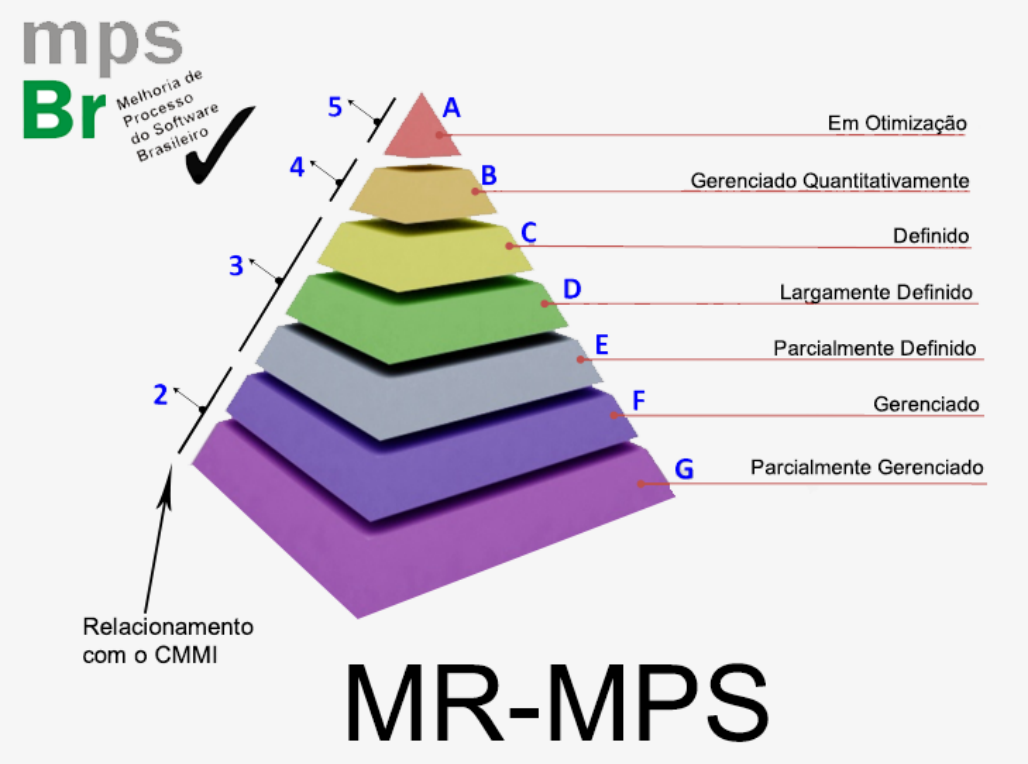

Figura 3 – Níveis de maturidade MR-MPS

"Os níveis de maturidade estabelecem patamares de evolução de processos, caracterizando estágios de melhoria da implementação de processos na organização. O nível de maturidade em que se encontra uma organização permite prever o seu desempenho futuro ao executar um ou mais processos." (10)

Em cada nível de maturidade da figura 2.2.1 são analisados os processos fundamentais, processos organizacionais e processos de apoio.

Processos Fundamentais:

- Aquisição
- Gerência de requisitos
- Desenvolvimento de requisitos
- Solução técnica
- Integração do produto
- Instalação do produto
- Liberação do produto

Processos Organizacionais:

- Gerência de projeto

- Adaptação do processo para gerência de projeto
- Análise de decisão e resolução
- Gerência de riscos
- Avaliação e melhoria do processo organizacional
- Definição do processo organizacional
- Desempenho do processo organizacional
- Gerência quantitativa do projeto
- Análise e resolução de causas
- Inovação e implantação na organização

Processos de apoio:

- Garantia de qualidade
- Gerência de configuração
- Validação
- Medição
- Verificação
- Treinamento

A engenharia de requisitos se encontra nos níveis D e G, Largamente Definido e Parcialmente Gerenciado, respectivamente.

- **Nível G**

  Gerência do Projeto e Gerência de Requisitos compõem o nível G de maturidade. A finalidade do processo de Gerência de Requisitos é gerenciar os requisitos e componentes do produto do projeto. Além de identificar as inconsistências entre os requisitos, planos e produtos do projeto.(12)

  Resultados esperados:

  GRE 1 - Comunicação constante com os fornecedores de requisitos;

  GRE 2 - Os requisitos são compreendidos;

  GRE 3 - Requisitos são aceitos;

  GRE 4 - Comprometimento com os requisitos;

  GRE 5 -Rastreabilidade entre os requisitos, planos do projeto e produtos do trabalho;

  GRE 6 - São corrigidas as inconsistências entre os requisitos, planos e produtos do projeto;

  GRE 7 - Ao longo do projeto as mudanças nos requisitos são gerenciadas.

– **Nível D**

Neste nível acontece o Desenvolvimento de Requisitos, Integração do Produto, Solução Técnica, Validação, e Verificação. (12)

Os requisitos são estabelecidos em conformidade com o cliente.

Resultados esperados:

DRE 1 - São identificadas as necessidades, expectativas, restrições e requisitos de interface do cliente;

DRE 2 - Requisitos funcionais e não-funcionais são estabelecidos;

DRE 3 - Requisitos são refinados;

DRE 4 - Conceitos operacionais e cenários são desenvolvidos;

DRE 5 - As funcionalidades são desenvolvidas;

DRE 6 -Requisitos são avaliados para assegurar as necessidades dos interessados;

DRE 7 - Requisitos são validados.

- MA-MPS

  É o método que permite a avaliação segundo o modelo MPS, seguindo o processo de avaliação e método de avaliação MPS, e características de qualificação dos avaliadores. (10)

  O processo de avaliação é constituído por 4 etapas :

  – Contratar a avaliação;
  – Preparar a realização da avaliação;
  – Realizar a avaliação;
  – Documentar os resultados da avaliação.

- MN-MPS

  O Modelo de Negócio tem o objetivo de credenciar as instituições que seguirem o Modelo de Referência (MR-MPS), Método de Avaliação (MA-MPS), tiverem experiência em processos de software e possuírem estratégias de implementação do modelo e treinamento dos consultores. (10)

2.3 Mapeamento do Contexto vs Abordagens

De acordo com a descrição e análise das abordagens, e as características da equipe, e do projeto, pode-se descrever alguns pontos chaves que influenciam na decisão.

1. A empresa está dispondo de um gerente para auxiliar a equipe de desenvolvimento em todas as tarefas de levantamento de requisitos.

2. A equipe de engenharia de requisitos é integrada e comunicativa, características que facilitam a comunicação com o cliente.

3. A formação da equipe está disposta em 4 membros, dos quais 3 membros já trabalharam juntos utilizando metodologias ágeis, e todos os membros possuem conhecimentos básicos sobre métodos ágeis.

4. A distância entre o cliente e a equipe é pequena, pois as dependências do cliente encontram-se no mesmo local onde a equipe realizará o trabalho.

3 Justificativa

"Aviso! Não cometa o erro de assumir que a agilidade lhe dará licença para abreviar soluções. Processo é um requisito e disciplina é essencial" (Pressman)

Segundo Pressman, para projetos que adotam uma filosofia de desenvolvimento de software ágil, um ambiente ideal seria aquele em que engenheiros de software trabalham juntos na mesma equipe.

Visto que a abordagem ágil requer uma proximidade com o cliente, e a cliente tem uma disponibilidade de tempo para que possamos fazer reuniões semanais, o grupo chegou a conclusão que a abordagem ágil se adéqua melhor ao nosso contexto.

Além da proximidade com a cliente, devido a mesma ser uma estudante da mesma faculdade dos componentes do grupo, em nossa primeira reunião, ela demonstrou possuir conhecimentos da área de software, que foi um dos fatores determinantes para nossa escolha, deixando os componentes mais seguros quanto a metodologia adotada.

Outro fator importante analisado, foi que ela já orientou outro aluno da faculdade que era da Eletrojun para fazer uma parte do sistema, demonstrando uma certa experiência na descrição de requisitos, outro fator analisado e que contou como ponto positivo para a definição da abordagem.

3.1 Panorama Ágil

O manifesto foi feito em 2001 por 17 desenvolvedores que foram na contramão da produção de software tradicional. O manifesto passava a valorizar os indivíduos e interações mais que os processos e ferramentas, o software funcionando mais que a documentação abrangente, colaboração do cliente mais que negociação de contrato, e responder a mudanças mais que seguir um determinado plano.

O manifesto ágil surge com o intuito de sanar as fraquezas reais e perceptíveis da maneira tradicional de produção de software. Apesar da metodologia fornecer benefícios importantes, não é indicada em todas as situações, porque "There is no silver bullet" (Fred Brooks) não existe a bala de prata para a metodologia de desenvolvimento empregada no projeto, tudo vai ter que ser analisado para que se chegue a metodologia mais sensata ao projeto.

3.2 Alterações e mudanças no ágil

Atualmente na economia é difícil prever como uma aplicação irá evoluir em acordo com o tempo devido as condições de mercado. O mercado muda com uma frequência muito difícil de prever, regras de negócio e as necessidades dos usuários finais são as áreas mais afetadas, além disso novas ameaças competitivas surgem inesperadamente no mercado. Em muitas dessas situações não será possível definir completamente os requisitos antes de iniciar o projeto para adaptar-se a competitividade do mercado.

As mudanças no projeto quando não planejadas implicam em custos elevados. Uma das grandes vantagens do ágil é sua adaptabilidade a mudança, reduzindo os custos ao longo de todo o processo de software.

IMAGEMMMMMMMMMMMMMMMMMMMMMMMMM

3.3 Desenvolvimento Ágil x RUP

Hoje em dia ainda existe o mito de que a metodologia ágil é a bala de prata para o processo de desenvolvimento, e que o método tradicional acaba por atrasar, enrolar, ou gerar um trabalho inútil com uma documentação que não serve para nada. O que é comprovado atualmente é que independente da abordagem, ou se são usadas as práticas do PMBOK. Atualmente grande partes das empresas andam adotando modelos híbridos, adaptando principalmente metodologias tradicionais ao contexto do projeto. Uma das grades reclamações sobre o IRUP – IBM Rational Unified process, da IBM, é a grande quantidade de artefatos gerados, sendo que a única coisa que é preconizada no IRUP são as 4 fases do processo, iniciação, construção, elaboração e transição. Uma das primeiras frases do livro IRUP fala que o processo deve ser customizado em acordo com as necessidades das organizações(26).

“The Rational Unified Process is a configurable process. No single process is suitable for all software development.”

Resumidamente é dito que o processo é reconfigurável, que não existe um único processo que seja adequado para todo desenvolvimento de software.

A metodologia ágil ainda é interpretada erroneamente como uma metodologia que tudo de pode, exemplificando melhor, podemos desenvolver sem documentação e sem nenhum padrão de projeto. Essa interpretação é totalmente errônea, tendo em vista que grades empresas, tanto do setor de produção software quanto de outros setores industriais usam ágil e mantêm um nível de organização de projeto e documentação excelente.

Pressman cita em seus livros que uma das prioridades do ágil é a entrega do produto, mas em meio a tantos dizeres sobre ágil, Ivar Jacobson diz que ser ágil virou

moda, e explica o que é ágil na sua concepção:

"Uma equipe ágil é aquela rápida e capaz de responder apropriadamente a mudanças. Mudanças têm muito a ver com desenvolvimento de software. Mudanças no software que está sendo criado, mudanças nos membros da equipe, mudanças devido a novas tecnologias, mudanças de todos os tipos que poderão ter um impacto no produto que está em construção ou no projeto que cria o produto."

Já é mais que comprovado que o ágil é a metodologia de desenvolvimento de software que melhor se adapta a mudanças. O suporte para mudanças no ágil é incorporado em tudo que é desenvolvido, tratado pelo próprio Ivar Jacobson como o coração e a alma do software.

Hoje em dia ainda existe o mito de que a metodologia ágil é a bala de prata para o processo de desenvolvimento, e que o método tradicional acaba por atrasar, enrolar, ou gerar um trabalho inútil com uma documentação que não serve para nada. O que é comprovado atualmente é que independente da abordagem, ou se são usadas as práticas do PMBOK.

Atualmente grande partes das empresas andam adotando modelos híbridos, adaptando principalmente metodologias tradicionais ao contexto do projeto. Uma das grades reclamações sobre o IRUP – IBM Rational Unified process, da IBM, é a grande quantidade de artefatos gerados, sendo que a única coisa que é preconizada no IRUP são as 4 fases do processo, iniciação, construção, elaboração e transição. Uma das primeiras frases do livro IRUP fala que o processo deve ser customizado em acordo com as necessidades das organizações(26).

"The Rational Unified Process is a configurable process. No single process is suitable for all software development."

Resumidamente é dito que o processo é reconfigurável, que não existe um único processo que seja adequado para todo desenvolvimento de software.

A metodologia ágil ainda é interpretada erroneamente como uma metodologia que tudo de pode, exemplificando melhor, podemos desenvolver sem documentação e sem nenhum padrão de projeto. Essa interpretação é totalmente errônea, tendo em vista que grades empresas, tanto do setor de produção software quanto de outros setores industriais usam ágil e mantêm um nível de organização de projeto e documentação excelente.

Pressman cita em seus livros que uma das prioridades do ágil é a entrega do produto, mas em meio a tantos dizeres sobre ágil, Ivar Jacobson diz que ser ágil virou moda, e explica o que é ágil na sua concepção:

"Uma equipe ágil é aquela rápida e capaz de responder apropriadamente a mudanças. Mudanças têm muito a ver com desenvolvimento de software. Mudanças no software

que está sendo criado, mudanças nos membros da equipe, mudanças devido a novas tecnologias, mudanças de todos os tipos que poderão ter um impacto no produto que está em construção ou no projeto que cria o produto."

Já é mais que comprovado que o ágil é a metodologia de desenvolvimento de software que melhor se adapta a mudanças. O suporte para mudanças no ágil é incorporado em tudo que é desenvolvido, tratado pelo próprio Ivar Jacobson como o coração e a alma do software.

4 Processo de Engenharia de Requisitos

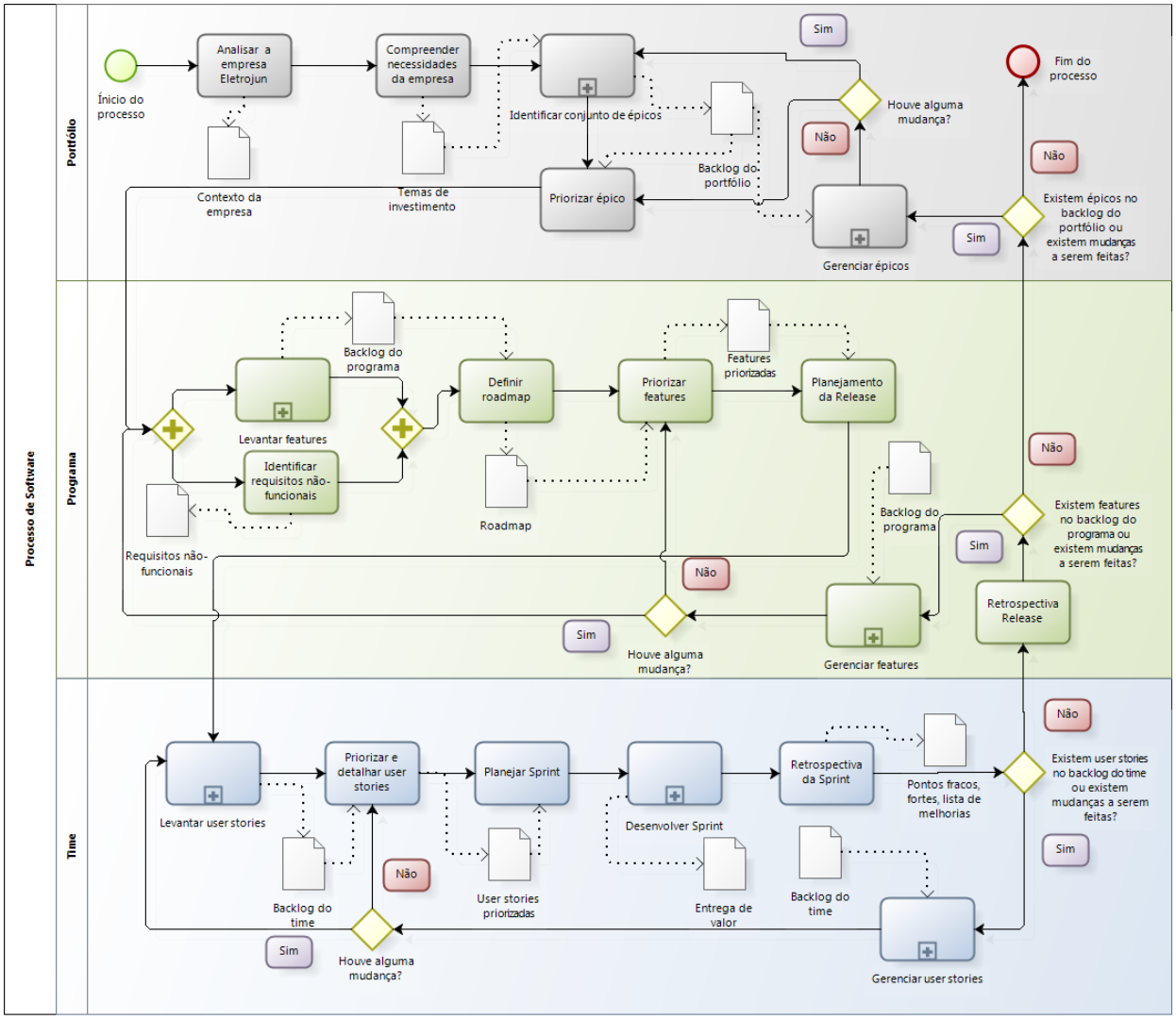

Figura 4 – Visão geral do processo para o projeto da Eletrojun

4.1 Portifólio

Nesta fase, todo o esforço encontra-se diretamente relacionado ao nível de negócio. Procura-se compreender o contexto do cliente, obter o conhecimento inicial do problema, identificar características e diretrizes que posteriormente resultarão no levantamento dos Épicos. Estes resultados serão obtidos através da efetivação das seguintes tarefas:

- Analisar a empresa Eletrojun
- Compreender necessidades da empresa
- Identificar conjunto de Épicos
- Priorizar Épico
- Gerenciar Épico

Tabela 1 – Atividade 1

ID	01
Nome	Analisar a empresa Eletrojun.
Objetivo	Levantar informações sobre a empresa e seu negócio.
Entrada	
Saída	Informações sobre o contexto da empresa.

Tabela 2 – Atividade 2

ID	02
Nome	Compreender as necessidades da empresa.
Objetivo	Realizar entrevista com o cliente no intuito de levantar necessidades gerais
Entrada	Contexto da empresa
Saída	Temas de investimento

Tabela 3 – Atividade 3

ID	03
Nome	Identificar conjunto de Épicos.
Objetivo	Levantar conjunto de Épicos, analisando e estudando os temas de investimento.
Entrada	Temas de investimento.
Saída	Backlog do portifólio.

Tabela 4 – Atividade 4

ID	04
Nome	Priorizar Épico.
Objetivo	Selecionar Épico dentre o Backlog de portifólio e dar preferência ao seu desenvolvimento.
Entrada	Backlog do portifólio
Saída	Backlog do portifólio com Épicos priorizados.

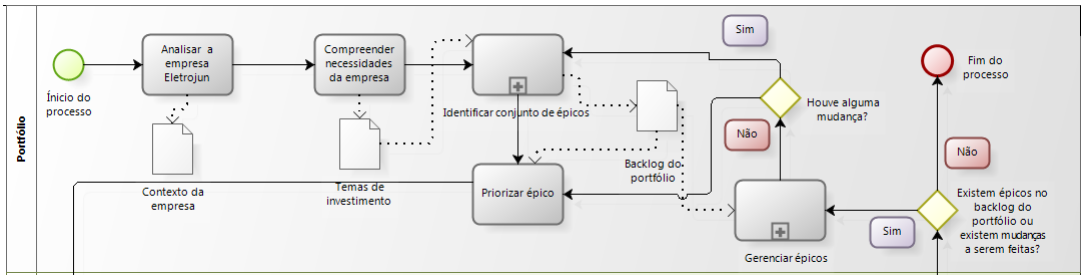

Figura 5 – Visão geral do nível de Portifolio

4.2 Programa

Na fase de programa, o principal objetivo encontra-se em estabelecer estratégias para elaboração da solução a ser implementada, a partir da obtenção de requisitos que atendam aos sete parâmetros de qualidade (Apendice x). Estes resultados serão obtidos através da efetivação das seguintes tarefas:

- Levantar Feature
- Identificar requisitos não-funcionais
- Definir Roadmap
- Priorizar Features
- Planejamento da Release
- Gerenciar Features zitem Planejamento da Release

Tabela 5 – Atividade 5

ID	05
Nome	Levantar Features.
Objetivo	Levantar conjunto de Features dentre o Backlog de portifólio e elaborar o Backlog de p
Entrada	Backlog do portifólio.
Saída	Backlog do programa.

Tabela 6 – Atividade 6

ID	06
Nome	Definir Roadmap.
Objetivo	Planejar as entregas/releases a partir da elaboração do Roadmap .
Entrada	Backlog do programa.
Saída	Roadmap.

Tabela 7 – Atividade 7

ID	07
Nome	Priorizar Features.
Objetivo	Selecionar Features dentre o Backlog de programa, a partir de uma análise do Roadmap, faze
Entrada	Roadmap, Backlog de programa.
Saída	Features priorizadas.

Tabela 8 – Atividade 8

ID	08
Nome	Planejamento da Release.
Objetivo	Realizar o planejamento para a release, definir os pontos chaves.
Entrada	Features priorizadas.
Saída	

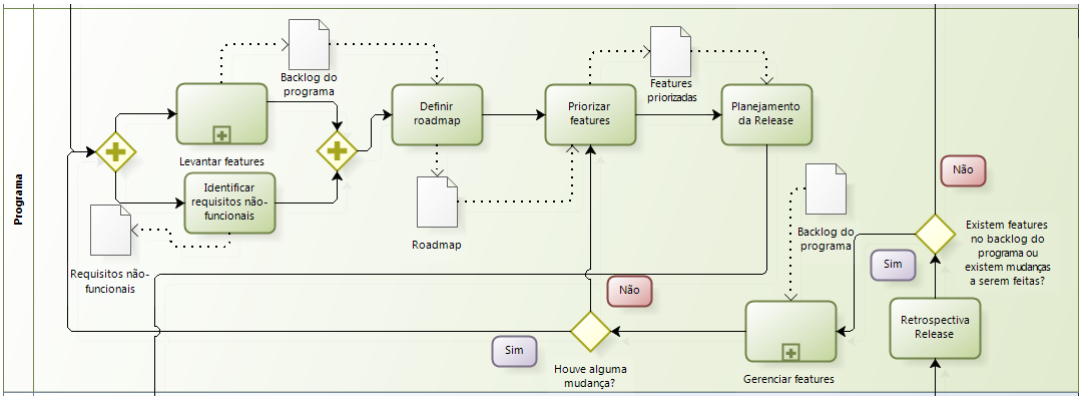

Figura 6 – Visão geral do nível de Programa

4.3 Time

O nível Time abrange as tarefas a nível de implementação. Seu principal objetivo encontra-se no planejamento e implementação das funcionalidades definidas nas User Stories. Estes resultados serão obtidos através da efetivação das seguintes tarefas:

- Levantar User Stories
- Priorizar e detalhar User Stories
- Planejar Sprints
- Desenvolver Sprints
- Retrospectiva da Sprint
- Gerenciar User Stories

Tabela 9 – Atividade 9

ID	09
Nome	Levantar user stories.
Objetivo	Realizar o levantamento das histórias de usuário e com isso elaborar o backlog do time.
Entrada	Features priorizadas.
Saída	Backlog do time

Tabela 10 – Atividade 10

ID	10
Nome	Planejar Sprint.
Objetivo	Realizar o planejamento da sprint a ser desenvolvida, definir e distribuir tarefas.
Entrada	Backlog do time com User Stories priorizadas.
Saída	Kanban atualizado.

Tabela 11 – Atividade 11

ID	11
Nome	Desenvolver Sprint.
Objetivo	Executar Sprint, desenvolvendo as user stories.
Entrada	Backlog do time atualizado, kanban.
Saída	Entrega de Valor.

Tabela 12 – Atividade 12

ID	12
Nome	Retrospectiva da Sprint.
Objetivo	Realizar reunião de retrospectiva da Sprint, de modo a discutir os pontos positivos e ne
Entrada	
Saída	Lista de melhorias a serem implementadas, pontos fortes e fracos da sprint, aspectos a

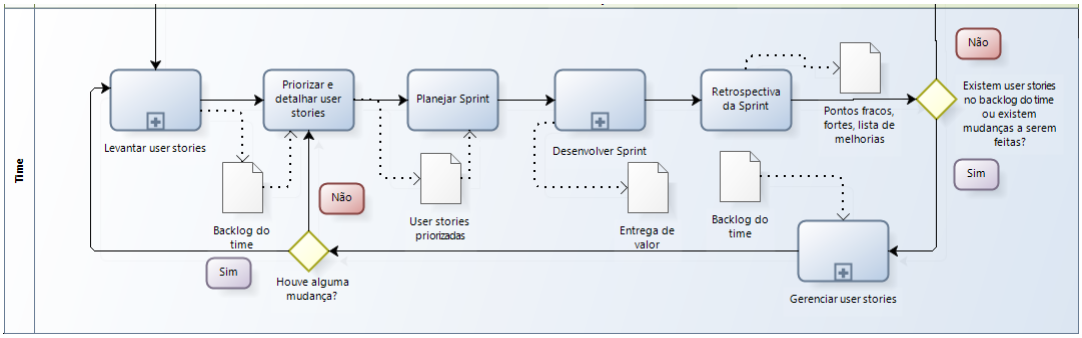

Figura 7 – Visão geral do nível de Time

5 Elicitação de Requisitos

A atividade de elicitação de requisitos tem o objetivo de entender junto ao cliente as necessidades do negócio e identificar as características do sistema. É importante ressaltar que essa é uma fase crucial no processo de software, e para se ter êxito serão utilizadas algumas técnicas de elicitação.

Por ser um processo muita das vezes complexo, o Sommerville cita algumas dificuldades para o processo de elicitação de requisitos, que são elas:

1- Os stakeholders muitas das vezes não sabem o que querem do sistema. Eles entendem o sistema de maneira geral, e encontram muitas dificuldade em articular as especificidades do sistema, podendo fazer pedidos não realistas por não saberem o custo de suas solicitações.

2- Os stakeholders usam termos técnicos que imprimem um conhecimento implícito e cabe aos engenheiros de requisitos conhecerem a área de conhecimento do cliente.

3- Diferentes stakeholders tem em mente diferentes requisitos, dificultando o trabalho dos engenheiros, que precisam identificar as semelhanças e os pontos conflitantes dos requisitos.

4- Fatores hierárquicos e políticos afetam o sistema. Eles podem fazer com que gerentes definam requisitos para aumentar seu poder de influência dentro da organização.

5- A adaptabilidade do sistema é um fator muito importante. Dentro do ambiente econômico e dos negócios a mutabilidade dos requisitos é inevitável, podendo mudar durante o processo de elicitação.

O Sommerville aborda a elicitação de requisitos a princípio de maneira mais simplificada, tendo como abordagem um modelo interativo incremental, com feedbacks contínuos entre as atividades. Que são elas:

“Compreensão do domínio: Os analistas devem desenvolver sua compreensão do domínio da aplicação;

Coleta de requisitos: É o processo de interagir com os stakeholders do sistema para descobrir seus requisitos. A compreensão do domínio se desenvolve mais durante essa atividade;

Classificação: Essa atividade considera o conjunto não estruturado dos requisitos e os organiza em grupos coerentes;

Resolução de conflitos: Quando múltiplos stakeholders estão envolvidos, os requisitos apresentarão conflitos. Essa atividade tem por objetivo solucionar esses conflitos;

Definição das prioridades: Em qualquer conjunto de requisitos, alguns serão mais importantes do que outros. Esse estágio envolve interação com os stakeholders para a definição dos requisitos mais importantes;

Verificação de requisitos: Os requisitos são verificados para descobrir se estão completos e consistentes e se estão em concordância com o que os stakeholders desejam do sistema."

Ratificando a informação supracitada, o modelo abaixo mostra uma imagem referente ao processo de levantamento e análise de requisitos.

IMAGEMMMMMMMMMMMMMMMMMM

5.1 Técnicas de elicitação de requisitos

5.1.1 Levantamento orientado a ponto de vista

Em sistemas de diferenciados tamanhos, tanto pequenos, como médios, como grandes existem diferentes pontos de vista. Por exemplo, em um sistema de caixa automático bancário os clientes do banco vão receber os serviços do sistema, os representantes de outros bancos que tem parceria no uso do caixa automático, gerentes de agência que obtém dados do sistema, administradores de banco de dados que vinculam os dados da conta do cliente com os dados que vão ser mostrados no caixa automático, gerentes de segurança bancária, que vão assegurar a integridade do sistema, o departamento de marketing para fazer divulgação dos produtos do banco e os responsáveis pela manutenção do caixa automático, que vão garantir a boa integridade do sistema. A relação supracitada evidencia que em um sistema relativamente simples, os pontos de vista são diferentes, contudo não independentes, existindo muitos pontos de interseção entre eles, bem como pontos conflitantes e/ou de duplicidade.

Na elicitação baseada em pontos de vista, ela reconhece esses diferentes pontos de vista para a estruturação e organização dos processos de elicitação de requisitos, e também dos requisitos. As vantagens do levantamento por ponto de vista é uma visão mais ampla no sistema, além de uma facilidade maior de identificar conflitos entre os requisitos que os stakeholders propuseram.

Existem abordagens de elicitações por pontos de vista como SADT (Ross, 1977; Schoman e Ross 1977) e o CORE (Mullery, 1979) que citam a elicitação de requisitos como uma fonte de dreno de dados, onde os pontos de vista vão ser responsáveis pela parte que produz ou consome dados. Esses dois métodos são os primeiros a propor pontos de vista explícitos.

Outros autores definem a análise por ponto de vista como um framework de repre-

sentação, onde o ponto de vista é considerado um tipo particular de modelo de sistema (Finkelsten et all..1990; Nuseibeh et all, 1994). Podemos exemplificar essa citação com diferentes engenheiros fazendo um diagrama de sequência, e outros fazendo um diagrama de classes, cada um dos diagramas feitos vai identificar uma abordagem diferente do sistema.

(Kotonya, Sommerville, 1994, 1996) definem o levantamento orientado a ponto de vista como um receptor de serviços, onde os pontos de vista são externos ao sistema, e do ponto de vista recebem serviços.

Apesar de existirem diferentes métodos de análises de pontos de vista citados acima, nenhum deles é adequado para a estruturar uma elicitação de requisitos, porque não existe nenhuma relação simples entre os pontos de vista e os stakeholders.

Em contraposição, sistemas interativos visam integrar da melhor forma os stakeholders. Consequentemente uma abordagem mais eficaz.

5.1.2 Prototipagem

Os protótipos são úteis para a elicitação de requisitos, porque os usuários poderão experimentar com o sistema e mostrar os pontos fortes e fracos do sistema. Eles terão algo concreto para criticar. (Castro, 2015).

Atualmente existem dois tipos de prototipagem, a descartável a e evolucionária. A descartável procura ajudar na na elicitação e no desenvolvimento dos requisitos, os requisitos que vão prototipados são aqueles que os clientes tem mais dificuldades para especificar, os requisitos bem interpretados não precisam de implementação.

Já a prototipação evolucionária tem como objetivo a entrega rápida de um sistema funcional para o cliente, assim, os requisitos que vão ser implementados serão aqueles que estão bem definidos pelo cliente. Entre os benefícios da prototipagem podemos ressaltar alguns pontos, que são eles:

- O protótipo permite que os usuários experimentem e descubram o que eles realmente necessitam para suportar o trabalho deles.
- Estabelece viabilidade e utilidade antes que altos custos de desenvolvimento tenham sido realizados.
- A prototipagem força estudos detalhados dos requisitos, apontando inconsistências e omissões.

5.1.3 JAD

O JAD é uma metodologia aplicada pela IBM em 1977, e implantada no Brasil em 1982. é uma técnica baseada em em reuniões, que permite que todos tenham um

visão do produto. Sua principal finalidade é acelerar o desenvolvimento de software, essa metodologia está sendo empregada em amplas áreas de projeto (Camila, 2014).

Os princípios da metodologia é a realização de dinâmicas em grupo, utilização de recursos audiovisuais, analisar o projeto de forma completa (top-down), garantindo que todas as partes do processo sejam abrangidas, utilização de uma documentação padrão, para que possa ser entendida por todos os membros do grupo.

O JAD se divide em duas etapas: Planejamento e projeto, onde no planejamento ocorre a elicitação dos requisitos, e na fase de projeto ocorre a gerência e desenvolvimento do sistema. Cada etapa possui três fases: Adaptação, sessão e finalização, onde na adaptação se prepara o material utilizado nas reuniões, as sessões seriam as sessões de reuniões, e na finalização, converte-se os documentos extraídos em documentos.

Os benefícios do JAD são o trabalho em equipe, maior produtividade e qualidade, além do baixo custo.

5.1.4 Brainstorming

Neste tipo de reunião, representantes de diferentes grupos de interessados engajam-se em uma discussão informal para gerar rapidamente tantas ideias quanto possível, sem focar a atenção em nenhuma delas(Aurum; Wohlin, 2005).

Como o nome já diz, o processo supracitado é um período de Brainstorming, se traduzirmos ao pé da letra: tempestade cerebral. Em geral esse período de tempestade cerebral tem como objetivo resolver um grande número de questões, ou tomar decisões. Geralmente essa técnica é usada para se obter uma visão preliminar do sistema. Uma das grandes vantagens do Brainstorming, é que ao permitir a livre expressão dos componentes da reunião, conseguem chegar soluções criativas e inovadoras para o sistema.

5.1.5 Entrevista

A entrevista visa identificar o problema, propor a solução, negociar os aspectos de abordagem e identificar um conjunto preliminar de requisitos do projeto. (9)

- As entrevistas serão conduzidas com a participação dos engenheiros de software e outros interessados.
- A agenda de entrevistas deve possibilitar uma entrevista que cubra todos os pontos importantes, e abra espaço para o fluxo de novas ideias.
- Deve haver um controle da reunião através de um plano de entrevista, para que não haja uma dispersão do conteúdo pertinente ao entrevistador.

5.2 Elicitação Escolhida

As técnicas são utilizadas para identificar os requisitos conscientes, inconscientes e subconscientes dos stakeholders. Cada projeto possui características específicas, e por isto, algumas técnicas são melhor aplicadas em determinado contexto. Segundo (20) os fatores mais influentes para a escolha das técnicas são:

- Distinção entre os requisitos conscientes, inconscientes e subconscientes a serem elicitados.
- As restrições em termos de tempo e de orçamento, bem como de disponibilidade dos stakeholders.
- A experiência do engenheiro de requisitos com determina técnica de elicitação.
- As oportunidades e risco do projeto.

Levando em consideração esses aspectos, foi feita uma análise do contexto para o levantamento de requisitos, percorrendo e analisando as diversas técnicas que são usadas atualmente, como o levantamento orientado a ponto de vista, etnografia, prototipagem, entrevista, questionário, brainstorming e JAD.

Partindo do princípio que já existe uma parte do nosso software pronto, e que a cliente conhece bem os requisitos e tem noções de desenvolvimento de software, técnicas como JAD, apesar de promover a cooperação, o trabalho em grupo e o entendimento, acabam não se adaptando, devido sua metodologia ser baseada na criação e na resolução de problemas, onde os desenvolvedores ajudam os usuários do sistema a criarem problemas e eles mesmos proporem soluções.(9)

O Brainstorming é uma técnica que no começo suas ideias não parecem convencionais, mas são encorajadas, estimulando frequentemente os participantes para a elaboração de soluções criativas. Mas a exploração de novas ideias não se aplica ao nosso caso, porque a ideia do software já existe.(9)

A prototipagem parte do pressuposto de fazer protótipos para a validação do cliente, implementando os aspectos críticos do sistema. Porém, já existe um protótipo validado pela cliente, e esta técnica ao ser adotada geraria um retrabalho.

O Sommerville recomenda a elicitação por pontos de vista. Apesar dessa elicitação ser bem respaldada acima por diversos autores de renome na elicitação de requisitos, como Finkelsten, Schoman, Ross, Mullery, Nuseibeh e Kotonya, os pontos de vista de diferentes stakeholders são necessários para a definição de requisitos, e atualmente temos apenas a visão de uma pessoa do grupo, e consequentemente não temos recurso necessário para a aplicação.

Após a análise destas ferramentas, chegou-se a conclusão que a técnica melhor ser aplicada a este projeto é a entrevista. Ela se enquadra melhor visto que o projeto já teve uma parte iniciada e que o grupo não possui limitações em termos de disponibilidade, distância, podendo assim serem feitas reuniões semanalmente utilizando a técnica de entrevista.

6 Gerenciamento de Requisitos

O gerenciamento de requisitos é um processo que consome muitos recursos. neste processo deve-se decidir sobre:

1 - Identificação de requisitos: Cada requisito deve ser identificado unicamente de modo que possa ser feita a referência cruzada entre este e outros requisitos para que ele possa ser utilizado nas avaliações de rastreabilidade. 2 - Processo de gerenciamento de mudanças: É o conjunto de atividades que avaliam o impacto e custo das mudanças. 3 - Políticas de rastreabilidade: Essas políticas definem os relacionamentos entre os requisitos e o projeto do sistema, que devem ser registrados, e como estes registros devem ser mantidos. 3 - Apoio de Ferramentas: O gerenciamento de requisitos envolve o processamento de grandes quantidades de informações sobre os requisitos. As ferramentas que podem ser usadas variam desde sistemas especializados de gerenciamento de requisitos a planilhas e sistemas simples de banco de dados. (SOMMERVILLE, 2007, p.108).

6.1 Atributos de Requisitos

Requisitos não são constituídos apenas pela especificação do que é requerido, mas também por um conjunto de informações adicionais que auxiliam a interpretar e gerenciar os requisitos. (25)

Buscando uma melhor forma de gerenciar, foram definidos os atributos que cada requisito deverá conter, e são eles:

1 - Origem

Os requisitos possuirão um código único que será sua identificação.

2 - Status

Irá indicar o grau de completude do requisito, podendo ser: Completo: requisito já foi implementado no sistema.

- Em progresso: requisito está sendo implementado no sistema.
- Não iniciado: indica que o requisito está no backlog mas não foi implementado ainda.

Tabela 13 – Identificação dos requisitos

E	Épico
F	Feature
H	User Story

3 - Prioridade

Este atributo indicará o nível de importância para os stakeholders, sistema e outros requisitos.

- Alta: quando o requisito é de suma importância para os stakeholders ou quando sem ele o sistema não funciona.
- Média: requisito que é importante para os interessados, mas que ainda sem ele o sistema funciona de forma básica.
- Baixa: requisito que não compromete o funcionamento do sistema, podendo ser uma funcionalidade opcional.

4 - Complexidade

Indica nível de dificuldade para implementar o requisito em questão.

- Alta: requisito com grau de dificuldade elevado, sendo necessário um grande esforço da equipe para implementação.
- Média: requisito com grau de dificuldade médio, sendo necessário um esforço da equipe mas de forma moderada.
- Baixa: requisito com baixo grau de dificuldade de implementação.

5 - Risco

Requisitos que apresentam alguma possibilidade de risco para o sistema durante o desenvolvimento.

- Alto: requisito com grande possibilidade de risco, necessita de uma atenção maior da equipe.
- Médio: requisito com média possibilidade de risco.
- Baixo: requisito com baixa possibilidade de risco.

Os requisitos seguirão este padrão:

6.2 Rastreabilidade de requisitos

Segundo Sommerville, “A rastreabilidade é a propriedade de uma especificação de requisitos que reflete a facilidade de encontrar requisitos relacionados”. No conceito de de

Tabela 14 – Atributos do Requisito

Origem	E, F ou H
Status	Completo, em progresso ou não iniciado.
Prioridade	Alta, média ou baixa.
Complexidade	Alta, média ou baixa
Risco	Alto, médio ou baixo.

requisitos de software, diz respeito a capacidade de se acompanhar um requisito em todo seu ciclo de vida, permitindo que se encontre artefatos e outros requisitos que estejam alinhados de alguma forma. É essencial a identificação da composição dos requisitos, das suas dependências entre outros, dos requisitos conflitantes e ainda sua origem e seus interessados, além da identificação de em quais artefatos produzidos o requisito é retratado (27).

Propriedade básica que deve estar presente em todos os tipos de rastreabilidade, de forma a capacitar o completo entendimento das funções, é a capacidade de rastrear para frente (Forwards) e para trás (Backwards), tal que o primeiro permite o rastreio de um requisito até seus refinamentos e o segundo permite o rastreio dos refinamentos até a sua origem. (28)

Sobre os tipos de rastreabilidade, existem basicamente duas classificações gerais: horizontal e vertical e pré e pós-rastreabilidade. (29)

A rastreabilidade horizontal é entre diferentes versões ou variações de requisitos, ou outros artefatos, em uma fase do ciclo de vida particular. A rastreabilidade vertical é realizada entre requisitos e artefatos produzidos pelo processo de desenvolvimento ao longo do ciclo de vida do projeto. Uma visão simplificada sobre os tipos de rastreabilidade é apresentada na Figura a seguir (30):

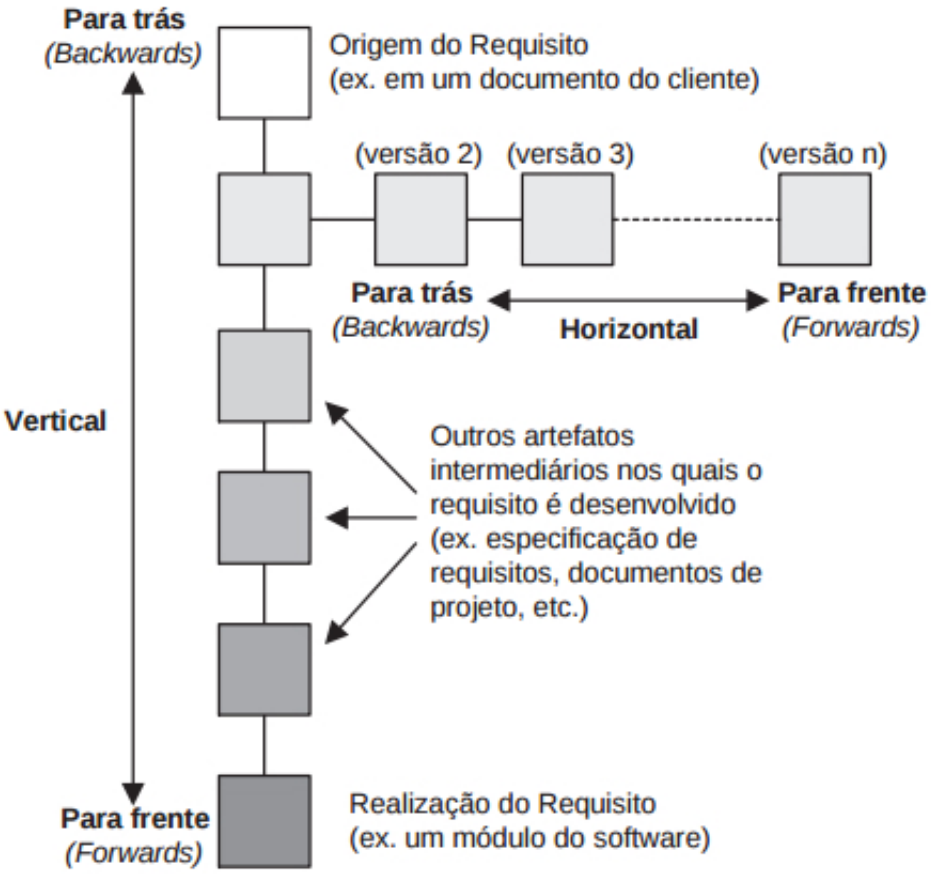

Figura 8 – Rastreabilidade

A segunda classe de rastreabilidade trata da pré-rastreabilidade, que está concentrada no ciclo de vida dos requisitos antes de serem incluídos no processo de especificação, e a pós-rastreabilidade, que está concentrada no ciclo de vida dos requisitos após serem incluídos na especificação de requisitos. A imagem a seguir ilustra a pré e pós-rastreabilidade (30):

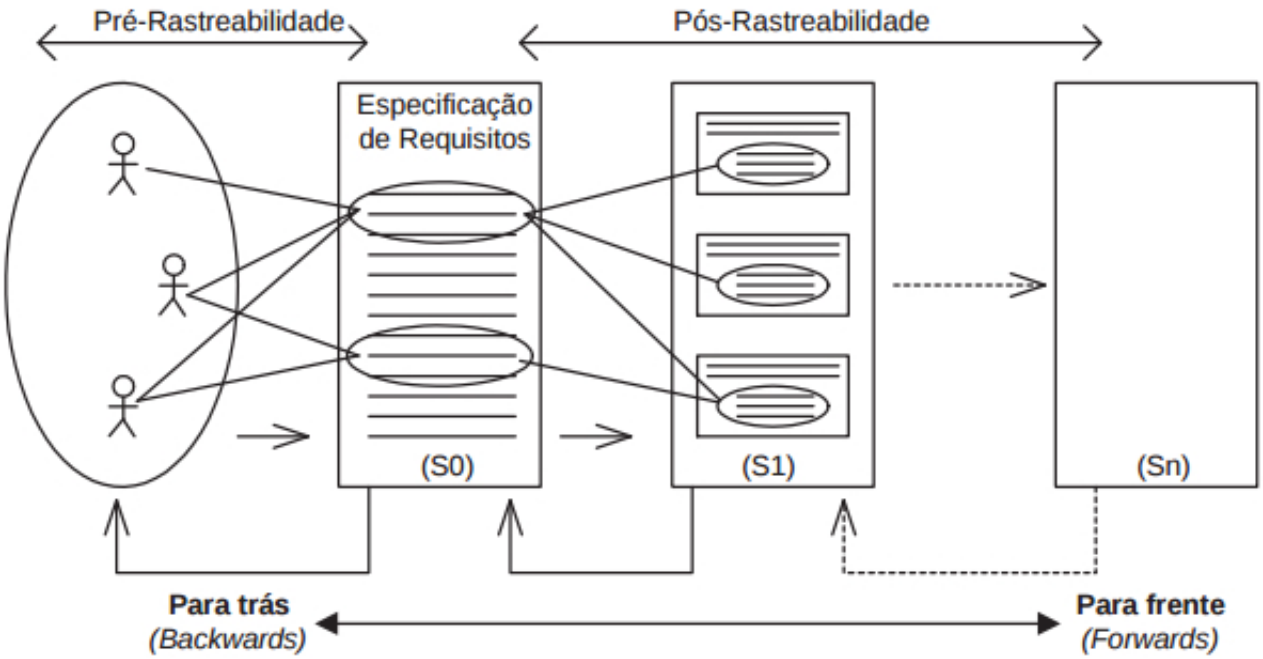

Figura 9 – Pré-rastreabilidade

Tomando por base os conceitos citados, foi adotado pelo grupo um modelo de rastreabilidade bi-direcional, que visa garantir o completo rastreio desde os Temas de Investimento até as Tarefas de cada História. Tais requisitos serão identificados da seguinte forma:

Tabela 15 – Rastreabilidade

Sigla	Descrição	Exemplo
T	Tema de Investimento	T01
T	Tema de Investimento	T01
E	Épico	T02E03
F	Feature	T01E02F05
H	História	T01E02F05H12
TR	Tarefa	T01E02F05H12TR09

A representação em mapa destes requisitos é dada da seguinte forma:

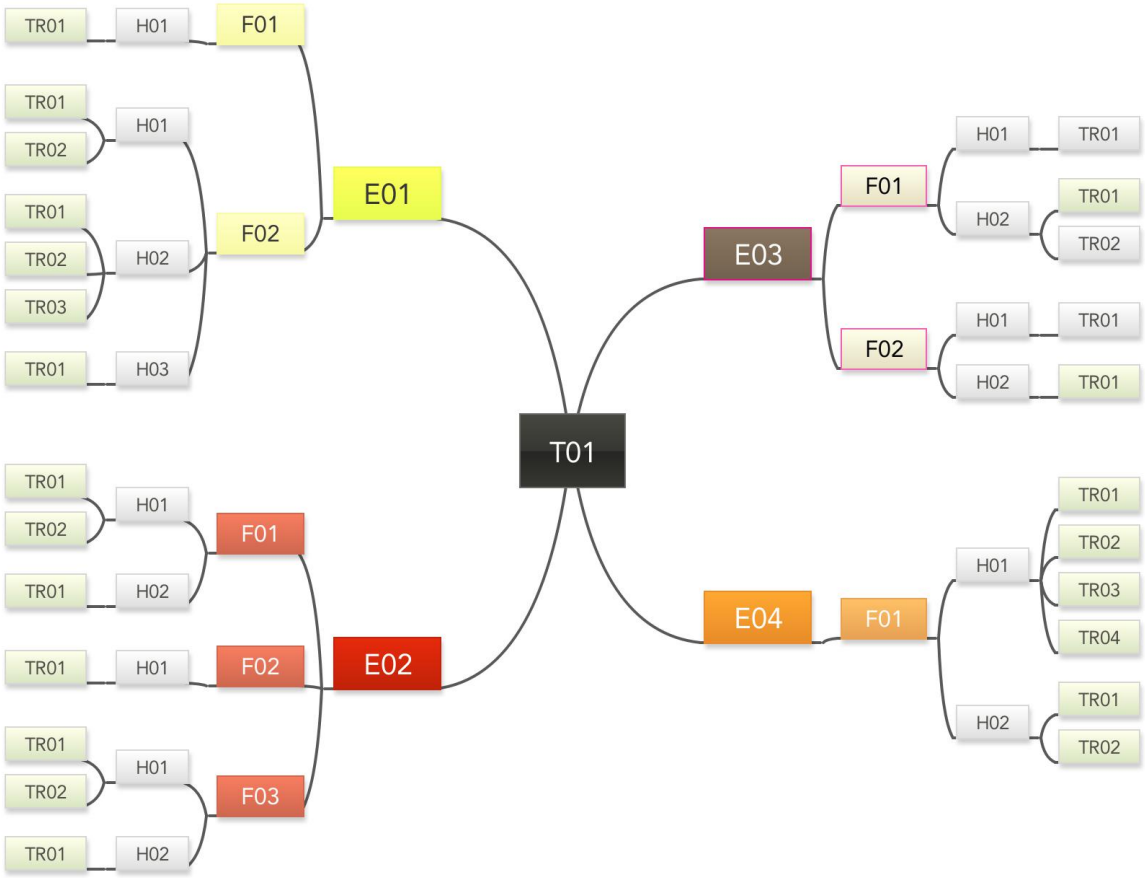

Figura 10 – Mapa Requisitos

7 Planejamento do Projeto

Antes do projeto ser iniciado foi feito um planejamento do mesmo para que pudesse ter uma melhor organização, um controle do tempo e das entregas previstas.

7.1 Cronograma

O cronograma foi feito para guiar o projeto, principalmente para que as atividades pudessem ser cumpridas no tempo determinado. Ele foi elaborado usando a ferramenta Gantter.

7.1.1 Cronograma - Primeira Entrega

		Nome	Duração	Ínicio	Fim	Predecessores	Recursos
1		**Trabalho**	63d?	**29/03/2016**	**23/06/2016**		
2		**Primeira Entrega**	34d?	**29/03/2016**	**13/05/2016**		
3		Reunião dos integrantes do grupo	4d?	29/03/2016	01/04/2016		Equipe de ER
4		Fazer o planejamento do projeto, definir cronograma	5d?	30/03/2016	05/04/2016		Sabryna
5		Primeira reunião com o cliente	1d?	08/04/2016	08/04/2016		Nicácio,Sabryna
6		Segunda Reunião com o cliente	4d?	14/04/2016	19/04/2016		Equipe de ER
7		Criar estrutura do relatório	1d?	15/04/2016	15/04/2016		Pedro
8		Escrever a introdução, contexto do cliente	3d?	15/04/2016	19/04/2016		Nicácio
9		Definir a abordagem de ER	4d?	15/04/2016	20/04/2016		Pedro
10		Escrever sobre o modelo de maturidade escolhido	4d?	15/04/2016	20/04/2016		Sabryna
11		Justificar a abordagem escolhida, fazer elicitação dos requis	4d?	15/04/2016	20/04/2016		Ruan
12		Elaborar o processo de engenharia de requisitos	7d?	21/04/2016	29/04/2016		Nicácio
13		Definir ferramenta de gestão de requisitos	7d?	21/04/2016	29/04/2016		Pedro
14		Definir a rastreabilidade dos requisitos	7d?	21/04/2016	29/04/2016		Ruan,Sabryna
15		Revisar o relatório	1d?	02/05/2016	02/05/2016		Equipe de ER
16		Ponto de Controle 1	1d?	03/05/2016	03/05/2016		Equipe de ER
17		Revisar o relatório com os pontos abordados no ponto de co	4d?	04/05/2016	09/05/2016		Equipe de ER
18		Apresentação 1	4d?	10/05/2016	13/05/2016		Equipe de ER
19		**Segunda Entrega**	29d?	**16/05/2016**	**23/06/2016**		

Figura 11 – Primeira Entrega

7.1.2 Cronograma - Segunda Entrega

		Nome	Duração	Início	Fim	Predecessores	Recursos
18		Apresentação 1	4d?	10/05/2016	13/05/2016		Equipe de ER
19		**Segunda Entrega**	29d?	**16/05/2016**	**23/06/2016**		
20		Criar estrutura do relatório final	1d?	16/05/2016	16/05/2016		Sabryna
21		Fazer o contexto de negócio	1d?	16/05/2016	16/05/2016		Nicácio
22		Explicar sobre o processo escolhido	1d?	16/05/2016	16/05/2016		Pedro
23		Falar sobre as técnicas de elicitação de requisitos	1d?	16/05/2016	16/05/2016		Ruan
24		Entrevista com o cliente	1d?	17/05/2016	17/05/2016		Equipe de ER
25		Levantamento dos épicos	1d?	18/05/2016	18/05/2016		Equipe de ER
26		Revisão e análise dos épicos	1d?	19/05/2016	19/05/2016		Nicácio
27		Levantamento das features	1d?	20/05/2016	20/05/2016		Equipe de ER
28		Planejamento da release	1d?	23/05/2016	23/05/2016		Sabryna
29		Entrevista com o cliente	1d?	24/05/2016	24/05/2016		Equipe de ER
30		Levantamentos das histórias	1d?	24/05/2016	24/05/2016		Equipe de ER
31		Detalhar histórias de usuários	2d?	25/05/2016	26/05/2016		Pedro,Ruan
32		Planejamento das sprints	1d?	27/05/2016	27/05/2016		Equipe de ER
33		Sprint 1	6d?	30/05/2016	06/06/2016		Equipe de ER
34		Retrospectiva da sprint	1d?	07/06/2016	07/06/2016		Equipe de ER
35		Revisar o relatório final	2d?	07/06/2016	08/06/2016		Equipe de ER
36		Ponto de Controle 2	1d?	09/06/2016	09/06/2016		Equipe de ER
37		Finalizar o relatório	4d?	10/06/2016	15/06/2016		Equipe de ER
38		Apresentação final	6d?	16/06/2016	23/06/2016		Equipe de ER

Figura 12 – Segunda Entrega

8 Ferramentas de Gestão de Requisitos

Os principais problemas do desenvolvimento de software estão associados a Engenharia de Requisitos. Requisitos que não refletem as reais necessidades dos usuários, que sejam incompletos ou inconsistentes, mudanças em requisitos já acordados e dificuldade em chegar em um entendimento comum entre usuários e desenvolvedores são grandes dificuldades relatadas, provocando retrabalho, atrasos no cronograma, custos ultrapassados e insatisfação dos clientes e usuários de software (13).

Tendo em vista a grande importância do Processo de Engenharia de Requisitos, foram criadas ferramentas para a gerência de requisitos, visando apoiar os responsáveis pelo desenvolvimento de software. Características importantes na escolha da ferramenta adequada ao projeto estão relacionadas à capacidade de armazenar, manter e gerenciar mudanças dos requisitos, além de permitir a rastreabilidade entre requisitos funcionais, casos de testes, cronogramas e código fonte.

8.1 Análise de Ferramenta de Gestão de Requisitos

8.1.1 Ferramentas

Para o presente projeto foram analisadas três ferramentas, afim de escolher dentre elas a que melhor se adaptasse ao contexto e necessidades do mesmo:

1- CodeBeamer

O CodeBeamer é uma plataforma integrada que oferece um conjunto de recursos de Gerenciamento de Requisitos, Desenvolvimento, Teste, Qualidade e outros. A ferramenta fornece escalabilidade Agile, completo gerenciamento de mudança e controle de processo robusto. Possui suporte ao fluxo de trabalho avançado com recursos de colaboração que apoiam metodologias Ágeis, Cascata e Híbridas, além de Gerenciamento de Ciclo de Vida de Aplicativos SAFe. O CodeBeamer permite o gerenciamento dos requisitos para várias variantes do produto de forma colaborativa em todo o ciclo de vida, buscando consistência e conformidade de forma a garantir que o produto atenda às necessidades das partes interessadas.

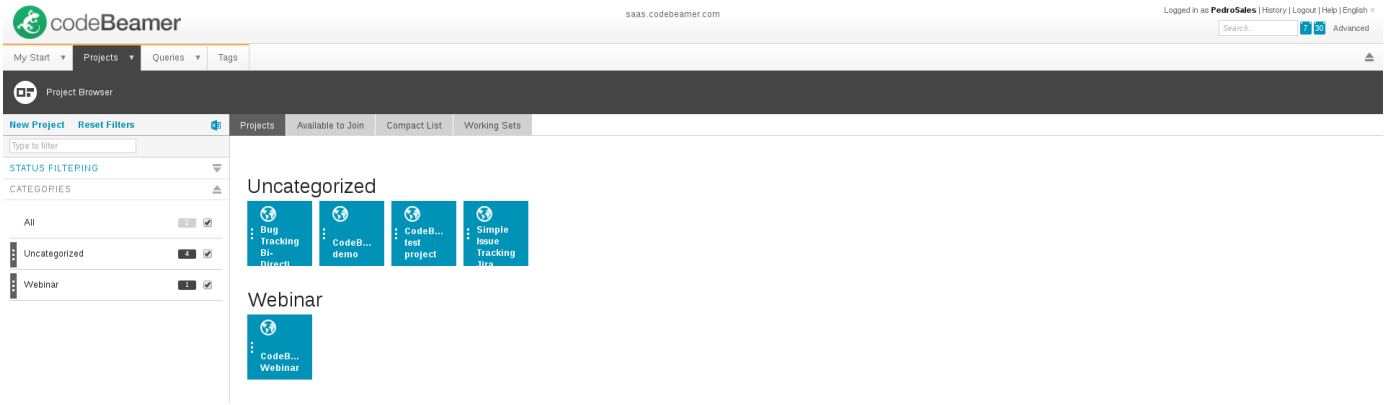

Figura 13 – CodeBeamer 1

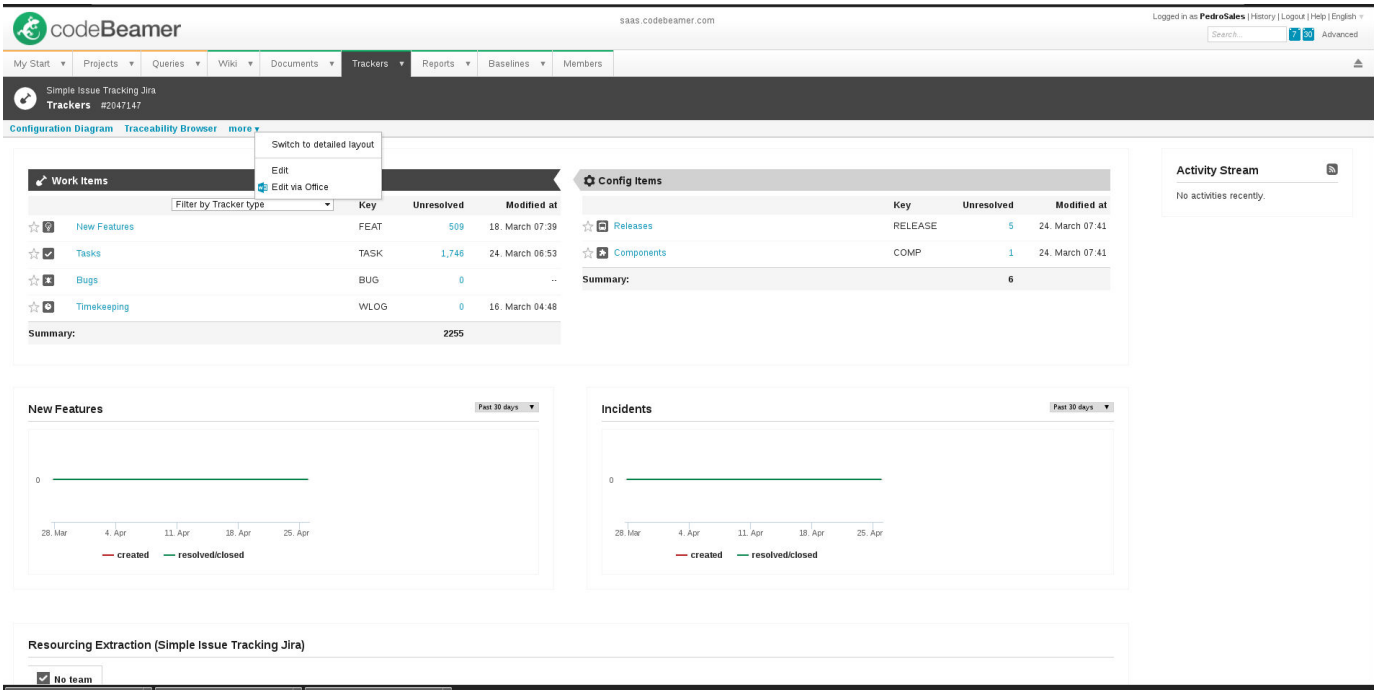

Figura 14 – CodeBeamer 2

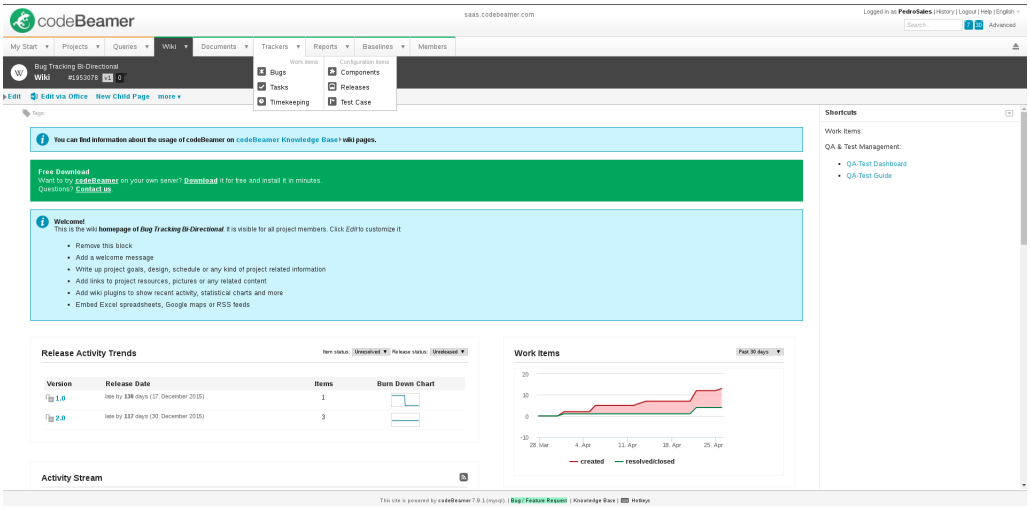

Figura 15 – CodeBeamer 3

2- Caliber

O Caliber vem com a proposta de integrar o setor de TI, e o setor de tomadas de decisões, visando assim eliminar as lacunas presentes entre esses dois setores. Uma das grandes vantagens dessa ferramenta é a parte visual, por ser o primeiro produto corporativo a integrar a definição de requisitos com uma ferramenta visual. O Caliber permite a especificação visual dos requisitos funcionais, a execução de quadros gráficos, modelar casos de utilização, e também de cenários, além disso, a ferramenta gera automaticamente exemplos de teste, para comprovar a qualidade do ciclo de vida do software.

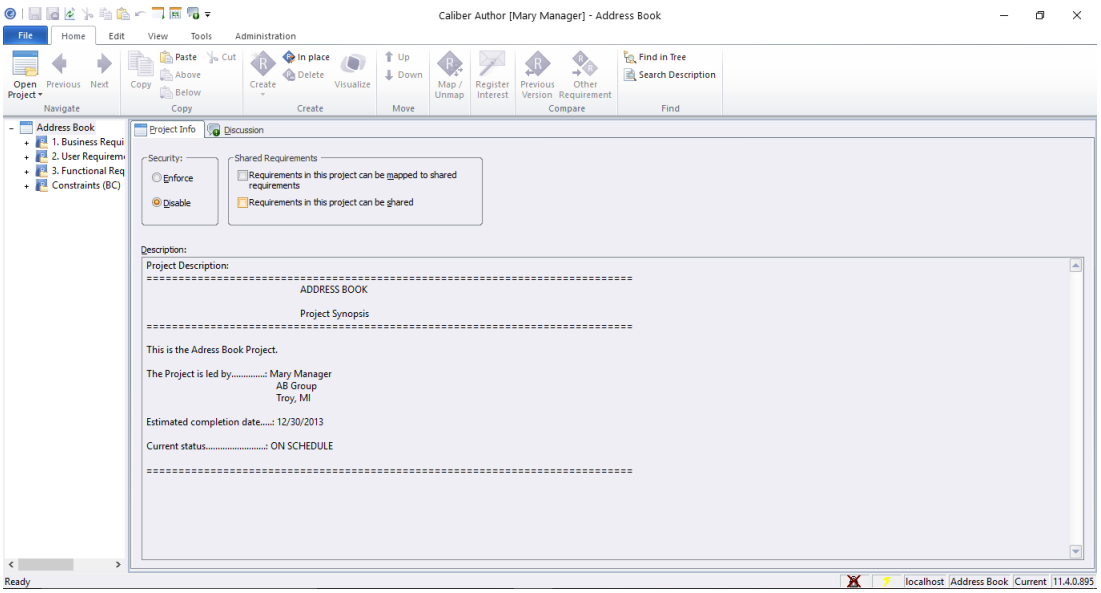

Figura 16 – Caliber 1

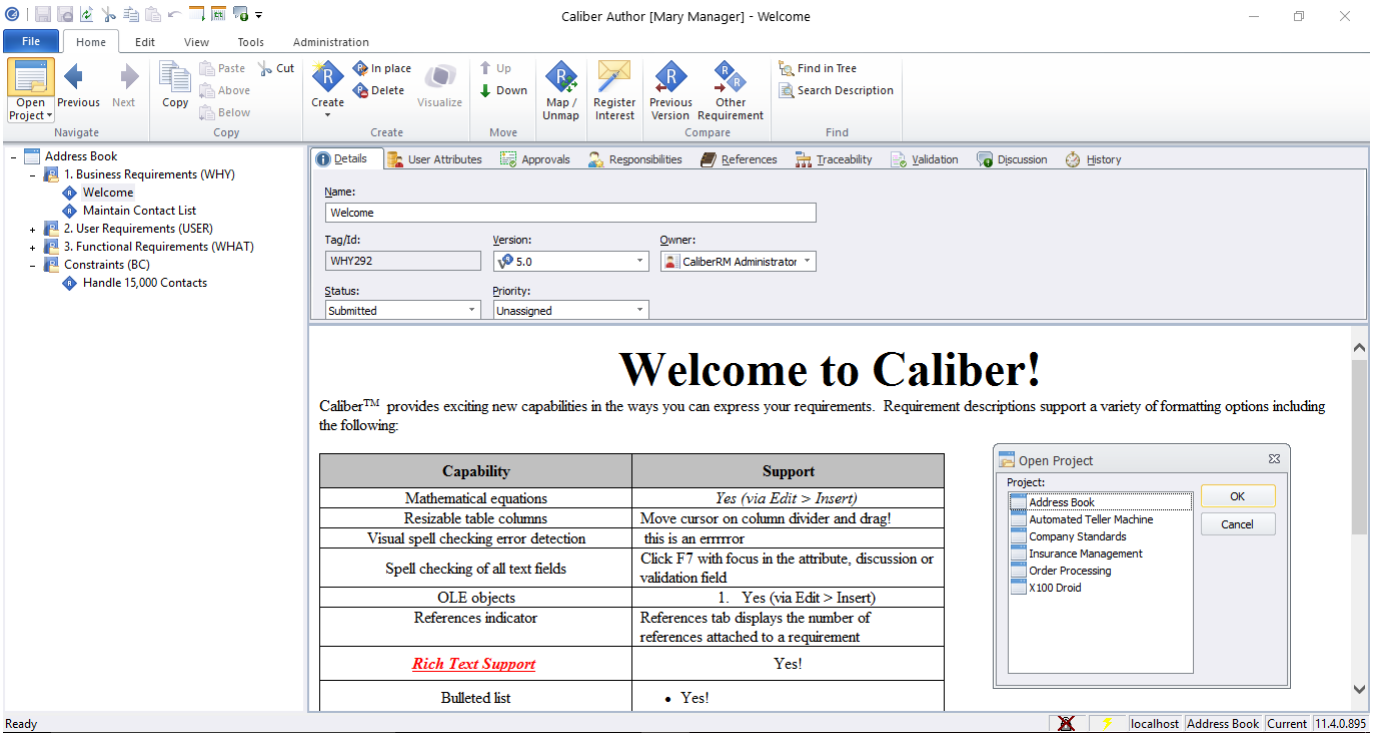

Figura 17 – Caliber 2

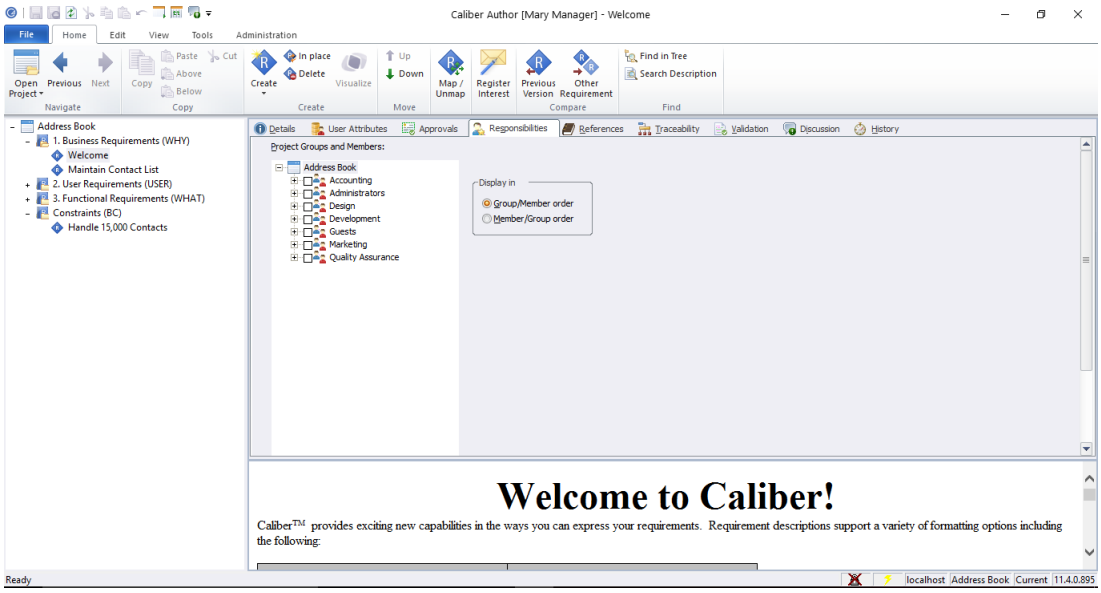

Figura 18 – Caliber 3

3- Traceloud

TraceCloud é uma solução de Gerenciamento de Requisitos Web, projetada para ser flexível, de forma a poder ser aplicado tanto a metodologias ágeis quanto tradicionais, ser ainda leve e com grande poder de gerenciamento. TraceCloud foi construído em um sistema de Business Intelligence, que identifica e apresenta previamente pontos de confli-

tos e possui sistema de gerência de falhas, requisitos e testes totalmente integrados. Possui uma grande quantidade de funcionalidades para: bloquear e controlar alterações, compartilhar requisitos, promover a colaboração, integração, fluxo para aprovação de trabalho, rastreabilidade, dinamicidade, documentação dinâmica, controle de versão, entre outros.

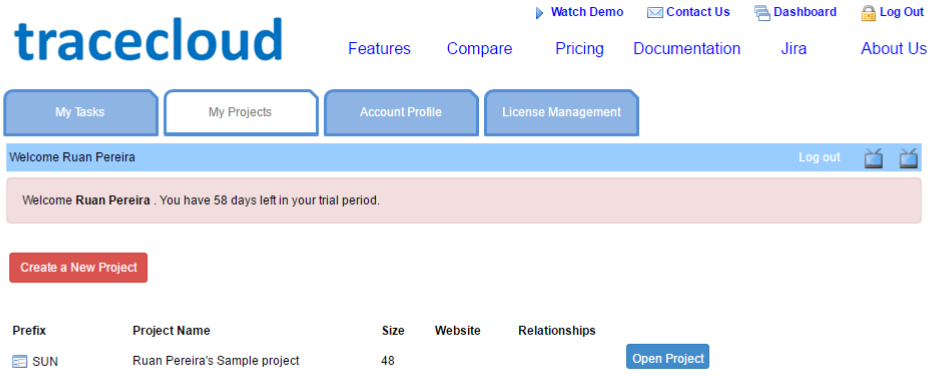

Figura 19 – Traceloud 1

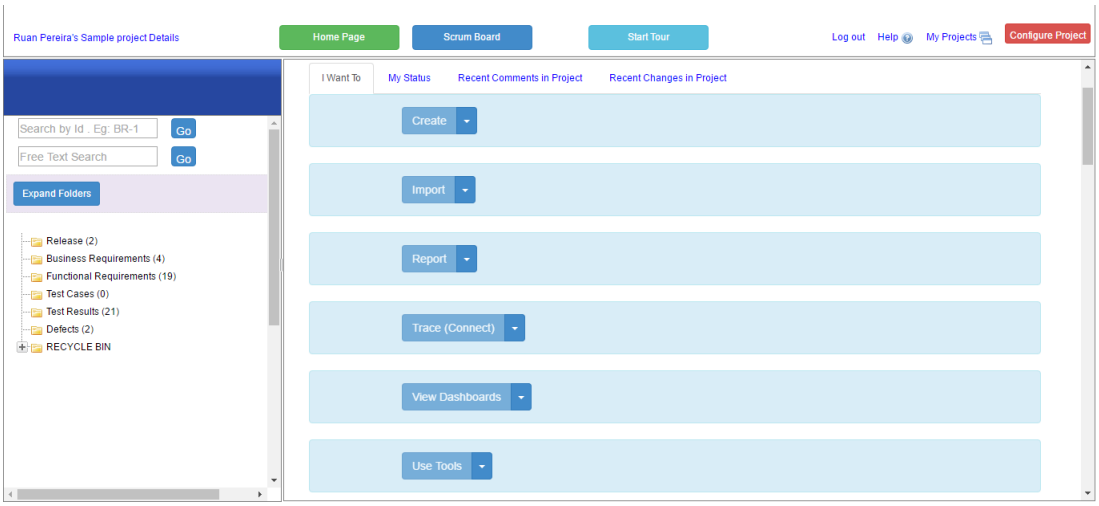

Figura 20 – Traceloud 2

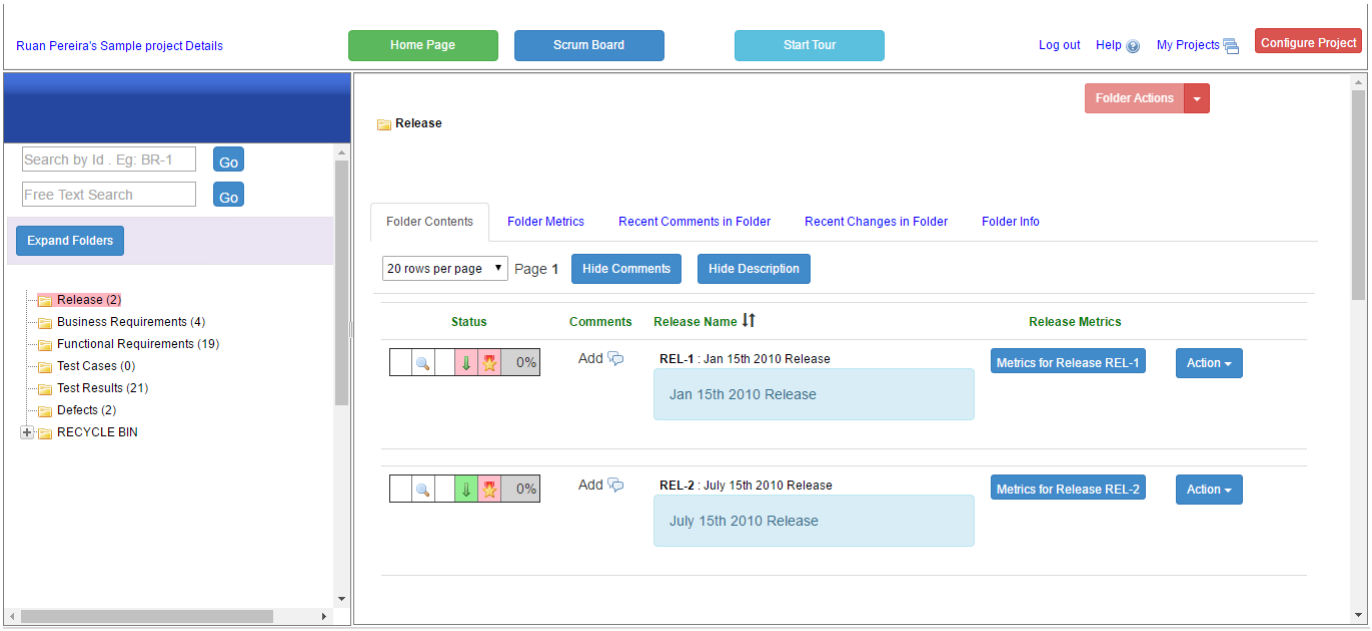

Figura 21 – Traceloud 3

8.1.2 Critérios

Para esta analise, foram listadas as 5 características mais importantes de cada ferramenta para este projeto, referentes à Usabilidade, Rastreabilidade, Gestão de Mudanças, Flexibilidade e Licença.

Em seguidas pontuadas, de 0 a 5 pontos, de forma que, quanto mais próximo da pontuação 5, maior a importância de tal característica para o contexto do projeto, e, consequentemente, a proximidade da pontuação 0 representa o quanto a característica foge do objetivo principal e necessidades para uso da ferramenta neste. Em relação ao critério Licença, a mesma será descrita em relação à ferramenta em enfoque.

8.1.3 Descrição dos Critérios

Tabela 16 – Critérios

Critério	Descrição
Usabilidade	Evidencia o esforço necessário para utilizar o software por um conjun
Rastreabilidade	Eficiência em rastreabilidade bidirecional dos requisitos.
Gestão de Mudanças	O gerenciamento de mudanças está relacionado a procedimentos, proc
Flexibilidade e Compatibilidade	Compatibilidade entre diferentes SO e configurações.
Licença	Relativo a licença de uso da ferramenta.

8.1.3.1 Usabilidade

CodeBeamer - O CodeBeamer tem design simples, favorecendo o entendimento de seus elementos e a navegabilidade, tanto na versão desktop quanto web. A ferramenta funciona de forma colaborativa e permite a integração com outras ferramentas, como MS World e Excel, Matlab, Jira, JUnit, Jenkins, entre outros.

Caliber - A solução viabiliza a colaboração, visualização de qualidade, gerenciamento robusto e capacidade de acompanhamento dos requisitos nos planos de fornecimento ágil. Permite a visualização dos requisitos como fluxos de usuários interativos e passíveis de teste para aumentar a clareza e a precisão.

TraceCloud - Tudo relacionado a um requisito, como rastreabilidade, atributos, status de teste, log, informação de versão, status de aprovação e informação básicas estão convenientemente localizado em uma página. Isto reduz a cliques do mouse e torna o sistema mais utilizável. Usa um código de cores para mostrar visualmente requisitos incompletos, que não foram testados ou com problema de rastreabilidade.

8.1.3.2 Rastreabilidade

CodeBeamer - O CodeBeamer garante a rastreabilidade e o gerenciamento de dependências estabelecendo links entre os artefatos. Possui Treceability Browser, uma tabela para mapeamento da rastreabilidade, relacionando dependências rastreadas.

Caliber - O Caliber fornece a geração automática e o relato de tabelas de rastreamento personalizado. O controle integrado detecta imediatamente elementos órfãos e destaca rastros suspeitos, para fornecer uma excelente transparência da qualidade do seu projeto.

TraceCloud - TraceCloud da pleno acesso e controle à instrumentos de rastreabilidade, como TraceMatrix, TraceTrees, Análise de Impacto de Mudanças (CIA), tendo como objetivo ainda a usabilidade, desempenho e escalabilidade.

8.1.3.3 Gestão de Mudanças

CodeBeamer - O CodeBeamer mantém o controle de mudança com um histórico completo de todos os ajustes de requisitos. Fluxos avançados de trabalho permitem personalização do processo, adicionando mecanismos de segurança para garantir que somente pessoas autorizadas façam alterações nos requisitos.

Caliber - O Caliber aposta em uma maior integração entre os requisitos de definição e gestão dos projetos para desenvolvimento de software, de forma a diminuir o tempo de resposta às mudanças de requisitos. O Caliber não possui mecanismo específico para gerência de mudanças de requisitos.

TraceCloud - O TraceCloud possui um sistema para análise de níveis de rastreabilidade para mudanças de requisitos. Também é possível fazer definições de papéis para controle e permissões de modificação de acordo com funções atribuídas aos membros da equipe.

8.1.3.4 Flexibilidade e Compatibilidade

CodeBeamer - CodeBeamer fornece versões desktop para Windows, Mac e Linux, além da versão em camada SaaS. Conta com aplicativos customizados, integração com pacote Office, softwares de Design, Modelagem, Gerenciamento de Defeitos, entre outros.

Caliber - Suportado somente em Windows 7, Windows 8/8.1, Windows Vista e Windows XP. Possui integração com outros plug-ins da Micro Focus, além de integração com Microsoft Visual Studio e Eclipse.

TraceCloud - Aplicação em camada SaaS. Possui integração com pacote Office (Word e Excel) e faz integração com projetos já existentes via serviço RESTFul Web API (JASON).

8.1.3.5 Licença

CodeBeamer - Possui versão trial de avaliação de 30 dias, havendo alternativas de licença anual ou vitalícia do produto, que variam de 1080 a 3240 euros para anual e 1800 a 5400 euros para vitalício.

Caliber - Possui versão de avaliação de 30 dias. Não foram encontradas informações para valores de contratação.

TraceCloud - Possui versão gratuita por tempo limitado. Para as versões pagas existem as licenças para usuário: leitura e estrita, com criação ilimitada de projetos e requisitos por 30 dólares por mês; somente leitura, com criação ilimitada de projetos e requisitos por 25 dólares por mês. Existe também a licença para projeto: 350 dólares por mês para projetos com menos de 5000 requisitos, 700 dólares para projetos entre 5000 e 10000 requisitos e 1000 dólares por mês para projetos com mais de 10000 requisitos.

8.1.4 Pontuação dos Critérios

8.1.5 Ferramenta Escolhida

Tomando por base os critérios descritos no tópico Descrição dos Critérios (8.1.3) e a pontuação atribuída pelo grupo a cada uma das ferramentas (Pontuação dos Critérios - 8.1.4), pode-se concluir que a ferramenta TraceCloud apresentou-se como a melhor alternativa dentre as ferramentas consideradas. Isto se dá por obter nota máxima em Usabilidade e Rastreabilidade, que são critérios de extrema importância para o sucesso do projeto, e ainda nota alta em Gestão de Mudanças, que é de grande importância, e

Tabela 17 – Pontuação ferramentas

	CodeBeamer	Caliber	Traceloud
Usabilidade	3	4	5
Rastreabilidade	5	5	5
Gestão de Mudanças	5	2	4
Flexibilidade e Compatibilidade	5	3	4
Licença	3	3	4
Total	21	17	22

em todos os outros critérios, de forma a não apresentar discrepâncias e pontos fracos que possam de alguma forma vir a afetar de forma prejudicial o pleno desenvolvimento do trabalho, impedindo-o de obter sucesso em sua conclusão.

Referências

Apêndices

APÊNDICE A – Primeiro Apêndice

Texto do primeiro apêndice.

APÊNDICE B – Segundo Apêndice

Texto do segundo apêndice.

Anexos

ANEXO A – Primeiro Anexo

Texto do primeiro anexo.

ANEXO B – Segundo Anexo

Texto do segundo anexo.