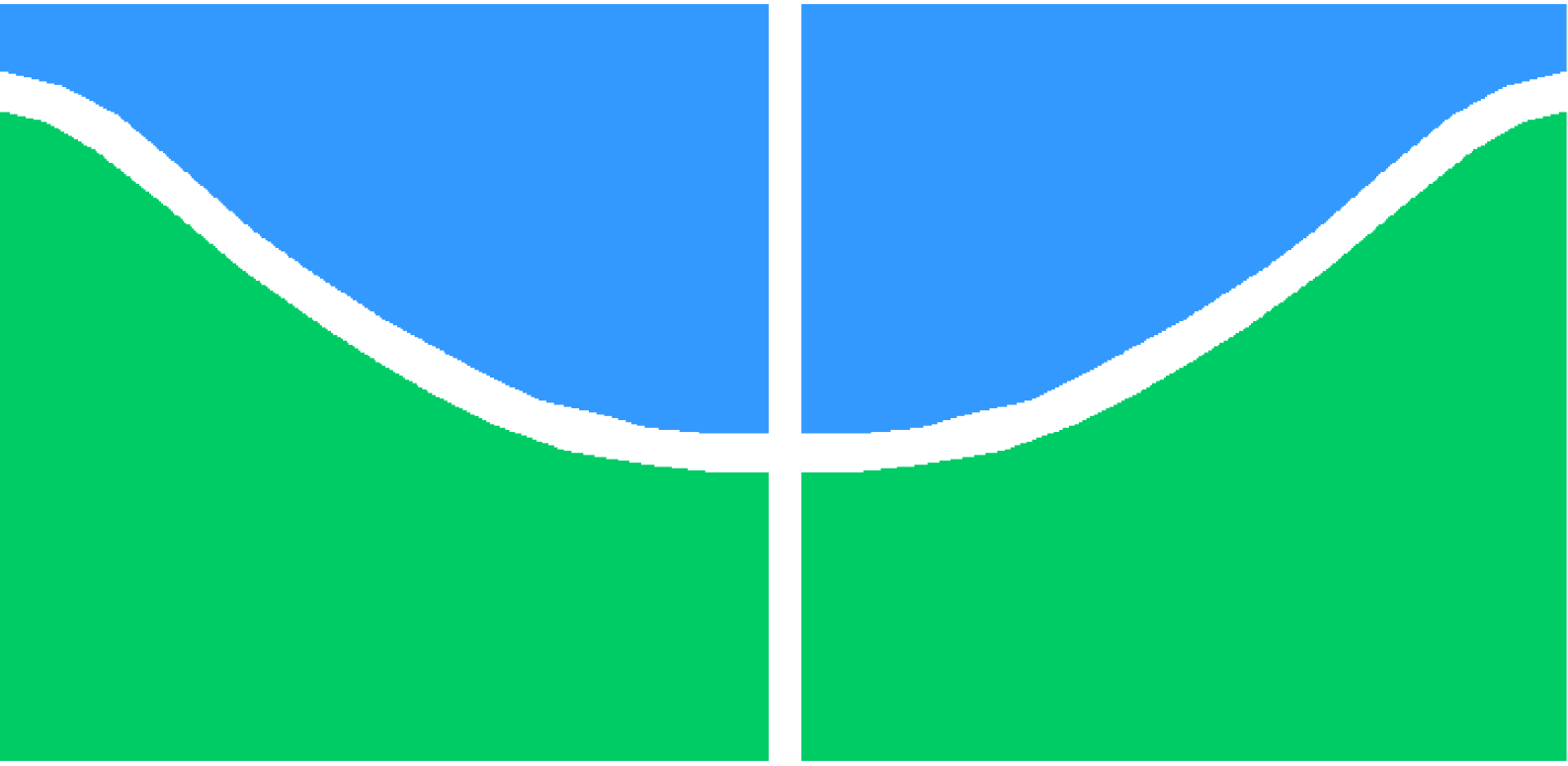

Universidade de Brasília - UnB
Faculdade UnB Gama - FGA
Requisitos de Software - 201308

Relatório de Projeto

Grupo 01
Autores: Dylan, Eduardo, Wilton, Pedro
Orientador: George Marsicano Corrêa, MSc

Brasília, DF
2015

Dylan, Eduardo, Wilton, Pedro

Relatório de Projeto

Relatório referente à disciplina de Requisitos de Software, do curso de Engenharia de Software da Universidade de Brasília.

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: George Marsicano Corrêa, MSc

Brasília, DF

2015

Dylan, Eduardo, Wilton, Pedro
Relatório de Projeto/ Dylan, Eduardo, Wilton, Pedro. – Brasília, DF, 2015-
Orientador: George Marsicano Corrêa, MSc

Relatório – Universidade de Brasília - UnB
Faculdade UnB Gama - FGA , 2015.

I. George Marsicano Corrêa, MSc. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Relatório de Projeto

Brasília, DF
2015

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

UP	Unified Process
RUP	Rational Unified Process
SAFe	Scaled Agile Framework
ER	Engenharia de Requisitos
PO	Product Owner
PM	Product Manager

Sumário

1 Introdução

Este documento apresenta considerações gerais e preliminares relacionadas à redação de relatórios de Projeto de Graduação da Faculdade UnB Gama (FGA). São abordados os diferentes aspectos sobre a estrutura do trabalho, uso de programas de auxilio a edição, tiragem de cópias, encadernação, etc.

1.1 Visão Geral do Relatório

1.2 Seções do Relatório

2 Contexto

Contexto do Ministério das Comunicações

3 Escolha da Abordagem

No desenvolvimento de um software é imprescindível a utilização de alguma abordagem que conduza seu processo produtivo.

Uma abordagem se refere a uma metodologia que será usada para estruturar, planejar, e controlar o processo de desenvolvimento do sistema (CMS, 2005).

Logo, neste capítulo será justificada a escolha de uma determinada abordagem feita pelo grupo 1 da disciplina Requisitos de Software, responsável por realizar a manutenção de sistemas de software mantidos pelo Ministério das Comunicações. Primeiro será exposto uma visão geral de cada abordagem, explicando suas diferenças e os pontos positivos e negativos presentes em cada uma delas, para que enfim uma justificativa seja apresentada. As abordagens escolhidas para representar cada uma das correntes, tradicional e adaptativa, são o Rational Unified Process (RUP) e o Scaled Agile Framework (SAFe), respectivamente.

3.1 Abordagem Tradicional - RUP

O Processo Unificado surgiu em meados dos anos 90 e tem como proposta reunir diversas práticas e filosofias que se provaram eficientes até então no contexto de desenvolvimento de software (KRUCHTEN, 2003a).

Valoriza conceitos como arquitetura de software, desenvolvimento iterativo, gerência de requisitos, construção de modelos visuais para descrever o sistema, entre outros (KRUCHTEN, 2003a). A abordagem é baseada em disciplinas, onde cada disciplina agrupa uma série de atividades relacionadas e cada atividade produz diversos artefatos, que por sua vez contêm informações do processo e são submetidos à controle de versão (são produzidos, modificados e evoluídos durante o ciclo de vida do projeto). As disciplinas representam áreas de interesse presentes no processo de desenvolvimento de software (KRUCHTEN, 2003a).

O RUP estabelece a divisão do processo em quatro fases, sendo que cada fase possui um marco (objetivo principal) que será almejado durante sua execução. As fases por sua vez são divididas em iterações que consistem em um conjunto de atividades a serem realizadas para se obter um incremento do produto. Argumenta que o planejamento a longo prazo do projeto fornece uma visão à equipe de desenvolvimento do que deve ser feito e quando deve ser feito para que o projeto seja concluído e entregue no prazo. Analisando o processo unificado sob a perspectiva da disciplina de requisitos, é possível observar que o principal recurso para se representar requisitos dentro do processo são os casos de uso

(KRUCHTEN, 2003a). Um caso de uso descreve uma interação entre uma entidade e o sistema e demonstra uma capacidade do software, assim como as funcionalidades que serão oferecidas ao usuário (KRUCHTEN, 2003b). Para atingir os objetivos da ER no RUP, a disciplina de requisitos descreve como definir uma visão do sistema, traduzir essa visão em um modelo de caso de uso (que, com especificações suplementares, definirá os requisitos detalhados do software), e como usar os atributos dos requisitos para ajudar a gerenciar o escopo do projeto e como mudar os requisitos do sistema (KRUCHTEN, 2003c).

Por fim, o RUP é um framework implementado por grandes empresas, mas determina que sua customização é aceita para que ele se adeque à realidade de organizações de menor porte, sendo possível definir as atividades e práticas do RUP que sejam do interesse da empresa em questão, desde que não sejam feridos os seus principais valores. As grandes empresas que o utilizam não o utilizam sempre da mesma maneira: enquanto algumas o utilizam de maneira formal, outras o customizam completamente (KRUCHTEN, 2003d).

3.2 Abordagem Adaptativa - SAFe

Bem mais novo, o Scaled Agile Framework (SAFe) traz ideias novas, mas tambem aspectos já consolidados na área de metodologias de software (Scrum, Kanban, etc).

Adaptável em diversos pontos, traz a ideia de dividir o projeto em três níveis: Portfolio, Program e Team. Cada nível produz seus artefatos, dispõe de diferentes papeis, tem responsabilidades, e interagem de maneira diferente (LEFFINGWELL, 2010a).

Uma das características mais importantes em relação ao UP, é que, enquanto o UP é dirigido a UC, possuindo assim somente ele como descritor de comportamentos desejáveis para o sistema, no SAFe, existem três níveis de abstração para descrever tais comportamentos: o Epic, a Feature e a User Story (LEFFINGWELL, 2010b). O Epic é bem mais alto nível, a Feature mais baixo nível que o Epic, e a User Story mais baixo nível que ambos (LEFFINGWELL, 2010b).

3.3 Escolha da Abordagem e Justificativa

Levando em consideração os fatos levantados, é evidente que ambas as abordagens apresentam o que há de melhor disponível na escolha de Frameworks/Abordagens de produção de Software. O grupo então optou pela abordagem ágil utilizando o SAFe.

Os principais motivos foram o fato de ser uma abordagem bem mais nova (trazendo assim a oportunidade de se ter um contato inicial com algo completamente novo), e que ter mais níveis de projeto (Portfolio, Program e Team) e de descritores de comportamento

(Epic, Feature e Story) parece ser mais interessante do que como essas coisas são lidadas no UP.

4 Engenharia de Requisitos

4.1 Scaled Agile Framework

4.1.1 Nível de Portfólio

Responsável pelos artefatos *epics* e *investiment themes* (LEFFINGWELL, 2010c), este nível de projeto traz os requisitos com maior nível de abstração entre os três níveis. Além disso, contém um tipo de time (*portfolio management team*), tem seu próprio backlog (*portfolio backlog*), e apresenta os conceitos de *portfolio vision* e de *architectural runway* (LEFFINGWELL, 2010c).

4.1.2 Nível de Programa

Esse nível tem como principais objetivos manter a visão (Documento de Visão), gerenciar a *release*, gerenciar a qualidade (integrando os resultados obtidos pelos times, e garantindo que os padrões de qualidade, performance, entre outros, estão sendo assegurados), fazer o *deploy* do sistema, gerenciar recursos (ajustando prazo e gastos), e eliminar possíveis impedimentos (serão os facilitadores) (LEFFINGWELL, 2010d).

4.1.3 Nível de Time

A unidade básica de trabalho para este nível é a *estória de usuário*. O objetivo deste nível é definir, construir e testar as estórias de usuário (com base nos *critérios de aceitação*) no escopo da iteração, afim de se concluir mais partes do produto final (LEFFINGWELL, 2010e).

4.2 Papéis

Um papel define comportamentos e responsabilidades de um indivíduo ou de um grupo de indivíduos que trabalham juntos como um time (KRUCHTEN, 2003e). O corportamento é expresso em termos de atividades que o papel pratica, e, cada papel é associado a um conjunto de atividades. No SAFe existem diferentes papéis para os diferentes níveis do sistema. No Nível de Portfólio vão haver, principalmente, papéis que irão interagir com os épicos e temas de investimento, no Nível de Programa, papéis que irão interagir com as *features*, e, no Nível de Time, papéis que irão interagir com estórias de usuário.

Nesta seção será dada uma breve definição de cada papel, e quem representa este papel no contexto do grupo 1.

4.2.1 Papéis no Nível de Portfólio

4.2.1.1 Epic Owner

Responsável por definir e analisar o trabalho que será seguido (LEFFINGWELL, 2010f).

4.2.1.2 Enterprise Architect

4.2.2 Papéis no Nível de Programa

4.2.2.1 Product Manager

Responsável por entender as necessidades do cliente, documentar, priorizar e validar requisitos (no nível de *feature*), gerenciar mudanças, entre outras coisas (LEFFINGWELL, 2010g). No contexto do grupo 1, o Gerente do Produto (*product manager*, PM) seriam os alunos de MPR.

4.2.2.2 Release Management

Consistindo do Product Manager e do Release Train Engineer, o papel de Release Management é responsavel por comunicar o *status* da release aos *stakeholders*, coordenar com a gerência do produto e gerência de marketing as comunicações internas e externas, validar a qualidade relevante do produto de acordo com critérios, prover a autorização final da *release*, ajudar a adptar e inspecionar o processo de *release*, entre outras coisas (SCALED AGILE INC., 2015). No contexto do grupo 1, o Release Management será feito por quem atua no papel de Product Manager, no caso, os alunos de MPR.

4.2.2.3 System Team

4.2.2.4 Business Owners

4.2.2.5 RTE

4.2.2.6 System Architect

4.2.2.7 UX

4.2.2.8 Shared Resources

4.2.3 Papéis no Nível de Time

Neste nível somente aparecerão integrantes da parte de Requisitos. Os papeis interagem principalmente com as estórias de usuário, e não serão completamente fixos durante o projeto.

4.2.3.1 Product Owner

Responsável por representar o interesse de todos os envolvidos no projeto, faz parte do time e esta junto no dia-a-dia dos desenvolvedores e testadores, elaborando as estórias de usuário e ajudando o time a alcançar seus objetivos (LEFFINGWELL, 2010h). No contexto do grupo 1, o *product owner* irá flutuar durante o projeto (não será um papel estático).

4.2.3.2 Scrum Master

É responsável por facilitar o progresso do time (ajudando assim a se alcançar o objetivo final), liderar os esforços do time, reforçar as regras do processo ágil e eliminar impedimentos (LEFFINGWELL, 2010i). No contexto do grupo 1, o papel de *Scrum Master* irá flutuar durante o projeto, e provavelmente será escolhido um integrante do grupo de requisitos para ser *Scrum Master* a cada sprint.

4.2.3.3 Desenvolvedores e Testadores

Responsáveis por desenvolverem e testarem as estórias de usuário. No contexto do grupo 1, todos os integrantes de Requisitos farão parte desse papel, sendo assim responsáveis pela elaboração da solução final.

5 Técnicas de Elicitação de Requisitos

Descobrir requisitos é um dos grandes desafios do processo de produção de Software. Não ser capaz de completar essa tarefa da maneira certa e com qualidade impossibilitará o projeto de obter sucesso, não importa quão bom sejam as qualidades do time (LEFFINGWELL, 2010c).

Por tal motivo, existe a necessidade de utilização de técnicas que ajudarão a levantar requisitos. Cada técnica apresenta melhor resultado em determinados contextos, e, muitas vezes, elas se completam. Por tal motivo, as técnicas foram escolhidas baseadas nas características do time.

As características que mais influenciaram na escolha das técnicas foram:

- Disponibilidade: técnicas que exigem grande quantidade de encontros são mais difíceis de serem aplicadas pois os horários em que os times poderiam aplicá-las não é suficiente, embora seja flexível.
- Experiência: certas técnicas exigem um certo nível de experiência para apresentar resultados satisfatórios. O grupo não tem experiência com as técnicas ou as metodologias, logo, opta-se por técnicas mais fáceis.

5.1 Workshop de Requisitos

5.2 Brainstorming

5.3 Entrevistas e Questionários

5.4 Mock-Ups

5.5 Product Council

5.6 Análise Competitiva

5.7 Customer change request system

5.8 Modelagem caso-de-uso

6 Tópicos de Gerenciamento de Requisitos

6.1 Rastreabilidade de Requisitos

6.2 Atributos de Requisitos

7 Ferramentas de Gerência de Requisitos

7.1 Ferramenta 1

7.2 Ferramenta 2

7.3 Ferramenta 3

7.4 Comparações

7.5 Definição

Referências

CENTERS FOR MEDICARE & MEDICAID SERVICES. *Selecting a Development Approach*. [S.l.], 2005. Disponível em: <http://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/XLC/Downloads/SelectingDevelopmentApproach.pdf>. Citado na página 17.

KRUCHTEN, P. *The Rational Unified Process: An Introduction*. [S.l.], 2003. 45 p. Citado na página 17.

KRUCHTEN, P. *The Rational Unified Process: An Introduction*. [S.l.], 2003. v. 3, 124-125 p. Citado na página 18.

KRUCHTEN, P. *The Rational Unified Process: An Introduction*. [S.l.], 2003. v. 3, 182-183 p. Citado na página 18.

KRUCHTEN, P. *The Rational Unified Process: An Introduction*. [S.l.], 2003. v. 3, 57 p. Citado na página 18.

KRUCHTEN, P. *The Rational Unified Process: An Introduction*. [S.l.], 2003. 61-65 p. Citado na página 19.

LEFFINGWELL, D. *Agile Software Requirements*. [S.l.], 2010. 31-33 p. Citado na página 18.

LEFFINGWELL, D. *Agile Software Requirements*. [S.l.], 2010. 454-455 p. Citado na página 18.

LEFFINGWELL, D. *Agile Software Requirements*. [S.l.], 2010. 227-228 p. Citado 2 vezes nas páginas 19 e 23.

LEFFINGWELL, D. *Agile Software Requirements*. [S.l.], 2010. 63-64 p. Citado na página 19.

LEFFINGWELL, D. *Agile Software Requirements*. [S.l.], 2010. 47-48 p. Citado na página 19.

LEFFINGWELL, D. *Agile Software Requirements*. [S.l.], 2010. 418-419 p. Citado na página 20.

LEFFINGWELL, D. *Agile Software Requirements*. [S.l.], 2010. 283-287 p. Citado na página 20.

LEFFINGWELL, D. *Agile Software Requirements*. [S.l.], 2010. 201-203 p. Citado na página 21.

LEFFINGWELL, D. *Agile Software Requirements*. [S.l.], 2010. 51-52 p. Citado na página 21.

SCALED AGILE INC. *Release Management*. [S.l.], 2015. Disponível em: <http://www.scaledagileframework.com/release-management/>. Citado na página 20.

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.