Projeto Final da Disciplina de Requisitos de Software

Ferramenta de Gerência de Requisitos

Requisitos de Software

Rafael Fazzolino Pinto Barbosa - 11/0136942
Thiago Ramires Kairala - 12/0042916
Eduardo Brasil Martins - 11/0115104
Bruno Contessotto Bragança - 09/0107853

Brasília, DF - 2014

Histórico de Alterações

Sumário

1 Introdução

O desenvolvimento de software perpassa inúmeras fases até que o mesmo seja concluído e entregue ao cliente, uma dessas fases, e provavelmente a mais importante é a fase em que devemos entender o problema do usuário, compreender exatamente o que o mesmo necessita e apresentá-lo uma solução. Nesta fase, negociações serão feitas, tanto sobre funcionalidades do sistema quanto custo do projeto, tempo para conclusão do mesmo e restrições de projeto.

O resultado desta fase é uma documentação robusta, principalmente ao utilizar metodologias tradicionais de desenvolvimento. Nesta documentação se encontram as funcionalidades do software, as características do mesmo e as restrições de projeto, podendo abranger todo o software ou apenas uma primeira etapa de desenvolvimento, como é feito em metodologias ágeis, estes ítens são conhecidos como requisitos.

Com esta documentação em mãos, os desenvolvedores podem começar o desenvolvimento do sistema, porém muitos problemas surgem tanto na construção da documentação quanto na utilização da mesma para o desenvolvimento do software. A gerência, organização, classificação e rastreabilidade dos requisitos geram inúmeros problemas, já que o conteúdo é muito extenso para ser organizado facilmente de forma adequada. A partir deste problema, surge a necessidade da utilização de ferramentas que auxiliem na organização, classificação e rastreabilidade dos requisitos.

1 .1 Propósito

Ao analisar este documento, todos os *stakeholders* deverão compreender todo o contexto de negócio, os objetivos e escopo do projeto, assim como, entender o problema que deverá ser resolvido, quais necessidades do cliente deverão ser analisadas e quais serão as funcionalidades do sistema, assim como todas as suas características.

1 .2 Escopo

Este documento abrange todo o contexto do desenvolvimento de software voltado para a fase de requisitos, desde a elicitação à gerência de requisitos. Encontra-se neste documento, o problema de negócio do cliente, suas reais necessidades, as características das mesmas e todas as funcionalidades do sistema.

Dessa forma, a partir deste documento, pode-se obter conhecimento total sobre o projeto de desenvolvimento da ferramenta de gerência de requisitos, desde a metodologia utilizada para o desenvolvimento até a forma de implementação do sistema.

1 .3 Definições, acrônimos e abreviações

- *Stakeholders*:

  Todos as partes envolvidas no contexto do sistema, desde o cliente e seus funcionarios até a equipe de desenvolvimento do sistema. Todos os interessados na solução de software são considerados stakeholders do sistema.

- *Requisitos*:

  Engloba tudo que o software deve possuir para solucionar o problema em questão, desde funcionalidades do sistema até características que o software deve possuir.

- *Requisitos Funcionais*:

  São chamados de requisitos funcionais todos aqueles que apresentam as funcionalidades do sistema. [Sommerville et al. 2003].

- *Requisitos não Funcionais*:

  São chamados requisitos não funcionais todos aqueles que apresentam as características do sistema, incluindo compatibilidade, o tempo de resposta ou qualquer outra exigência que não inclua funcionalidades. [Sommerville et al. 2003].

- *Engenharia de Requisitos*:

  Engenharia de Requisitos é um conceito que engloba todo um contexto de desenvolvimento de software que envolve elicitação de requisitos, negociação, verificação e validação, e documentação e gerência de requisitos para o desenvolvimento de um sistema computacional. O uso da palavra *Engenharia* garante que técnicas sistematicas serão utilizadas para que os requisitos sejam completos, corretos e consistentes [Espindola, Majdenbaum e Audy 2004].

- *Fishbone*:

  Consiste em uma técnica utilizada para o reconhecimento do problema macro do cliente. A utilização desta técnica garante uma facilidade maior para entender o problema de negócio.

- *Framework do problema*:

  Consiste em uma técnica para organizar e auxiliar o entendimento do problema, apresentar os stakeholders afetados pelo problema, o impacto que o problema gera para o cliente e uma possivel soluçao bem sucedida. A utilização do framework garante maior facilidade no entendimento do contexto do cliente.

- *Framework de Necessidades*:

  Consiste em uma técnica para oganizar uma tabela identificando Necessidade, Problema, Solução atual e Solução Proposta. A utilização do framework de necessidade garante um melhor entendimento da necessidade do cliente.

- *WorkShop*

  Workshop é uma tecnica no qual os partipantes discutem um problema em comum onde são aplicadas tecnicas que ajudam em uma melhor identificação das necessidades do cliente e ajudam a melhorar o rendimento das reuniões.

- *Brainstorming*

  Brainstorming é uma tecnica que consiste em uma dinamica de grupo para recolher ideias a respeito de um determinado assunto e para a resolução de problemas.

- *Casos de Uso*:

  Caso de uso define uma sequência de ações que produz um resultado de valor observável. Os casos de uso fornecem estrutura para expressar requisitos funcionais no contexto dos processos de negócio e de sistema.

- *Sprint*:

  Representa o espaço de tempo no qual deverão ser realizadas atividades previamente estabelecidas para a resolução de um problema. [Beck 2000].

- *Release*:

  São entregas de código funcional, as quais são feitas por etapa, entregando pequenas partes do software de tempos em tempos. [Beck 2000].

- *Product Owner (PO)*:

  É o responsável pela atividade de repassar o conhecimento de todo o contexto de negócio para a equipe de desenvolvimento. Muitas vezes, o PO pode ser o próprio cliente ou qualquer funcionário que tenha conhecimento do problema e faz o intermédio entre a equipe de desenvolvimento e o cliente. [Beck 2000]

- *Product Backlog*:

  Representa a produção do trabalho executado durante o desenvolvimento.[Sanches, Luiz et al. 2010].

- *Sprint Backlog*:

  Representa o trabalho a ser desenvolvido durante uma *sprint* com o objetivo de criar um produto apresentável para a equipe. O *backlog* da *sprint* deve ser produzido de forma incremental.

1 .4 Referências

Não sei o que colocar aqui, pra que isso se já tem a bibliografia no final?

1 .5 Visão geral

Basicamente, neste documento, encontra-se todo o registro do contexto de desenvolvimento da Ferramenta de Gerência de Requisitos. O mesmo é organizado de forma a buscar o melhor entendimento a partir de qualquer *stakeholder* do projeto, desde leigos até funcionários da área.

2 Escolha da Metodologia

Cada uma das rotas possíveis para o desenvolvimento de software, Ágil e Tradicional, possuem características bem definidas, por isso em alguns casos é necessário a utilização de algumas práticas de cada uma delas para poder otimizar o processo.

Afim de chegar à melhor prática possível para o projeto a seguir foi desenvolvida uma tabela com cada uma das características de projeto, equipe e negócio, analisando cada uma delas para identificar em qual das duas rotas o projeto se encaixa melhor, e o resultado foi a tabela 1.

Itens	Características	Tradicional	Ágil	Descrição
Projeto	Entregas parciais		x	Utilização de Sprint para entregar software funcional em partes.
	Mudança de equipe de desenvolvimento	x		Documentação do projeto necessária ara apresentar para a nova equipe de desenvolvimento.
Equipe	Reuniões frequentes com o cliente		x	Cliente faz parte da equipe de desenvolvimento, com reuniões semanais
	Maior afinidade com metodologia ágil		x	A equipe prefere desenvolver em ágil
	Equipe pequena		x	A equipe conhece todo o código programação em pares
Negócio	Requisitos mutáveis		x	Provável evolução do sistema após o fim da primeira etapa de projeto
	Documentação extensiva para manter o sistema	x		Utlização de metodologia tradicional para documentação do projeto

Tabela 1. Tabela de características do projeto

O trabalho será feito em cima de uma metodologia Híbrida, onde a rastreabilidade e a definição dos requisitos será feita com base no método tradicional, contendo:

- Problemas
- Necessidades
- Características
- Casos de uso

A escolha da metodologia Tradicional nesta etapa, se originou pelo fato de que o projeto em si está sendo tratado como um projeto grande, onde uma documentação e uma análise inicial do contexto de negocio mais completa trariam maiores benefícios, como uma previsibilidade de recursos, estabilidade do processo, assim como uma alta garantia das funcionalidades do projeto.

No quesito de extensa documentação, para o desenvolvimento deste projeto é um ponto forte devido ao fato de que provavelmente a equipe atual, com o tempo, será substituída por uma nova.

Porém, como a equipe de desenvolvimento é pequena, possui um conhecimento maior em desenvolvimento ágil e o PO possui bastante tempo disponível para reuniões e desenvolvimento junto da equipe, o desenvolvimento do projeto e o nível de aproximação cliente/equipe seguirão a metodologia ágil, buscando maior garantia de qualidade de desenvolvimento.

Outro ponto importante a ser frizado é a possibilidade de que os requisitos mudem após a primeira etapa de desenvolvimento, já que o sistema que será implementado durante a disciplina de Requisitos de Software abrangerá apenas uma pequena parte do sistema completo.

Para o desenvolvimento do sistema, alguns valores da metodologia ágil serão utilizados. Tais como:

- Responder a mudanças

- Colaboração com cliente
- Entrega contínua de software
- Em intervalor regulares, refletir em como ficar mais efetivo
- Desenvolvimento dividido em Sprints de 15 dias cada
- Desenvolvimento em pares para nivelar o conhecimento da equipe

A definição destas atividades foi devida à possibilidade de frequentes reuniões entre a equipe e o cliente. Para este projeto estamos seguindo padrão definido por [Cho 2009] colocado a seguir:

- Através dos indivíduos e interações, definir um processo para o decorrer do projeto
- Através de documentação, ter software funcional que agregue valor ao cliente
- Colaboração do cliente antes de negociações e contratos
- Através de um plano, responder as mudanças até a chegada ao destino

3 Documento de visão

O documento de visão tem como objetivo definir uma visão geral do projeto, apresentar os problemas, os requisitos funcionais, não funcionanis, atores entre outras informações que serão definidos com o cliente a fim de garantir que a equipe de desenvolvimento e o cliente estejam na maior sincronia possível [IBM 2014].

3 .1 Posicionando

3 .1.1 Oportunidade de Negócios

Atualmente as ferramentas no mercado possuem limitações, como de qualidade, falta de flexibilidade na gerência, ou até mesmo o fechamento do código, que pode ser considerado uma limitação devida a redução de mão de obra para manutenção e evolução.

3 .1.2 Instrução do Problema

A Engenharia de Requisitos possui diversas *rotas* possíveis para se seguir, como por exemplo a rota ágil, tradicional ou até mesmo uma mistura das duas.

Infelizmente cada ferramenta de gerência de requisitos é voltada para uma dessas possibilidades, tornando díficil a tarefa voltada para outras, gerando assim nos engenheiros de requisitos a necessidade de aprender a utilizar diversas ferramentas para poder organizar projetos com *rotas* diferentes.

A utilização de apenas uma ferramenta que abrangesse as duas metodologias e ainda uma mistura das duas resolveria todo problema de gerência de requisitos em projetos que não se adequam a uma metodologia específica perfeitamente.

3 .1.3 Instrução de Posição do Produto

Para os engenheiros de Engenharia de Requisitos, a Nome da ferramentarepresentará um avanço nas atividades de gerenciamento, pois os mesmos apenas precisarão aprender as funcionalidades de uma ferramenta, simplificando a mudança entre projetos que tomam rotas distintas.

3 .2 Descrições da Parte Interessada e do Usuário

Nesta sessão serão identificados e detalhados os interessados e usuários da Nome da ferramenta.

3 .2.1 Resumo da Parte Interessada

- **Nome**: Engenheiros de software ou qualquer interessado no desenvolvimento de software.
- **Representa**: Todos os desenvolvedores de software do mundo que precisam gerenciar requisitos de forma flexivel e prática.
- **Função**: Função crítica, já que os mesmos serão os clientes diretos da aplicação, deve-se focar na qualidade da ferramenta para garantir boa aceitação dos mesmos.

3 .2.2 Resumo do Usuário

Acho que não se adequa ao documento, voto em retirar este item daqui. Já que nosso usuário é genérico, e por isso, não tem como diferenciá-los.

3 .2.3 Ambiente do Usuário

Acho que este item também não se adequa ao documento, pensem aí e me digam o que acham... Por mim eu tiro. Da mesma forma que muitos aí pra baixo... Pensemos até amanhã.

3 .2.4 Perfis das Partes Interessadas

3 .2.5 Perfis do Usuário

3 .2.6 Principais Problemas e Necessidades da Parte Interessada

O problema a ser atacado pela solução de software deve estar bastante claro entre todos os *stakeholders* para que o desenvolvimento da ferramenta não passe por nenhuma dificuldade quanto ao entendimento de onde focar para realizar a solução do problema.

Para o entendimento do problema principal, foi utilizada a técnica de *fishbone*. Segue o resultado da mesma na figura abaixo:

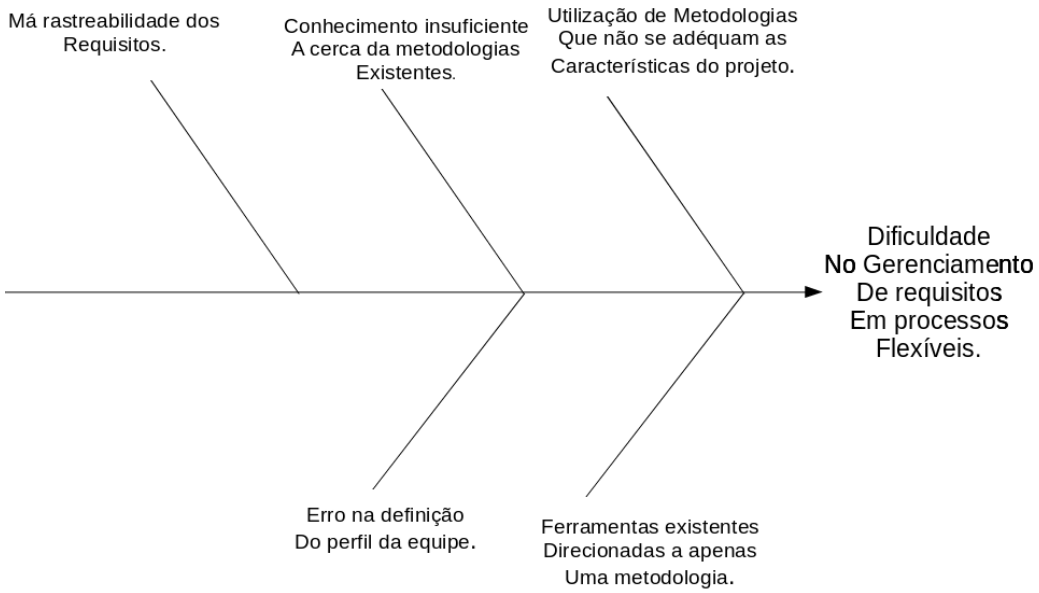

Figura 1. Fishbone de Problema

Para melhor entendimento do problema, utilizamos a técnica de *Framework de Problema*, na qual apresentamos o problema, os afetados, o impacto do problema e qual seria uma solução bem sucedida. Segue o *framework de Problema*:

O Problema:	Dificuldade no Gerenciamento de requisitos em processos flexíveis.
Afeta:	Todos os desenvolvedores de software que necessitam de uma flexibilidade maior na gerência de requisitos.
Cujo impacto é:	Software mal desenvolvidos, proporcionando maior possibilidade de erros.
Uma solução bem sucedida seria:	Utilização de uma ferramenta que faça gerência de requisitos de forma flexivel, podendo utilizá-la em qualquer metodologia.

Tabela 2. Framework de Problema

Alunos

- Rafael Fazzolino Pinto Barbosa

- Thiago Ramires Kairala
- Eduardo Brasil Martins
- Bruno Contessotto Bragança

Professores

- Mr. George Marscicano

6 Anexos

Referências Bibliográficas

[Beck 2000]BECK, K. Extreme programming explained: embrace change. [S.l.]: Addison-Wesley Professional, 2000.

[Cho 2009]CHO, J. A hybrid software development method for large-scale projects: Rational unified process with scrum. Journal of Issues in Information Systems, v. 5, n. 2, p. 340–348, 2009.

[Espindola, Majdenbaum e Audy 2004]ESPINDOLA, R. S. de; MAJDENBAUM, A.; AUDY, J. L. N. Uma análise crítica dos desafios para engenharia de requisitos em manutenção de software. In: WER. [S.l.: s.n.], 2004. p. 226–238.

[IBM 2014]IBM. Rational Unified Process. out. 2014. Disponível em: <http://pic.dhe.ibm.com/infocenter/rpcmpose/v2r0/inc

[Sanches, Luiz et al. 2010]SANCHES, F.; LUIZ, M. et al. Aplicação das abordagens scrum e xp em um processo de software. Sistemas de Informação & Gestão de Tecnologia., n. 3, 2010.

[Sommerville et al. 2003]SOMMERVILLE, I. et al. Engenharia de software. [S.l.]: Addison Wesley, 2003.