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 - 11/09*****

Brasília, DF - 2014

Histórico de Alterações

Sigla	Significado
V	`Versão`
MF	Número de `arquivos modificados`.
AL	Número de `linhas adicionadas`.
DL	Número de `linhas deletadas`.

V	Autor	Data	Mensagem do Commit	MF	AL	DL
0	Rafael Fazzolino	2014-10-17	Criando a estrutura do documento	40	3598	0
1	Rafael Fazzolino	2014-10-17	Criação do contexto de negócio	5	35	31
2	RafaelFazzolino	2014-10-17	Atualizando detalhes da estrutura de documentação.	1	2	40
3	Thiagokairala	2014-10-21	Inseridos objetivos gerais e específicos e justificativa	10	174	139

Sumário

1 Introdução

1 .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.

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.

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.

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. [5].

- *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. [5].

- *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 [3].

- *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. [1].

- *Release*:

  São entregas de código funcional, as quais são feitas por etapa, entregando pequenas partes do software de tempos em tempos. [1].

- *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. [1]

- *Product Backlog*:

  Representa a produção do trabalho executado durante o desenvolvimento.[4].

- *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.

Referências

Visão geral

2 Metodologia utilizada

3 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 2.

Itens	Características	Tradicional	Ágil	Descrição
Projeto	Entregas parciais		x	Utilização de Sprint para entregar software funcional em partes.
	Mudança de equipe de newline 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 2. 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 [2] 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

4 Documento de visão

4 .1 Posicionando

Oportunidade de Negócios

Instrução do Problema

Instrução de Posição do Produto

4 .2 Descrições da Parte Interessada e do Usuário

Resumo da Parte Interessada

Resumo do Usuário

Ambiente do Usuário

Perfis das Partes Interessadas

Perfis do Usuário

Principais Necessidades da Parte Interessada ou do Usuário

Alternativas e Concorrência

4 .3 Visão Geral do Produto

Perspectiva do Produto

Resumo das Capacidades

Suposições e Dependências

Licenciamento e Instalação

4 .4 Recursos do Produto

Recurso 1

Recurso 2

4 .5 Restrições

4 .6 Faixas de Qualidade

4 .7 Precedência e Prioridade

4 .8 Outros Requisitos do Produto

Padrões Aplicáveis

Requisitos do Sistema

Requisitos de Desempenho

Requisitos Ambientais

4 .9 Requisitos de Documentação

Notas sobre a liberação, arquivo Leia-me

Ajuda On-line

Guias de Instalação

4 .10 Atributos do Recurso

Status

Prioridade

Arquitetura

5 Documento de casos de uso

5 .1 Objetivo

5 .2 Identificação dos atores

Ator-01 Visitante

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

Professores

- Mr. George Marscicano

Referências Bibliográficas

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

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

[3] R. S. de Espindola, A. Majdenbaum, and J. L. N. Audy. Uma análise crítica dos desafios para engenharia de requisitos em manutenção de software. In WER, pages 226–238, 2004.

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

[5] I. Sommerville, S. S. S. Melnikoff, R. Arakaki, and E. de Andrade Barbosa. Engenharia de software, volume 6. Addison Wesley, 2003.