KAREN ALINE ALVES BEZERRA

PROPOSTA DE SOLUÇÃO DE FALHAS DE COMUNICAÇÃO UTILIZANDO DOCUMENTAÇÃO NORMATIZADA

LAVRAS – MG

2012

KAREN ALINE ALVES BEZERRA

PROPOSTA DE SOLUÇÃO DE FALHAS DE COMUNICAÇÃO UTILIZANDO DOCUMENTAÇÃO NORMATIZADA

Monografia de Graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso de Bacharelado em Sistemas de Informação para obtenção do título de Bacharel em Sistemas de Informação.

Orientador
Prof. Eric Fernandes de Mello Araújo

Co-Orientador
Jaime Daniel Corrêa Mendes

LAVRAS – MG

2012

KAREN ALINE ALVES BEZERRA

PROPOSTA DE SOLUÇÃO DE FALHAS DE COMUNICAÇÃO UTILIZANDO DOCUMENTAÇÃO NORMATIZADA

Monografia de graduação apresentada ao Colegiado do Curso de Sistemas de Informação, para obtenção do título de Bacharel em Sistemas de Informação.

APROVADA em 29 de novembro de 2012.

CRISTIANE SANTOS FREIRE

PAULO HENRIQUE DE LIMA SIQUEIRA

ERIC FERNANDES DE MELLO ARAÚJO (orientador/a)

JAIME DANIEL CORRÊA MENDES (Co-Orientador)

LAVRAS-MG

2012

Dedico esta monografia aos meus pais...

AGRADECIMENTOS

Àquele a quem agradeço todos os dias, e a todos que contribuiram para a realização deste trabalho, por meio de:

Ensino, agregando não somente o conhecimento, mas ensinando a arte de usá-lo. Orientação, que ocorreu de forma exigente, paciente, atenciosa e rápida, sem ela a conclusão deste trabalho seria inviável. Correções, feitas de forma crítica, contribuíram para o aumento da qualidade no trabalho. Equipe desenvolvedora no Análise Ambiental, principalmente pelo auxílio prestado. Sugestões nas entrevistas, estas permitiram o levantamento das necessidades da proposta deste trabalho. Realização de testes, possibilitou avaliar se a proposta deste trabalho apresentou contribuição para o desenvolvimento de *software*. Auxílio na construção de artefatos, pelo tempo dedicado apenas em prol da amizade. Oportunidade de produzir, permitindo-me transformar trabalho em produção. Interesse, estes ajudaram a manter o foco e contribuíram para o sentimento de importância em relação ao trabalho. Força, acreditem, sem ela, talvez estivesse vendendo coco na praia. Torcida, toda energia é materializada.

Meus sinceros agradecimentos.

"De nada vale o conhecimento sem que este possa servir ao teu próximo. "

RESUMO

No processo de desenvolvimento de *software* o teste é essencial para garantir a qualidade do produto final. Para que sua execução seja confiável, a comunicação entre analistas, desenvolvedores e clientes deve ocorrer com um mínimo ou nenhum ruído, pois de outra forma, pode ocasionar uma série de problemas ao longo do processo, prejudicando o produto final. Diante deste cenário, foi realizado um estudo de caso que analisou as falhas e dificuldades que interferem nesse processo. A interpretação desse levantamento permitiu, então, traçar uma estratégia de melhoria que centralizou as informações necessárias à execução dos testes por meio de uma documentação normatizada acessível por todas as áreas atuantes no processo.

Palavras-Chave: Comunicação; Documentação; Teste de software;

ABSTRACT

In the development process of *software* the software testing is essential to ensure the quality of the final product. For the execution of the development process be trusted, communication between analysts, developers and customers must occur without or a minimal of noise, because otherwise, it may cause a number of problems along the way, damaging the final product. Given this scenario, it were performed a study of case that analyzed the failures and difficulties that may interfere in this process. The interpretation of this study allowed devise a strategy for improvement that centralized the information necessary to perform the tests using a standardized documentation accessible from all active areas in the process.

Keywords: Comunication; Documentation; Software Testing;

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

1 INTRODUÇÃO

O processo de desenvolvimento de *software* é um conjunto de atividades que guia a produção de um produto de *software* (SOMMERVILLE, 2007). Existem diversos passos ou processos para se criar um *software* seguindo modelos de desenvolvimento descritos em uma área conhecida como engenharia de *software*. Independente do paradigma de engenharia de *software*, o processo de desenvolvimento é composto de três fases: definição, desenvolvimento e manutenção que estão presentes em todo o desenvolvimento para quaisquer áreas de aplicação, tamanho do projeto ou complexidade.

Durante o processo de desenvolvimento de *software* é essencial que haja uma comunicação clara e objetiva para que o resultado seja produzido conforme o esperado, o que exige maior atenção dos envolvidos no processo de comunicação. Existe uma dificuldade em estabelecer uma comunicação consistente, pois durante o processo de desenvolvimento é comum que em diferentes áreas desse processo, sejam utilizadas diferentes linguagens de comunicação. Para contornar essas falhas na comunicação, existe uma linguagem que pode ser adotada em todas as áreas, a linguagem de negócio, pois é uma linguagem única e baseada no modelo de negócio.

Outro fator essencial que contribui para que o resultado do *software* atenda as especificações estabelecidas é a garantia da qualidade de *software* a qual é obtida também por meio de teste de *software*. Para que os testes sejam executados corretamente são necessárias informações oriundas das diferentes áreas do desenvolvimento, o que acarreta em buscas distintas. Estas buscas, consistem em reunir informações oriundas de fontes e formatos diferentes, o que pode ocasionar em desacelaração no processo de desenvolvimento de *software*, já que essas informações são uma premissa para conduzir e executar os testes.

O Laboratório de Estudos e Projetos em Manejo Florestal (LEMAF), situado no Departamento de Ciências Florestais (DCF) da Universidade Federal de Lavras (UFLA) possui um processo de desenvolvimento de *software* que representa bem o cenário descrito. A comunicação existente no processo de desenvolvimento de *software* no LEMAF não é clara e objetiva o bastante, o que dificulta a produção de software, pois acarreta uma certa morosidade para executar os testes devido à dificuldade de encontrar informações para testar o sistema.

Em levantamento realizado no LEMAF, identificou-se que o fator que contribui para que o testador não saiba o porquê está executando a tarefa é a ausência de um artefato que o guie, auxiliando na compreensão da execução da tarefa. Avaliando a documentação dos sistemas já desenvolvidos, não foi encontrada uma que permeie todo o processo de testes de *software* utilizando uma linguagem clara e objetiva. Foi identificado, então que essas falhas de comunicação e a descentralização das informações constituem um problema que deve ser resolvido para reduzir os prejuízos ocorrentes no processo de desenvolvimento de *software*.

1.1 Objetivo Geral

Dada a desaceleração no processo de teste de *software* identificado pela gerência do LEMAF, o objetivo deste trabalho é identificar as falhas e dificuldades que causam esse quadro, por meio de uma solução centralizada.

1.2 Objetivos Específicos

Para identificar as falhas e dificuldades que causam a desaceleração do processo de desenvolvimento, as seguintes atividades devem ser cumpridas:

- Detecção de falhas: a partir de entrevistas com funcionários do LEMAF, foram coletadas informações referentes à dificuldades para se executar os testes.

- Agrupamento de informações e busca de um meio solucionador: a partir dos dados coletados no levantamento realizado, as informações foram analisadas e agrupadas por similaridade.

- Proposta de uma documentação: proposta de um modelo de documentação normatizado e centralizado, para guiar as atividades de teste, elaborado a partir do agrupamento das informações.

- Validação da documentação: validação do modelo proposto por meio de *feedback* dos entrevistados.

1.3 Estrutura do trabalho

O capítulo 2 apresenta um embasamento teórico necessário para a compreensão do trabalho. O capítulo 3 apresenta o método proposto para a resolução do problema. O capítulo 4 apresenta a aplicação do método proposto em um estudo de caso e os resultados embasados no método proposto juntamente com análise e discussão sobre os resultados obtidos no estudo de caso. O capítulo 5 apresenta as conclusões do trabalho.

2 REFERENCIAL TEÓRICO

Neste capítulo, são descritos os conceitos fundamentais para o completo entendimento deste trabalho. Primeiramente são apresentados conceitos relacionados a teste de *software* seguido do conceito de comunicação. No fim do capítulo estão contidos conceitos sobre documentação de *software*, juntamente com propostas relacionadas ao resultado deste trabalho.

2.1 Teste de *software*

O teste de *software* contribui para a garantia da qualidade de um *software*. Segundo TOSETTO; BELLINI (2008) o teste de *software* define atividades de verificação e validação com o objetivo de encontrar problemas. O universo do teste de *software* é um meio amplo e sua importância é altamente relevante considerando que menores serão os custos, manutenção e tempo dedicado a correção dos erros o quanto antes os erros forem detectados.

WALLACE; FUJII (1989) define verificação e validação como um conjunto de atividades de análise e teste que são realizadas durante todo o ciclo, que complementa os esforços de outras funções da qualidade de engenharia. Determinando se o *software* executa funções como desejado, garantindo que não haja a execução de funções não pretendidas, e medindo a qualidade e confiabilidade.

Verificação envolve avaliação de *software* em cada fase do ciclo de vida do desenvolvimento assegurando que os requisitos estabelecidos na fase anterior sejam cumpridos. Já a validação envolve teste de *software* ou sua especificação no final do esforço de desenvolvimento para garantir que ele atenda suas necessidades. Embora verificação e validação possuírem definições distintas pode-se obter um

maior benefício utilizando-as sinergicamente e tratando-as como uma definição integrada (WALLACE; FUJII, 1989).

Para uma melhor compreensão da atividade de teste de *software* é necessário discriminar a diferença entre os conceitos definindo erro, defeito e falha. NETO (2007) define defeito como uma ação inconsistente, ocasionada por uma tentativa de entender uma determinada informação, resolver um problema ou utilizar um método ou uma ferramenta. Erro como uma manifestação concreta de um defeito num artefato de *software*, ou seja diferença entre o valor obtido e o valor esperado. E falha como o comportamento do *software* diferente do esperado pelo usuário.

Segundo ROCHA; MALDONADO; WEBER (2009) o teste é dividido em cinco tipos ou fases:

- Teste de Unidade: explora a menor unidade do projeto, objetiva provocar falhas ocasionadas por defeitos de lógica ou implementação em cada módulo. O foco deste tipo de teste são métodos dos objetos ou pequenos trechos de código, evidenciando que a unidade testada funciona de forma independente.

- Teste de Integração: objetiva encontrar falhas provenientes da integração interna dos componentes de um sistema. As partes do *software* são integradas e executadas,verificando se a interação entre elas funciona de maneira esperada sem a presença de erros.

- Teste de Sistema: busca provocar falhas por meio da utilização do *software*, simulando o usuário final no ambiente, condições e dados. Objetiva verificar se as funcionalidades especificadas nos documentos de requisitos estão implementadas.

- Teste de Aceitação: objetiva verificar se o comportamento está de acordo com o esperado, por meio de simulação de operações de rotina de sistema, verifica se o sistema está conforme especificação.

- Teste de Regressão: Consiste em aplicar, a cada nova versão do *software* ou a cada ciclo, os testes já aplicados nas versões ou ciclos anteriores do sistema, evitando que novos defeitos sejam introduzidos.

Para uma melhor qualidade é necessário que o teste permeie todas as fases. Isso possibilitará um aumento da confiança no produto, cumprindo o papel do teste de garantir a qualidade de *software*. Dessa maneira é possível obter um *software* funcional e que atenda as especificações do cliente. Um outro quesito importante para que isso ocorra é que os casos de testes sejam elaborados de forma a cobrir os requisitos especificados. Segundo VICENTE (2010) os casos de testes devem ser planejados de acordo com diferentes objetivos, verificando se há incompatibilidade com os requisitos do usuário, ou avaliando a robustez por meio de verificações de condições de *stress*, ou ainda medindo outros atributos como performance, usabilidade, ou estimando a confiabilidade operacional, entre outras coisas.

Um caso de teste descreve uma condição particular a ser testada e é composto por valores de entrada, restrições na execução e um resultado ou comportamento esperado. Por serem baseados no comportamento esperado ao executar um caso de teste, verifica-se o o valor obtido com o valor esperado. Um estado não previsto durante a execução do caso de teste indica um erro. Este erro deve ser corrigido e o caso de teste re-executado. Este ciclo deve se repetir até que o caso de teste obtenha a saída esperada contribuindo para a qualidade do *software*, atendendo aos requisitos do teste.

NETO (2007) diz que testar um *software* significa verificar por meio de uma execução controlada se o seu comportamento ocorre de acordo com o especificado, com o objetivo principal de revelar o número máximo de falhas dispondo

do mínimo de esforço. Por isso é importante levar em consideração os métodos e maneiras de realizar o teste de *software*. Dentro da área de teste *software* é possível notar uma evolução voltada para a aplicação de testes em sistemas dinâmicos, incluindo propostas para o aumento de qualidade e produtividade de *software*.

Para BORGES (2010) a definição de teste de *software* é a investigação do *software* a fim de fornecer informações sobre sua qualidade em relação ao contexto em que ele deve operar, abordando o processo de explorar o produto para encontrar seus defeitos. Sendo um processo realizado pelo testador de *software*, que permeia outros processos da engenharia de *software*, e que envolve ações que vão do levantamento de requisitos até a execução do teste.

Um outro fator que merece atenção é o fato de que *software* tem menos tempo para ficar pronto e que as saídas da execução do teste devem ser exaustivamente analisadas. Assim os casos de teste devem verificar as condições inválidas de execução e as condições válidas. Esta verificação pode ser realizada de duas maneiras, teste manual e teste automatizado. O teste manual é realizado por uma pessoa que simula o uso do usuário final, testando se o comportamento resulta nas saídas especificadas nos casos de uso. Já o teste automatizado corresponde a um programa ou *script* executável que roda o programa a ser testado fazendo verificações automáticas. (NETO, 2007), (ROCHA; MALDONADO; WEBER, 2009) e (VICENTE, 2010)

2.2 Comunicação

A comunicação possui muitas definições distintas, contudo as definições existentes giram em torno do conceito de transmitir uma mensagem ou compartilhar uma ideia ou informação. Quanto a sua classificação SALLES (2011) afirma que

o processo de comunicação pode ser formal ou informal, e definir como este processo ocorre é determinado pelos envolvidos.

De acordo com CHIAVENATO apud SALLES (2011) o processo de comunicação é composto por seis elementos:

- Fonte: pessoa, grupo ou organização que deseja transmitir alguma ideia ou informação por meio de uma mensagem;
- Transmissor: é o meio utilizado para codificar a ideia ou significado por meio de uma mensagem;
- Canal: é o meio escolhido para a mensagem fluir entre a fonte e o destino;
- Receptor: é o meio que decodifica ou interpreta a mensagem para atribuir um significado;
- Destino: pessoa, coisa ou processo para o qual a mensagem é enviada.
- Ruído: é o termo que indica qualquer distúrbio indesejável dentro do processo de comunicação e que afeta a mensagem enviada pela fonte ao destino.

SALLES (2011) diz que cuidados específicos devem ser tomados para que o ruído não esteja presente no processo de comunicação. Esta afirmação deve ser considerada com atenção, pois segundo LANA (2008) os ruídos existentes podem ocasionar erros ou equívocos de comunicação.

MEGGINSON; C.MOSLEY; JR. (1998) definem comunicação como o processo de transferir significado de uma pessoa para outra como informação ou ideia. Para OLIVEIRA (2003), a comunicação é uma ação contínua do ser humano, sendo essencial para a existência do homem, principalmente quando os indivíduos se organizam, instituindo grupos. Já para FURLANETTO (2001), a comunicação

é a forma básica de se estabelecer contatos, criar e manter diversos tipos de grupos, sendo a responsável pela difusão de ideias, desejos e objetivos de cada ser humano.

Quanto ao meio organizacional, SOUSA (2004) afirma que o processo de comunicação organizacional tem sido cada vez mais objeto de estudos, por envolver pessoas com diferentes culturas e hábitos. De acordo com FERNANDES (1999), a comunicação é muito importante dentro de uma organização, pois é necessária ao desenvolvimento de planos, estabelecimento e disseminação de metas, organização dos recursos humanos, seleção, desenvolvimento e avaliação dos membros que fazem parte de qualquer organização.

Dentro de uma organização a maioria dos projetos são desenvolvidos em equipe. MORAES; SOUZA; OLIVEIRA (2011) afirmam que em relação a gerência e interação de pessoas em projetos a principal forma de realizá-la é por meio da comunicação. Para LANA (2008) ao se desenvolver um trabalho em equipe é importante que os diálogos sejam estabelecidos baseados em um pensamento comum, considerando que em uma organização a comunicação visa o consenso.

Segundo MORAES; SOUZA; OLIVEIRA (2011)a eficácia da comunicação utilizada durante o desenvolvimento de um projeto influencia de forma determinante na qualidade do processo executado, desta forma interferindo na qualidade do produto final gerado pelo projeto. Para que o objetivo do projeto seja alcançado PAULA (2007) diz que, a comunicação entre os profissionais durante o processo de desenvolvimento é fundamental para se criar um entendimento consistente do problema e do que deve ser construído, tentando evitar que cada profissional trabalhe baseando-se em premissas diferentes uns dos outros, além de contribuir para diminuir o retrabalho.

Destarte, temos a comunicação como elemento essencial no desenvolvimento de projetos, afirmado por MORAES; SOUZA; OLIVEIRA (2011) em: "Uma co-

municação eficaz sempre foi de importância fundamental em projetos das mais variadas naturezas e áreas. Para projetos específicos à construção de software, esta preocupação é igualmente válida."

O processo de desenvolvimento de *software* pode ser definido como um produto resultante de trabalho humano que provêm da interação entre vários envolvidos, que possuem experiências, interesses e necessidades diferentes LANA (2008). Visualizando estas diferenças SALLES (2011) diz que o entendimento entre os envolvidos é fator fundamental para o sucesso dos projetos de desenvolvimento de *software*.

Este processo é composto por áreas distintas que trabalham em conjunto. Segundo PAULA (2007) além destas áreas atuarem em uma mesma fase do processo, o trabalho realizado por cada uma delas afeta, direta ou indiretamente, o trabalho das outras. Isto implica que um problema de comunicação ocorrido, afeta o processo de desenvolvimento como um todo, não se resumindo apenas na área de origem. Desta forma, devem ser buscadas soluções que visem melhorar a comunicação de todos os envolvidos de modo que a interação entre eles proporcione um desenvolvimento satisfatório.

Outro fator que influencia na comunicação é a dificuldade de se comunicar que algumas pessoas apresentam. Segundo MORAES; SOUZA; OLIVEIRA (2011), é comum este problema afetar as outras pessoas que não têm receio de se comunicar, pois estas acabam não sendo informadas de questões ou alterações importantes do sistema. Esta dificuldade tem taxa ainda maior na computação, área em que está inserido o processo de desenvolvimento de *software*.

MORAES; SOUZA; OLIVEIRA (2011) citam um documento que guia programas de graduação, focado em aspectos humanos elaborado pela IEEE e ACM (ABERNETHY *et al.*, 2005). No documento é sugerido que os cursos de computação devem abordar atividades como: trabalho em equipe, gerenciamento, comu-

nicação . Também afirmam que o mercado já constatou a importância desta área na computação.

Quanto à melhoria do processo de comunicação MEGGINSON; C.MOSLEY; JR. (1998) ressaltam três práticas:

- Ter consciência da necessidade da comunicação eficaz;
- Criar um ambiente que estimule o *feedback*;
- Ter um ouvinte mais eficaz.

SALLES (2011) nos diz que a literatura tem demonstrado a importância da comunicação no processo de desenvolvimento de *software*, devido a muitos problemas terem origem em falhas oriundas de ruídos no processo de comunicação. Por este motivo, a necessidade de melhorias é evidente no processo de desenvolvimento de *software*, a conscientização desta necessidade é fundamental para que falhas na comunicação não interfiram no desenvolvimento.

Percebe-se a importância da comunicação no processo de desenvolvimento de *software*, sendo também reconhecida nas mais diversas áreas.

2.3 Documentação de *software*

Um documento de *software* pode ser definido como um artefato cuja finalidade é comunicar uma informação sobre o sistema de *software* ao qual ele pertence (FORWARD, 2002). Em outras palavras, documentar é comunicar. Segundo AMBLER (2004), do ponto de vista da modelagem ágil, um documento é qualquer artefato externo ao código fonte cujo propósito seja transmitir informação de uma maneira persistente. Já um modelo é uma abstração que descreve um ou mais aspectos de um problema ou de uma solução potencial para resolver um problema.

Alguns modelos podem se tornar documentos, ou incluídos como parte deles, ou simplesmente serem descartados quando cumprirem seu papel. Por fim, código fonte é uma sequência de instruções, incluindo os comentários que descrevem essas instruções, para um sistema de computador. O termo documentação inclui documentos e comentários de código fonte.

A documentação é uma atividade essencial no processo de desenvolvimento de *software*, sendo feito por meio dela o registro da evolução do *software* para que sejam criadas as bases necessárias para as etapas posteriores do processo de *software* (NUNES; SOARES; FALBO, 2004). Todavia, apesar de acompanhar o processo de *software*, nem sempre essa documentação pode ser válida ou adequada em engenharia de *software* FORWARD (2002).

FORWARD (2002) definem que o papel da documentação é comunicar e conter informação do sistema que descreve para seu público. Esta afirmação implica que a documentação deve ser capaz de atender as informações de acordo com sua finalidade, sendo compreensível a quem busca. Além do papel de contribuir para comunicação, a documentação deve ter um propósito, como sugere AMBLER (2004). Isso evidencia, que se não há um propósito específico para um determinado documento, então a necessidade desse artefato deve ser reconsiderado ou determinar uma finalidade antes de investir recursos de produção.

2.3.1 Propostas de documentação

Diferente do que propõe esse trabalho, NUNES; SOARES; FALBO (2004) afirmam que documentos são artefatos de *software* não passíveis de execução, constituídos tipicamente de declarações textuais, normalmente associados a padrões organizacionais (roteiros) que definem a forma como eles devem ser produzidos. Já neste trabalho, é proposto um modelo de documentação, que possa ser compilado, extraindo apenas as informações desejadas.

O trabalho de FORWARD (2002) concentra-se na questão da qualidade da documentação. Em particular, ao tornar a documentação eficaz para o público e proporcionar uma melhor perspectiva sobre a relevância da documentação em um projeto de *software*. Desta forma, a documentação deve ser compreensível aos colaboradores que é destinada. Ainda nesta linha, o trabalho deste autor identificou que o problema da documentação é geralmente descrito por um código incompreensível, ou por algum item que dificulte a compreensão do sistema como um todo. Ainda afirma em seu trabalho que documentos corretos e informações adequadas são difíceis para localizar e utilizar corretamente, fator também identificado neste trabalho.

FORWARD (2002) apresenta um modelo de manutenção de documentos, semelhante à metodologia utilizada neste trabalho, foi utilizada coleta de opiniões por meio de entrevistas para levantamento dos artefatos necessários a documentação. Bem como a forma como os seus atributos e fatores externos afetam na utilidade da documentação. As seções nas mesmas áreas das propostas neste trabalho, foram propostas no trabalho de FORWARD (2002), entre elas estão testes e documentos de qualidades e documentação de requisitos.

NUNES; SOARES; FALBO (2004) desenvolvem uma ferramenta de apoio, o XMLDoc. Que visa prover apoio à documentação de forma única e consistente em todo o ambiente, de modo a garantir uma apresentação integrada, usando XML. Oferecendo funcionalidades para apoiar a definição de modelos de documento, permitindo uma padronização, por meio de definição da estrutura do documento, similar ao que pode ser visto no presente trabalho, que propõe uma documentação em formato XML e busca uma padronização da linguagem.

Considerando que um *software* tem cada vez menos tempo para ficar pronto, que a comunicação é essencial para seu desenvolvimento e que os casos de testes devem definir a saída esperada de forma a reduzir a interpretação do critério de su-

cesso, a saída da execução do teste deve ser exaustivamente analisada. Alinhando o problema apresentado na seção 1 com as soluções de embasamento encontradas na literatura, esse trabalho propõe a detecção de falhas no processo de teste de *software* resultando em um meio solucionador elaborado a partir das necessidades dos colaboradores do *software*. Este documento conterá as funcionalidades, requerimentos, armazenamento, regras de negócio do *software*, auxiliando na realização das atividades de teste de *software*, contribuindo de maneira gradual para a qualidade do projeto.

3 METODOLOGIA

O tipo de pesquisa deste trabalho é classificado quanto à natureza como aplicada, pois visa levantar a causa da desaceleração no processo de teste de *software* e aplicar como solução uma documentação ao qual é avaliada a sua contribuição.

Quanto aos objetivos, a pesquisa pode ser classificada como exploratória, pois levanta as falhas do processo de teste de *software* e tenta soluciona-las buscando uma estrutura de documentação mais eficiente que aquelas já existentes para solução dos problemas de comunicação em questão.

Quanto aos procedimentos, este trabalho é classificado como estudo de caso, pois avalia se houve contribuição com o resultado. A pesquisa também é classificada como de laboratório, pois variáveis que possam vir a prejudicá-la podem ser controladas (JUNG, 2004). O trabalho foi realizado no LEMAF, localizado no DCF- UFLA.

3.1 Material

3.1.1 Formulário de entrevista para coleta de opiniões

Para a realização das entrevistas foi utilizado um formulário que contém uma breve explicação sobre a tarefa a ser executada, questionamentos sobre falhas e dificuldades existentes no processo de teste de software e um espaço para sugestões que solucionariam o problema. O formulário utilizado neste trabalho encontra-se no apêncide A desta monografia.

3.1.2 Formulário de análise de sugestões

Na coleta de opiniões para verificação se o meio solucionador atendeu as sugestões e falhas detectadas no setor de teste, foi apresentada a documentação preenchida a cada colaborador juntamente com um formulário para resposta. O formulário de análise de sugestões foi personalisado para cada colaborador. O formulário foi composto pela lista de sugestões de cada colaborador e as opções de resposta para cada sugestão: sim, não e parcialmente. Desta forma, pode-se classificar se cada sugestão foi atendida. O modelo deste formulário encontra-se no apêndice B desta monografia.

3.2 Levantamento de falhas por meio de entrevista com funcionários

A proposta foi elaborada a partir de estudos e pesquisa com os funcionários do LEMAF. O procedimento adotado para o estudo e pesquisa consistiu na detecção de falhas e dificuldades no processo de teste de *software* por meio de entrevista com funcionários. Apesar do foco da documentação ser na área de testes, optou-se por entrevistar toda a equipe envolvida no processo de desenvolvimento, almejando uma visão mais abrangente do problema.

O método entrevista foi escolhido pelas suas características de proximidade entre e entrevistador e entrevistado. A conversa sendo frente-a-frente pode ser conduzida e orientada pelo entrevistador, facilitando o relato de experiências e acontecimentos. Seguindo esta linha a entrevista sucedeu da seguinte forma:

1. Comunicado:

   Comunicados aos integrantes via e-mail. Por meio da lista de comunicação interna do LEMAF, os funcionários foram avisados sobre a coleta de

informações. A comunicação continha uma curta descrição do trabalho e solicitação de organização de ideias para facilitar a coleta de opiniões.

2. Entrevista:

   Ao iniciar a entrevista o entrevistador realizou uma breve explicação sobre o trabalho como foco no objetivo e conduzir a conversa de modo a realizar a coleta de falhas e dificuldades que desaceleram o processo de teste de *software*.

   Nesta etapa cada integrante da empresa foi entrevistado, sendo a entrevista composta de três sub etapas.

   2.1. Explicação da descrição do trabalho para o entrevistado, evidenciando o objetivo da detecção de falhas. Ao fim foi questinado ao funcionário quais são as sugestões que podem resolver o problema, visando uma contribuição da área de atuação de cada funcionário.

   2.2. Coleta de sugestões, com perguntas direcionadas a área de atuação.

   2.3. Manutenção do contato para sugestões futuras.

3. Agrupamento de sugestões:

   Os dados coletados foram agrupados por área de desenvolvimento, eliminando as sugestões duplicadas.

4 RESULTADOS

Neste capítulo serão apresentados os dados coletados, a documentação proposta, o estudo de caso o qual foi aplicada a documentação proposta e os resultados da validação obtidos a partir da aplicação da documentação.

4.1 Coleta de Dados

Os dados coletados nas entrevistas foram levantados em quatro áreas: banco de dados, desenvolvimento, gerência e qualidade de *software*. Em cada área, foi realizado um estudo das falhas identificadas juntamente com a viabilidade das sugestões de artefatos, sendo selecionados os considerados essenciais em cada área. Após a coleta identificou-se como um meio benéfico de dispor os resultados a composição de um documento padronizado e centralizado, visto que esta também foi uma carência identificada no LEMAF. O documento foi padronizado com a utilização da linguagem de negócio, e tornou-se centralizado por ser acessível aos colaboradores.

Após coleta, as informações foram agrupadas em sugestões de artefatos para a documentação, falhas detectadas no processo de teste de *software* e sugestões de meios para melhorar o processo de teste de *software*, conforme Tabelas 4.1, 4.2, 4.3, 4.4 e 4.5.

Após síntese de cada área, foram eliminadas as sugestões duplicadas, foi realizada análise de viabilidade de cada sugestão. Para solucionar os problemas ocasionados pelas falhas identificadas foi sugerida como meio solucionador a criação de uma documentação normatizada e centralizada.

Para elaborar a estrutura dessa documentação foi escolhida a linguagem de marcação de dados XML (*Extensible Markup Language*). A escolha do XML

também deu-se por ser uma linguagem metadescritiva, ou seja, mesmo na forma crua, é possível uma interpretação e comprenssão da informação do documento.

4.2 Documento

Nesta seção é apresentada a estrutura do documento proposto, o qual é apresentado em seções, que são descritas junto com suas finalidades. O documento proposto encontra-se no apêncide C desta monografia.

O documento foi proposto após a coleta de dados (Seção 4.1) como meio solucionador para atender as sugestões e falhas identificadas. Este documento foi composto de oito seções: Equipe de Desenvolvimento, Visão, Requisitos, Casos de Uso, Interfaces Gráficas, Regra de Negócio, Armazenamento e Teste.

4.2.1 Equipe de desenvolvimento

Esta seção contém os dados da equipe que está executando o serviço, inserida por conter informações relativas a quem desenvolve o *software*, representantes e responsáveis com seus contatos. No levantamento das necessidades foi sugerido que os integrantes das equipes estivessem presentes na documentação, evidenciando a importância dessa seção.

4.2.2 Visão

Esta seção corresponde à visão do sistema, deve incluir as principais funcionalidades do sistema, restrições que o sistema está passível e políticas de homologação. Ainda deve incluir potenciais usuários do sistema, caracterizando-os, informando como o sistema impacta em seu trabalho.

Sua importância está associada a uma visão sucinta do sistema, que permite o entendimento do sistema de forma eficaz. Esta contribuição ajuda que as atividades sejam elaboradas com mais segurança. Ao avaliar a contribuição desta seção na realização dos testes, pode-se afirmar que possibilitou ao testador entender o sistema sem o auxílio de outro integrante da equipe de desenvolvimento.

4.2.3 Requisitos

Esta seção contém os requisitos funcionais e não funcionais do sistema, contendo descrição detalhada das atividades. Esta seção é importante por facilitar o acesso aos requisitos do sistema, informações muitas vezes de difícil acesso e necessárias para verificação e validação do *software*.

Deve conter uma especificação não ambígua e completa dos requisitos do *software*, com o intuito de especificar o produto que será entregue ao cliente. Destarte possibilitando que o teste elaborado valide o cumprimento dos requisitos especificados.

4.2.4 Casos de uso

Esta seção contém os casos de uso do sistema, informando os atores e seus relacionamentos, descrição do caso de uso, lista de pré-condições que devem ser satisfeitas para realizar o caso de uso e lista de possíveis estados resultantes do caso de uso, registrando e descrevendo os requisitos do sistema.

Os testadores utilizam os casos de uso para compreender a sequencia de ações das funcionalidades, auxiliando para uma melhor execução da tarefa, diminuindo o nível de dificuldade para a realização da tarefa.

4.2.5 Interfaces Gráficas

Esta seção contém as interfaces do sistema associadas aos elementos e seus dados presentes em cada interface, assim como os fluxogramas contidos com suas informações e respectivas ações. Permite visualizar o comportamento independente de cada elemento e o elo entre os elementos de cada tela.

A principal contribuição desta seção para o processo de teste de *software* está associada à descrição dos fluxos das telas, possibilitando a visão da sequência de telas. E permite averiguar a conformidade das telas, verificando se as telas estão conforme especificadas.

4.2.6 Regra de Negócio

Esta seção contém as regras de negócio presentes no sistema. Cada regra além das informações básicas como nome, descrição, entrada e saída, traz os fluxogramas presentes detalhando como é seu funcionamento. Diferente da visão geral contida na Seção 4.2.2, esta seção possibilita uma visão do sistema de forma mais detalhada. Esta visão é possível devido à descrição do fluxo de cada regra por meio de sequência de ações.

A descrição da lógica auxilia no teste por deixar claro como deve ser o comportamento a ser testado. A contribuição desta seção também está associada a possibilitar que integrantes da equipe de teste que não estão inseridos no desenvolvimento possam compreender o funcionamento da lógica. E também na comunicação entre banco de dados e lógica.

4.2.7 Armazenamento

Esta seção contém as tabelas do sistema, trazendo informações detalhadas de cada campo, descrição da tabela, relacionamentos e mapeamento dos arquivos gerados pelo banco. Possibilita uma melhor compreensão do funcionamento do armazenamento do sistema, contribuindo para o entendimento necessário à verificação da consistência de dados. E também permite saber onde estão endereçados os documentos relacionados ao armazenamento do sistema.

A origem de grande parte dos dados para o preenchimento desta seção é proveniente do documento de dicionário de dados oriundo da equipe de banco de dados. O que torna esta seção um diferencial para a área de teste, são os relacionamentos entre tabelas e banco, contribuindo para que a visão dos elos pertinentes aos bancos da aplicação seja mais abrangentes e claros.

4.2.8 Testes

Apresenta o plano de teste com sua contextualização e casos de teste detalhados contendo descrição, fluxo, entrada e saída, estórias com suas funcionalidades e cenários detalhando passo a passo o caminho a ser realizado no teste. É a seção mais utilizada neste trabalho e é inserida na documentação proposta por servir de guia para os testes realizados, apresentando uma cobertura total de casos cobertos e descrevendo as funcionalidades a serem testadas.

Contém as estórias para realização dos testes passo-a-passo, permitindo a compreensão da funcionalidade a ser testada. Também possibilita uma maior abordagem dos casos de testes além de guiar o teste contribuindo para que ele seja realizado conforme desejado. Esta seção trás ao testador a possibilidade da execução da tarefa, com conhecimentos de base da funcionalidade.

4.3 Estudo de Caso

Nesta seção será apresentado o processo de desenvolvimento do LEMAF e a escolha do sistema no qual será avaliado a documentação proposta.

4.3.1 Processo de desenvolvimento do LEMAF

Como metodologia de desenvolvimento o LEMAF utiliza o *Scrum* adaptado, conhecida com *ScrumBut*. O *Scrum* é um processo de desenvolvimento ágil iterativo e incremental, que visa o aumento da produtividade e a redução do tempo de desenvolvimento. No *Scrum* os requisitos são levantados no começo de cada iteração e podem mudar conforme opinião do cliente.

O processo de desenvolvimento do LEMAF em sua maioria consiste na sequência: especificação, desenvolvimento e teste manual. Devido a metodologia de desenvolvimento utilizada, após o estabelecimento do contrato com o cliente, o escopo do projeto é elaborado em um *Product Backlog* que consiste em uma lista de itens priorizados a serem desenvolvidos.

O *Product Backlog* e a especificação do cliente são a documentação que acompanham o processo de desenvolvimento, as quais são utilizadas na realização dos testes que ocorrem na conclusão de cada ciclo. Durante a execução dos testes são necessárias mais informações além das documentadas, sendo necessária a busca por mais informações, ocasionando uma certa morosidade para executar os testes. Outro fator que ocasiona desacelaração no processo de teste de *software* é o tempo que o testador leva para compreender a execução da tarefa.

4.3.2 Escolha do sistema e funcionalidades

O sistema escolhido para aplicar o estudo da proposta foi o Análise Ambiental, que consiste em analisar processos de atividades na área ambiental que necessitem de licença governamental para prosseguirem com suas atividades. O fluxo do processo depende da classificação da atividade para ao qual a licença está sendo solicitada.

O sistema Análise Ambiental foi desenvolvido utilizando as tecnologias *ActionScript* para interface gráficas, *Java* para lógica de negócio e *Oracle* para camada de armazenamento de dados.

Por ser uma parte do fluxo comum a todos os processos e não comprometer informações sigilosas do governo foram escolhidos duas funcionalidades, que consistem na designação de um gestor responsável pela análise do processo e a designação de uma equipe:

- Vincular Gestor (Figura 4.1)
- Vincular Equipe (Figura 4.2)

Vincular Gestor:

Para vincular um Gestor é necessário estar logado com o perfil Diretor Técnico no módulo Análise Ambiental. Neste módulo, o Diretor deve pesquisar o processo ao qual deseja vincular um Gestor técnico. Este processo deve estar no status aguardando vinculação do Gestor. Com o processo aberto o Diretor Técnico deverá selecionar a opção de Vincular Gestor que trará a lista de gestores que podem ser vinculados juntamente com a quantidade de processo que cada um já tem participação. Depois da vinculação o gestor tem a opção de aceite ou recusa, estas duas ações são realizadas por meio da Agenda de Trabalho, um sistema paralelo que gerencia as mensagem recebidas por cada usuário, validando as ações no Aná-

lise Ambiental. Depois do aceite a vinculação está pronta, e o gestor se torna o responsável pela análise do processo. Associadas a esta etapas do fluxo há outras ações: Desvincular Gestor, Recusar Vinculação.

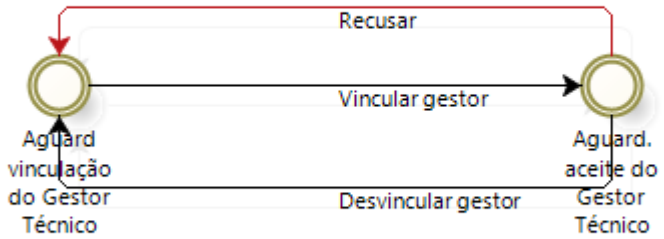

Figura 4.1: Fluxo Vincular Gestor

Vincular Equipe:

Para vincular uma equipe é necessário estar logado com o perfil de Gestor no módulo Análise Ambiental. Neste módulo, o Gestor deve pesquisar o processo ao qual deseja vincular uma equipe. Este processo deve estar no status aguardando vinculação da Equipe. Com o processo aberto o Gestor deverá selecionar a opção de Vincular Equipe que trará a lista de técnicos que podem ser vinculados juntamente com a quantidade de processo que cada um já tem participação. Depois da vinculação o(s) técnico(s) tem a opção de aceite ou recusa, estas duas ações são realizadas por meio da Agenda de Trabalho, um sistema paralelo que gerencia as mensagem recebidas por cada usuário, validando as ações no Análise Ambiental. Depois do aceite a vinculação está concluída, e o(s) técnico(s) auxilia(m) na análise do processo. Associadas a esta etapa do fluxo há outras ações: Desvincular Técnico, Recusar Vinculação, Vincular Novo Técnico, Vincular Técnico de Outra

Unidade Administrativa. Depois da finalização desta etapa o processo passa para o status Em Análise, e esta pronto para ser analisado.

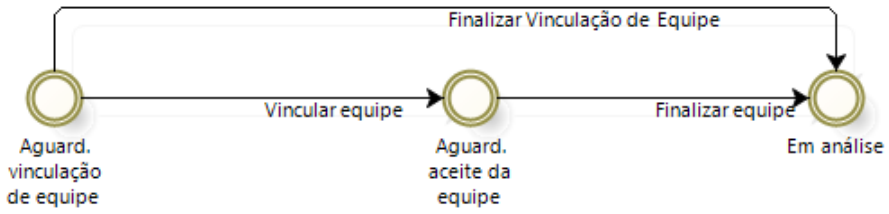

Figura 4.2: Fluxo Vincular Equipe

Para que a documentação pudesse ser avaliada, foi preenchida a partir de dados das funcionalidades escolhidas. No processo de preenchimento da documentação observou-se em cada seção:

- Equipe de desenvolvimento: Este dado não estava disponível em nenhuma documentação acessível aos colaboradores. Para o preenchimento desta seção foi necessário levantamento de dados da equipe desenvolvedora do sistema.
- Visão: A maioria dos dados desta seção estava contida no manual da aplicação, entretanto informações referentes a restrições do sistema, usuários e recursos tiveram que ser levantados com a equipe desenvolvedora do sistema.
- Requisitos: Houve dificuldades para o preenchimento desta seção. Os requisitos das funcionalidades não foram encontrados em nenhum registro do LEMAF. Esta situação ocorreu devido à metodologia de desenvolvimento

da empresa. Para preencher esta seção, foi necessária uma reunião com os desenvolvedores para recapitular os requisitos das funcionalidades.

- Casos de Uso: Não havia documentação com casos de usos das funcionalidades, todavia, para preenchimento dessa seção utilizou-se como base os casos de testes, contidos nos cenários dos testes automatizados.

- Interfaces gráficas: Foi realizada consulta ao código da interface para que esta seção fosse preenchida. Desta forma o código teve de ser interpretado e descrito em linguagem de negócio.

- Regra de negócio: Foi realizada consulta ao código da lógica para que esta seção fosse preenchida. Desta forma o código teve de ser interpretado e descrito em linguagem de negócio.

- Armazenamento: Foram consultados banco do sistema e dicionário de dados para que esta seção fosse preenchida. As informações foram extraídas e utilizadas para preencher a documentação.

- Teste: Seção preenchida a partir da documentação existente, nos casos de teste foram utilizados os cenários dos testes automatizados.

4.4 Validação

Nesta seção são apresentadas as validações realizadas para verificar se a documentação atendeu a coleta de dados presente na Seção 4.1. Inicialmente verificou-se cada dado coletado estava relacionado a documentação proposta, mapeando por meio de qual seção em caso afirmativo. Posteriormente foi apresentada a documentação e realizada coleta para verificar a porcentagem de sugestões atendidas. No fim foi coletado o depoimento da equipe de qualidade e gerencia em relação a documentação.

4.4.1 Mapeamento

Para verificar se as informações coletadas foram abordadas pelo meio solucionador proposto, realizou-se um mapeamento entre cada informação coletada e as seções presentes na documentação. Desta forma, relacionou de acordo com tipo da informação coletada (sugestão de artefato para documentação, falha detectada ou meio para o processo de teste de *software*) a(s) seção(ões) que contemplou(aram).

4.4.1.1 Mapeamento de artefatos sugeridos

Os artefatos sugeridos para guiar os testes foram agrupados e sintetizados totalizando dezesseis sugestões listados nas Tabelas 4.1 e 4.2. Após preenchimento da documentação com os dados das funcionalidades presentes na 4.3.2, relacionou-se cada artefato sugerido e a seção que o contemplou, em alguns casos mais de uma seção contemplou o artefato sugerido. Este mapeamento está presente nas Tabelas 4.6 e 4.7.

Analisando o mapeamento verificou-se que todas as seções estiveram presentes para contemplar as sugestões e que 93,75% das sugestões estão presentes na documentação e 6,25% das sugestões não foram contempladas pela documentação, conforme apresentado no gráfico da Figura 4.3.

4.4.1.2 Mapeamento de falhas identificadas

As falhas identificadas no processo de testes de *software* do LEMAF foram agrupadas e sintetizadas totalizando cinco falhas listadas na Tabela 4.3. Após preenchimento da documentação com os dados das funcionalidades presentes na 4.3.2, relacionou-se cada falha identificada e a seção que sanou a falha. Este mapeamento está presente na Tabela 4.8.

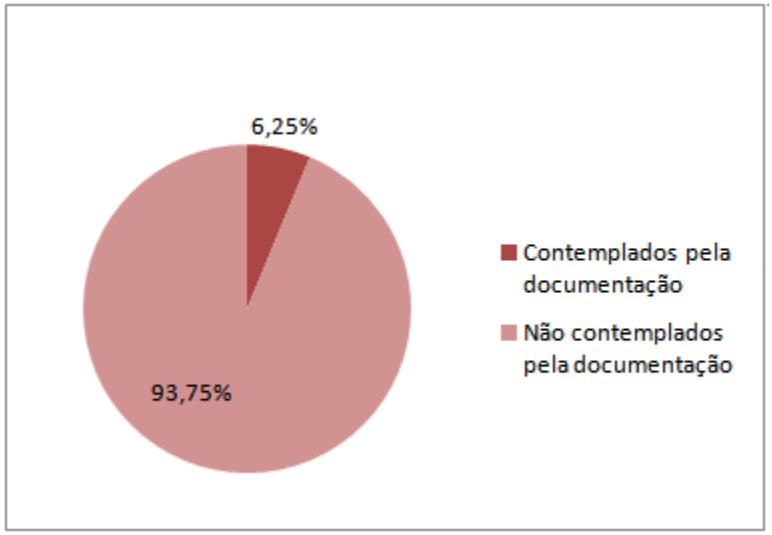

Figura 4.3: Mapeamento dos artefatos que foram contemplados pela documentação

Analisando o mapeamento verificou-se que 50% das seções foram utilizadas para sanar as falhas, sendo elas: Requisitos, Casos de Uso, Regra de Negócio e Testes. Quanto à porcentagem de falhas atendidas pela documentação, 60% das falhas foram sanadas, 20% não foram sanadas pela documentação proposta e 20% não puderam ser resolvidas por meio de documentação, sendo necessária uma mudança organizacional, conforme apresentado no gráfico da Figura 4.4.

4.4.1.3 Mapeamento de meios sugeridos para melhorar o processo de teste de *software*

Os meios sugeridos para melhorar o processo de testes de *software* foram agrupados e sintetizados totalizando quinze meios listados nas Tabelas 4.4 e 4.5. Após preenchimento da documentação com os dados das funcionalidades presentes na 4.3.2, relacionou-se cada meio sugerido e a seção que contemplou o meio, em alguns casos mais de uma seção contemplou o meio sugerido. Este mapeamento está presente nas Tabelas 4.9 e 4.10.

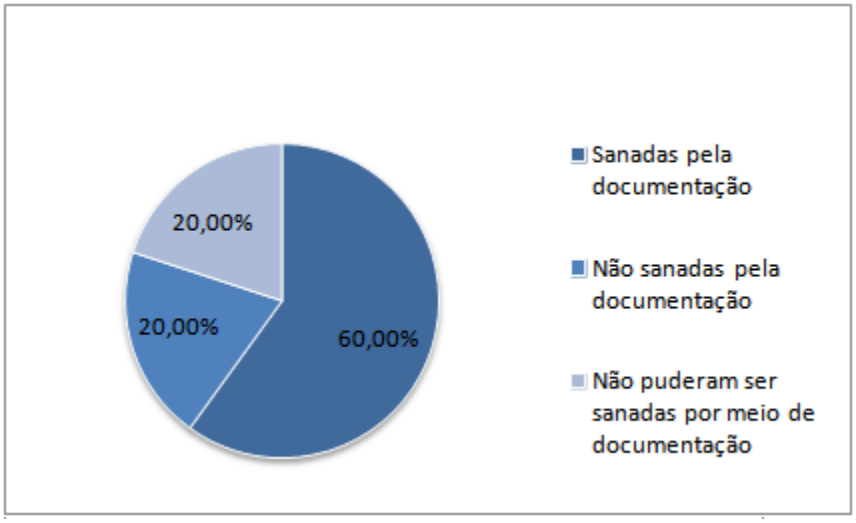

Figura 4.4: Mapeamento das falhas identificadas sanadas pela documentação

Analisando o mapeamento verificou-se que todas as seções foram utilizadas para contemplar os meios sugeridos. Quanto à porcentagem dos meios que foram contemplados pela documentação, 66,66% dos meios foram contemplados, 13,33% não foram contemplados pela documentação proposta, já 13,33% não puderam ser resolvido por meio de documentação, sendo necessária uma mudança organizacional e adoção de novas metodologias de desenvolvimento. Em um percentual de 6,66% dos meios sugeridos não pode ser definido se houve contemplação ou não, visto que é necessária uma análise mais profunda para verificar se o meio foi contemplado, conforme apresentado no gráfico presente na Figura 4.5.

4.4.1.4 Mapeamento de seções e dados coletados

Para uma melhor visualização referente a quais seções atenderam aos dados coletados, foi realizado um mapeamento englobando todos os dados. Este mapeamento está presente na Tabela 4.11.

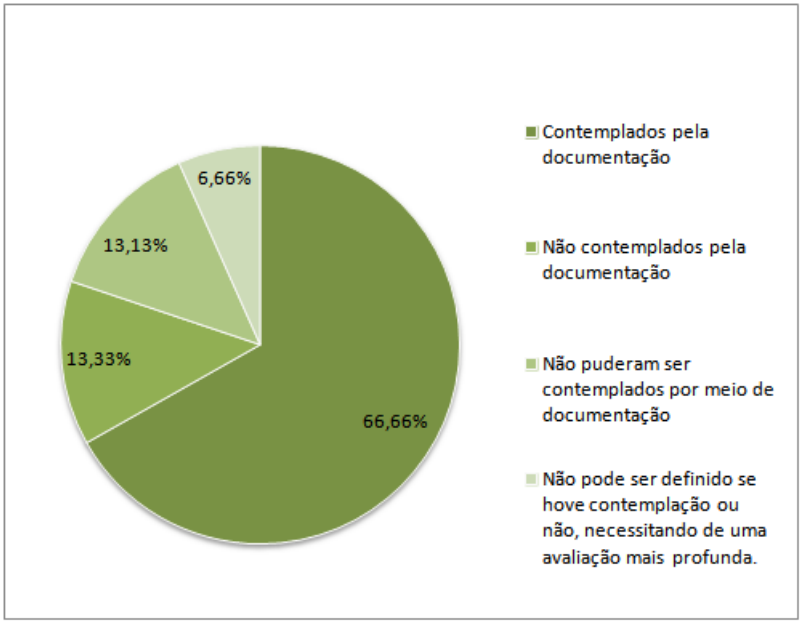

Figura 4.5: Mapeamento de meios sugeridos contemplados pela documentação

4.4.1.5 *Feedback* da equipe

A validação por meio de *feedback* se deu de duas formas. A primeira delas com a utilização do formulário descrito na Seção 3.1.2. A segunda utilizando a documentação para realizar os testes descritos na seção 4.3.2.

Na primeira foram coletados 119 *feedbacks* de sugestões, aos quais 91 alegaram que a documentação atendeu a sugestão, 19 respostas diziam que a documentação não atendeu a sugestão, e 9 coletas disseram que a documentação atendeu parcialmente a sugestão. Estes resultados estão dispostos no gráfico presente na Figura 4.6:

Para que se pudesse verificar por meio de opinião se a documentação atendeu as sugestões coletadas, quatro testadores, utilizaram a documentação para realizar

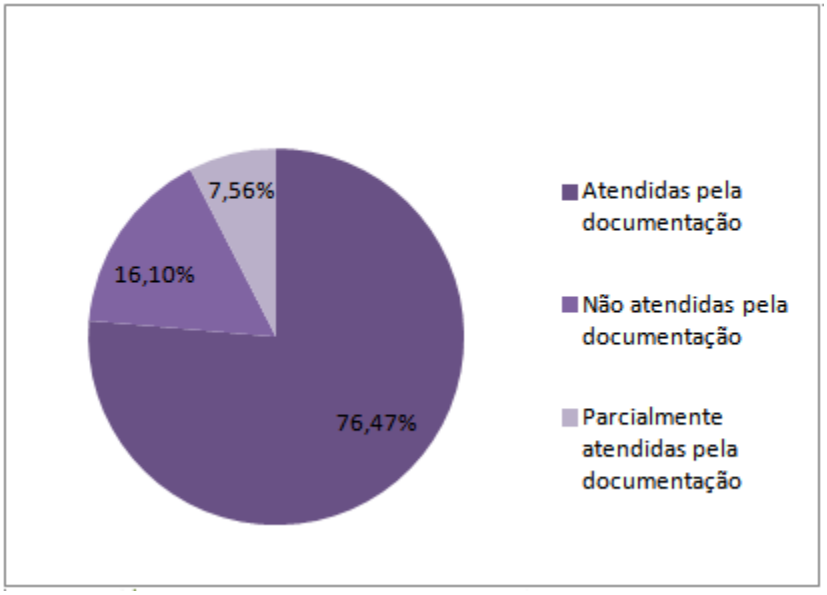

Figura 4.6: Feedback se as sugestões dos colaboradores foram atendidas pela documentação

os testes referente as funcionalidades descritas na Seção 4.3.2. Após realização dos testes, cada testador apresentou seu *feedback* relatando o uso da documentação;

O testador I evidenciou que os fluxos tornaram o estudo necessário para realizar o teste mais simples. Por meio da documentação foi possível entender a funcionalidade a ser testada com mais facilidade, e entender o contexto. Outro ponto positivo da documentação foi o detalhamento da regra de negócio. Relatou que as informações foram suficientes para a realização do teste. Ao fim sugeriu que um índice otimizaria a usuabilidade da documentação.

Já o testador II relatou que apesar da dificuldade inicial de localizar as informações, todas as informações necessárias ao teste estavam presentes na documentação. A documentação facilitou na compreensão do caso de testes, assim como o conhecimento das entradas e saídas esperadas. A documentação também permitiu conhecer a sequência de telas, assim como o funcionamento de cada uma delas. Afirmou que outra contribuição bastante evidente da documentação foi o

acesso aos requisitos, permitindo fazer a validação e verificação do teste conforme estabelecido com o cliente.

O testador III declarou que em sua percepção a documentação é bem intuitiva e clara sendo de fácil leitura por utilizada a linguagem de negócio. Confirmou que auxilia na realização de testes, principalmente à quem não tem conhecimento específico sobre as funcionalidades do sistema. Ao concluir sua opinião declarou "Documentação muito útil. Gostei bastante.".

Enquanto o testador IV, afirmou que a documentação ajuda no entendimento do funcionamento da tela, casos de uso e na montagem dos cenários de teste. Ainda afirma que interfere positivamente na qualidade do teste gerado e no tempo despendido para gerá-lo, mas que isto é condicionado a uma boa qualidade da documentação. Diz que tratando do contexto geral de desenvolvimento do *software* os ganhos adquiridos com a utilização da documentação devem ser comparados ao esforço gasto na geração da documentação e analisando-se os ganhos, pois uma vez gerada a documentação, esta pode ser utilizada várias vezes para a geração dos testes.

Nº	Sugestão de artefatos para a documentação		
	Sugestão	Objetivo	Área
1	Tutorial com atividades básicas	Contribuir para que atividades, ao serem realizadas possam ser executadas em menor tempo e com menor incidência erros.	Banco de Dados
2	Prover maior detalhamento da especificação	Contribuir para que nas realizações dos testes possam ser feitas a verificação e validação da atividade em teste, averiguando se está atendendo o que foi definido.	Banco de Dados
3	Conexão do projeto com outros sistemas	Possibilitar uma visão simplificada do todo, permitindo compreender os elos existentes entre os sistemas e identificas as possíveis saídas esperadas de acordo com as entradas inseridas.	Banco de Dados
4	Integrantes do projeto	Permitir localizar com falicilidade quem são os responsáveis pelo projeto em caso de necessidade	Banco de Dados
5	Telas do sistema	Contribuir para visualizar os elementos da tela, transmissão de uma tela para outra permitindo conhecer o funcionamento de cada telas e fluxo entre as telas do sistema.	Banco de Dados
6	Documentação de requisitos	Contribuir para que o teste verifique se o sistema atende as especificações que foram realizadas junto ao cliente.	Banco de Dados
7	Noção geral de como o sistema funciona	Permite compreender o funcionamento do sistema como um todo, contribuindo para que o teste seja realizado de maneira integrada, testando também as funcionalidades do sistema em conjunto.	Desenvolvimento / Qualidade
8	Visão de negócio / Regra de negócio	Contribuir para que na execução do teste seja testada a regra de negócio, por possibilitar uma compreensão da lógica por meio de descrição dos fluxos de ações do sistema. O que torna o teste mais eficaz.	Desenvolvimento

Tabela 4.1: Sugestão de artefatos para a documentação

Nº	Sugestão de artefatos para a documentação (Continuação)		
	Sugestão	Objetivo	Área
9	Funcionalidades e seus impactos	Tornar mais clara a identificação dos impactos causados por alterações nas funcionalidades, contribuindo para verificação de onde as ações terão efeitos.	Desenvolvimento
10	Regra de negócio por requisito	Contribuir para que o teste da verificação de cada requisito seja mais abrangente por meio de uma visão melhorada das regras de cada requisito.	Desenvolvimento
11	Dependência entre telas	Facilitar a compreensão de como as telas são agrupadas por meio do conhecimento do funcionamento e conexão entre elas.	Desenvolvimento
12	Fluxo	Facilitar a compreensão da sequência lógica e dinâmica do processo, desta forma contribuindo mais para o entendimento do que explicações em texto puro.	Gerência
13	Contextualização do ambiente	Contribuir para compreensão de qual meio está inserido o teste a ser realizado, permitindo que o teste seja realizado de maneira mais segura e precisa, sendo a contextualização do ambiente um fator essencial para que um teste seja bem realizado.	Qualidade
14	Origem e finalidade da estória	Permitir saber em qual contexto a estória está inserida e qual sequência que existe antes e após.	Qualidade
15	Transição de tela	Facitar a compreensão dos fluxos das funcionalidades, e contribuir para que o teste verifique se a transição de telas ocorre de maneira correta.	Qualidade
16	Fluxos de ações do usuário	Possibilitar entendimento de sequência de ações do usuário, permitindo que o teste seja realizado simulando a utilização do usuário.	Qualidade

Tabela 4.2: Sugestão de artefatos para a documentação

Nº	Falhas e dificuldades do setor de teste		
	Falha	Efeito	Área
1	Não conhecimento exato do teste a ser realizado	Testadores alegaram que algumas vezes ao executar atividades de testes, principalmente em funcionalidades novas, não há compreensão de como o teste deve ser realizado, o que deve ser testado e as saídas esperadas.	Qualidade
2	Carência de teste para validar lógica	Não há um artefato que facilite a compreensão da lógica, guiando como esta funciona, sendo necessária consulta ao desenvolvedor para conhecimento da lógica, ou até mesmo não execução de testes para validar todas as lógicas.	Desenvolvimento
3	Carência de teste de carga	teste importante, pouco existente na empresa.	Desenvolvimento
4	Não há um guia para o teste	Ausência informações que descrevam o funcionamento do sistema, sendo necessário um estudo e/ou um levantamento de informações antes de ser realizado o teste.	Qualidade
5	Testador separado da equipe de desenvolvimento	Dificulta o entrosamento e faz com que o testador perca as vantagens de acompanhar o desenvolvimento de perto, o que deixa de acarretar em beneficios na hora de realizar o teste.	Desenvolvimento

Tabela 4.3: Falhas e dificuldades do setor de teste

Nº	Meios para melhorar o processo de desenvolvimento		
	Meio	Justificativa	Área
1	Padronização	Facilitar a compreensão e contribuir com a familiariadade independente de qual sistema está sendo realizado o teste.	Banco de Dados
2	Especificar melhor o que deve ser desenvolvido	Os requisitos devem ser bem definidos, a modelagem deve ser feita antes do desenvolvimento e especificação de como as entidades são amarradas. Isto contribui para que o teste possar acompanhar o desenvolvimento e ter visão melhor do todo, melhorando o teste e tornando-o mais fácil de ser realizado.	Desenvolvimento
3	Testar coerência dos dados	Teste importante, pouco utilizado na empresa. Verificar se os dados estão sendo processados como esperados.	Desenvolvimento
4	Testador alocado na equipe	Implica em conhecimento do testador da aplicação e como esta fluindo o desenvolvimento da empresa, resultando em testes mais elaborados.	Desenvolvimento
5	Casos de uso prontos antes da realização do teste	Facilitar a realização do teste, sem a necessidade de criar os casos de teste durante a realização dos testes.	Desenvolvimento
6	Ter no documento apenas o que agrega valor	Informações em excesso prejudicam a busca por informações e são desnecessárias.	Desenvolvimento
7	Treinamento sobre como funciona o desenvolvimento geral da equipe	Possibilita que um novo integrante se adapte a equipe mais rápido e contribui para que o integrante atenda às expectativas da equipe.	Desenvolvimento
8	Estratégia de testes mais profundas	Sistemas devem ser testados mais a fundo, garantindo que o máximo das possíveis combinações das ações de cada funcionalidade estejam funcionando.	Gerência

Tabela 4.4: Meios para melhorar o processo de desenvolvimento

Nº	Meios para melhorar o processo de desenvolvimento (Continuação)		
	Meio	Justificativa	Área
9	Mecanismo de validação de teste da prática XP	Utilização do desenvolvimento guiado por testes.	Gerência
10	Documentação que ajude a compreender melhor a execução da tarefa	Algumas vezes tarefas são realizadas sem conhecimento adequado.	Qualidade
11	Integração maior entre equipes	Há falta de integração entre as equipes, o que torna a execução dos testes, tarefa que necessita de informações de outras equipes, mais dispendiosa.	Qualidade
12	Visão mais profunda	Visão existente é superficial, o que acarreta em testes superficiais.	Qualidade
13	Validar regra de negócio de forma automatizada	Agregar as vantagens do teste automatizado na verificação da Regra de Negócio.	Qualidade
14	Unificar caso de uso e caso de teste	Os dois tipos de casos possuem a mesma sequência e apesar de finalidades diferentes, podem ser elaborados em conjunto, evitando que um trabalho seja replicado.	Qualidade
15	Planejamento bem feito, detalhado	A partir de um planejamento de testes mais elaborado, tentar depender menos da equipe de desenvolvimento.	Qualidade

Tabela 4.5: Meios para melhorar o processo de desenvolvimento

Nº	Mapeamento de sugestões atendidas pela documentação		
	Sugestão	Seção(ões) que atende(m)	Comentário
1	Tutorial com atividades básicas	4 e 8;	Contemplada pois as atividades de teste são descritas em formato de fluxo na 4.2.4 e em formato de texto passo-a-passo nas estórias presentes na 4.2.8.
2	Prover maior detalhamento da especificação	3;	Contemplada pois os requisitos são descritos detalhadamente na 4.2.3.
3	Conexão do projeto com outros sistemas	2 e 7;	Contemplada pois a 4.2.2 permite uma visão do todo abrangendo as conexões existentes com outros sistemas e a 4.2.7 contém os relacionamentos entre os bancos de outros sistemas.
4	Integrantes do projeto	1;	Contemplada pois a 4.2.1 contém os dados da equipe do processo.
5	Telas do sistema	5;	Contemplada pois a 4.2.5 contém os elementos presentes em cada interface e o fluxograma de telas.
6	Documentação de requisitos	3;	Contemplada pois a 4.2.3 descreve os requisitos dos sistema.
7	Noção geral de como o sistema funciona	2;	Contemplada pois a 4.2.2 permite uma visão do todo, abrangendo as funcionalidades de forma integrada.
8	Visão de negócio / Regra de negócio	6;	Contemplada pois a 4.2.6 contém a descrição das regras de negócio e o fluxograma de regras.

Tabela 4.6: Sugestões de artefatos da coleta de dados atendidas pela documentação

Nº	Mapeamento de sugestões atendidas pela documentação (Continuação)		
	Sugestão	Seção(ões) que atende(m)	Comentário
9	Funcionalidades e seus impactos	2, 4 e 8;	Contemplada pois a 4.2.2 aborda as funcionalidades de forma integrada, permitindo o conhecimento entre elas, a 4.2.4 contém os relacionamento entre as funcionalidades e pela 4.2.8 que contém as estórias passo-a-passo possibilitando conhecer quais serão os resultados das funcionalidades.
10	Regra de negócio por requisito	Não atendida.	Sugestão não contemplada pelo documento.
11	Dependência entre telas	5;	Contemplada pois a 4.2.5 contém o fluxograma de telas, permitindo visualizar a dependência entre telas.
12	Fluxo	4, 6 e 8;	Contemplada pois a 4.2.4 contém o fluxo de ações do usuário em fluxogramas enquando a Seção a 4.2.8 em formato de texto e a 4.2.6 o fluxo das regras de negócio.
13	Contextualização do ambiente	2;	Contemplada pois a 4.2.2 permite uma visão do todo abrangendo, contextualizando a finalidade do sistema e seu funcionamento.
14	Origem e finalidade da estória	4 e 8;	Contemplada pois a 4.2.4 contém os casos de usos de ações do usuário permitindo conhecer a origem e finalidade da estória por meio de fluxo, assim com a 4.2.8 que fornece as mesmas condições em formato de texto.
15	Transição de tela	5;	Contemplada pois a 4.2.5 contém o fluxograma de telas, permitindo visualizar a sequencia de telas.
16	Fluxos de ações do usuário	4 e 8;	Contemplada pois a 4.2.4 contém o fluxo de ações do usuário em fluxogramas e a 4.2.8 contém o fluxo em formato de texto.

Tabela 4.7: Sugestões de artefatos da coleta de dados atendidas pela documentação

Nº	Mapeamento das falhas identificadas sanadas pela documentação		
	Falha	Seção(ões) que sana(m)	Comentário
1	Não conhecimento exato do teste a ser realizado	2, 4 e 8;	Sanada pois a 4.2.2 contém uma visão geral do sistema, a 4.2.4 contém o fluxo de ações e a 4.2.8 contém a descrição passo-a-passo do teste a ser executado.
2	Carência de teste para validar lógica	6;	Sanada pois a 4.2.6 contém a descrição das regras e o fluxograma entre elas;
3	Carência de teste de carga	Não sanada.	Não sanada por meio da documentação proposta.
4	Não há um guia para o teste	8;	Sanada pois a 4.2.8 contém a descrição passo-a-passo do teste a ser executado, sendo um guia em texto puro para a realização do teste.
5	Testador separado da equipe de desenvolvimento	Não pode ser resolvido por meio de documentação	Esta falha identificada não pode se resolvida por meio da documentação, deve ser uma mudança organizacional.

Tabela 4.8: Falhas identificadas na coleta de dados sanadas pela documentação

Nº	Mapeamento de meios sugeridos que tiveram contribuição pela documentação		
	Meio	Seção(ões) que contribui(em)	Comentário
1	Padronização	1, 2, 3, 4, 5, 6, 7 e 8;	Contemplado pois todas as Seções seguem um padrão de linguagem normatizada.
2	Especificar melhor o que deve ser desenvolvido	3;	Contemplado pois a Seção 3 contém a descrição dos requisitos estabelecidos junto ao cliente.
3	Testar coerência dos dados	4, 7 e 8;	Contemplado pois a Seção 4 contém a relação de entradas e saídas esperadas em formato de fluxo, assim como a Seção 8 atende em formato de texto puro.
4	Testador alocado na equipe	Não pode ser resolvido por meio de documentação	Este meio não pode se resolvido por meio da documentação, deve ser uma mudança organizacional.
5	Casos de uso prontos antes da realização do teste	4;	Contemplado pois a Seção 4 contém os casos de uso.
6	Ter no documento apenas o que agrega valor	Indefinido	Todas as áreas da documentação atenderam de alguma forma uma sugestão, falha ou meio, contudo não se pode afirma que não há nenhuma informação nescessária a partir do experimento realizado.
7	Treinamento sobre como funciona o desenvolvimento geral da equipe	Não atendido.	Este meio não pode se resolvido por meio da documentação, deve ser uma mudança na metodologia de trabalho.
8	Estratégia de testes mais profundas	2, 3, 4, 5, 6, 7 e 8;	Contemplado pois todas as Seções contribuem para que o teste seja realizado com mais detalhes, abrangendo todas as àreas que compoem o *software*

Tabela 4.9: Sugestões de meios para melhorar o processo de teste de *software* contribuidos pela documentação

Nº	Mapeamento de meios sugeridos que tiveram contribuição pela documentação		
	Meio	Seção(ões) que contribui(em)	Comentário
9	Mecanismo de validação de teste da prática XP	Não pode ser resolvido por meio de documentação	A documentação não aborda os meios de validação presentes na técnica XP.
10	Documentação que ajude a compreender melhor a execução da tarefa	2, 3, 4, 5, 6, 7 e 8;	Contemplado pois todas as Seções contribuem com informações que ajudam na compreensão do teste.
11	Integração maior entre equipes	1, 2, 3, 4, 5, 6, 7 e 8;	Contemplado pois todas as Seções contém informações referente ao *software* oriundas de áreas diferentes um um único lugar, centralizando informações de equipe. Desta forma, permitindo o conhecimento de de uma equipe diferente.
12	Visão mais profunda	5, 6 e 7;	Todas as áreas da documentação contribuem para uma visão mais detalhada, contudo as Seções 5, Seção 6 e Seção 7 contém informações do funcionamento em um nível detalhado.
13	Validar regra de negócio de forma automatizada	Não atendido.	A documentação não aborda meios para validar a regra de negócio de forma automatizada.
14	Unificar caso de uso e caso de teste	4 e 8;	A documentação contém os mesmos fluxos de caso de uso na Seção 4 e Seção 8, sendo apresentados em formato de fluxograma na Seção 4 e em formato de texto na Seção 8.
15	Planejamento bem feito, detalhado	1, 2, 3, 4, 5, 6, 7 e 8;	O documento proporciona meios para que o planejamento do teste seja melhor elaborado, e informações de todo o *software*, evitando a dependência de integrantes de outra equipe.

Tabela 4.10: Sugestões de meios para melhorar o processo de teste de *software* contribuidos pela documentação

Nº	Mapeamento de seções e dados coletados			
	Nome Seção	Sugestão de artefatos	Sugestão de Meio	Falhas identificadas
1	Equipe de desenvolvimento	4;	1, 8, 11 e 15;	
2	Visão	3, 7, 9 e 13;	1, 8, 10, 11 e 15;	1;
3	Requisitos	2 e 6;	1, 2, 8, 10, 11 e 15;	
4	Casos de Uso	1, 9, 12, 14 e 16;	1, 3, 5, 8, 10, 11, 14 e 15;	1;
5	Interfaces Gráficas	5, 11 e 15;	1, 8, 10, 11, 12 e 15;	
6	Regra de negócio	8 e 12;	1, 8, 10, 11, 12 e 15;	2;
7	Armazenamento	3;	1, 3, 5, 8, 10, 11, 12 e 15;	
8	Testes	1, 9, 12, 14 e 16;	1, 3, 5, 8, 10, 11, 14 e 15;	1 e 4;

Tabela 4.11: Dados coletados atendidos pela documentação

5 CONCLUSÃO

No processo de desenvolvimento de *software* há uma dificuldade em estabelecer uma comunicação consistente, devido as diferentes linguagens de comunicação utilizadas pelas áreas desse processo. Esta dificuldade impacta na realização dos testes devido a necessidade de unir informações oriundas das diferentes áreas do desenvolvimento, ocasionando desaceleração no desenvolvimento.

Tendo em vista essa desaceleração o principal objetivo deste trabalho foi identificar as falhas e dificuldades que causam esse quadro, buscando um meio solucionador. Os objetivos envolveram coletar dados no LEMAF, aos quais submetidos a análise, levaram como meio de dispor os resultados a criação de um documento padronizado e centralizado. Destarte, essa documentação direcionou a pesquisa de informações necessárias para realização do teste para uma única fonte de informações otimizando o tempo da pesquisa, contribuiu para que a comunicação no processo de desenvolvimento fosse clara e compreensível e serviu de guia para as atividades de teste de *software*.

A documentação proposta contém oito seções que objetivam englobar o maior número das informações coletadas, auxiliando na compreensão da execução do teste. Para que documentação pudesse ser validada, foi preenchida com os dados das funcionalidades escolhidas, sendo realizados testes para análise da contribuição da proposta, mapeamento das sugestões atendidas na documentação e coletado *feedback* da equipe.

As informações coletadas foram sintetizadas e agrupadas em: sugestões de artefatos para a documentação , falhas detectadas no processo de teste de *software* e sugestões de meios para melhorar o processo de teste de *software*, a síntese consistiu em dezesseis sugestões, cinco falhas e quize meios.

Analisando os dados, todas as seções da documentação foram utilizadas para atender as sugestões, o qual atendeu à 93,75% da sugestões, sanou 60% das falhas e contribui para 66,66% do meio sugeridos. As porcentagens apresentadas mostram que a documentação foi eficiente na resolução dos dados coletados, conseguindo melhorar o processo de teste de *software*.

Considerando a avaliação da documentação pela equipe, os *feedbacks* foram positivos e comprovaram que a documentação contribuiu para a melhoria da qualidade do *software*, guiando a atividade de teste, permitindo uma percepção do todo além de proporcionar condições para estratégias de testes mais profundas.

Os resultados apresentados mostram que uma documentação centralizada e uma boa comunicação são essenciais para o desenvolvimento com qualidade visto que influem no processo de teste *software*.

REFERÊNCIAS BIBLIOGRÁFICAS

ABERNETHY, K.; GABBERT, P.; TREU, K.; PIEGARI, G.; REICHGELT, H. Impact of the emerging discipline of information technology on computing curricula: some experiences. *J. Comput. Sci. Coll.*, Consortium for Computing Sciences in Colleges, USA, v. 21, n. 2, p. 237–243, dez. 2005. ISSN 1937-4771. Disponível em: <http://dl.acm.org/citation.cfm?id=1089053% -.1089089>.

AMBLER, S. W. *Agile/Lean Documentation: Strategies for Agile Software Development*. 2004. Http://www.agilemodeling.com/essays/agileDocumentation.htm.

BORGES, G. *Engenharia de Software: Metodologia de Desenvolvimento de Sistemas*. 2010.

FERNANDES, A. da G. F. Comunicação interna na empresa: um grande desafio. *Revista IMES - Instituto Municipal de Ensino Superior de São Caetano do Sul*, ano XVI, n. 47, p. p.13–18, 1999.

FORWARD, A. *Software Documentation – Building and Maintaining Artefacts of Communication*. 2002. Ottawa-Carleton Institute for Computer Science - University of Ottawa.

FURLANETTO, S. *Diagnóstico do processo de comunicação nas empresas do extremo sul catarinense, segundo a ótica dos acadêmicos da área de negócios da UNESC*. 2001. Dissertação (Mestrado) - Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia de Produção.

JUNG, C. F. *Metodologia Para Pesquisa & Desenvolvimento: Aplicada a Novas Tecnologias, Produtos e Processos*. [S.l.]: 01/2004 1ª edição, 2004.

LANA, F. V. D. *A COMUNICAÇÃO NO PROCESSO DE DESENVOLVIMENTO DE SOFTWARE E A SATISFAÇÃO DO USUÁRIO*. 2008. Dissertação apresentada ao Curso de Mestrado em Administração da Universidade Federal de Santa Maria.

MEGGINSON, L. C.; C.MOSLEY, D.; JR., P. H. P. *Administração Conceitos e Aplicações*. [S.l.]: Harbra, 1998.

MORAES, T. M.; SOUZA, A. S. de; OLIVEIRA, J. L. de. *Revisão sistemática sobre a comunicação dentro do processo de desenvolvimento de software*. 2011. Revisão Sistemática do Instituto de Informática da Universidade Federal de Goiás.

NETO, A. C. D. Introdução a teste de software. *Engenharia de Software Magazine*, Edição Especial, p. 54–59, 2007.

NUNES, V. B.; SOARES, A. O.; FALBO, R. A. *Apoio à Documentação em um Ambiente de Desenvolvimento de Software*. 2004. Memorias de VII Workshop Iberoamericano de Ingeniería de Requisitos y Desarrollo de Ambientes de Software.

OLIVEIRA, F. S. *A comunicação nas organizações do terceiro setor - Serviço Assistencial Salão do Encontro: Um Estudo de Caso.* 2003. Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico. Programa de Pós-Graduação em Engenharia de Produção.

PAULA, M. G. de. *ComunIHC-ES: Ferramenta de Apoio à Comunicação entre Profissionais de IHC e Engenheiros de Software*. 2007. Tese (Doutorado em Informática) – Pontifícia Universidade Católica do Rio de Janeiro.

ROCHA, A. R. C. da; MALDONADO, J. C.; WEBER, K. C. Qualidade de software: teoria e prática. *Prentice Hall, 2001*, p. 303, 2009.

SALLES, T. P. Melhorias no processo de comunicaÇÃo e gerenciamento de requisitos alinhado ao babok – um estudo de caso. *Revista de Sistemas e Computação, Salvador*, v. 1, n. 2, p. p. 120–138, 2011.

SOMMERVILLE, I. *Engenharia de Software*. [S.l.]: 8. ed. São Paulo, 2007.

SOUSA, C. M. da Silva e. *CENTRO DE MEMÓRIA DA CTBC E SUA INTEGRAÇÃO AO PROCESSO DE COMUNICAÇÃO DA EMPRESA: UM ESTUDO DE CASO*. 2004. Dissertação (Mestrado) - Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia de Produção.

TOSETTO, M.; BELLINI, C. G. P. Gestão sociotécnica do teste de software em projetos de sistemas de informação. *Journal of Information Systems and Technology Management*, Vol. 5, p. p. 325–346, 2008.

VICENTE, A. A. *Definição e gerenciamento de métricas de teste no contexto de métodos ágeis*. Março 2010. Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação — ICMC/USP, como parte dos requisitos para obtenção do título de Mestre em Ciências - Ciências de Computação e Matemática Computacional. USP - São Carlos.

WALLACE, D. R.; FUJII, R. U. Software verification and validation: An overview. *IEEE Software*, 1989.

A FORMULÁRIO DE COLETA DE DADOS

Formulário de coleta de opiniões

A comunicação existente no processo de desenvolvimento de software no LEMAF não é clara e objetiva o bastante, o que dificulta a produção de software, pois ocasiona em uma dificuldade para executar os testes devido à dificuldade de encontrar informações para testar o sistema.

Dada a desaceleração no processo de teste de software objetivamos identificar as falhas e dificuldades que causam esse quadro.

Em sua opinião:

1- O teste realizado no LEMAF supri todas as necessidades?

2- O que você indica como falha que ocorre no setor de teste?

3- Como você acha que "isso" pode ser solucionado?

4- Há outro fator importante no desenvolvimento de teste que não abordamos?

Obrigada pela participação!

Figura A.1: Formulário de coleta de dados

B FORMULÁRIO DE FEEDBACK DADOS

Formulário de feedback se as sugestões dos colaboradores foram atendidas pela documentação

Sugestões coletadas para detecção de falhas que constam na documentação.

Colaborador A

SUGESTÃO	RESPOSTA		
	SIM	NÃO	PARCIALMENTE
Sugestão A.1			
Sugestão A.2			
Sugestão A.3			
...			
Sugestão A.n			

Figura B.1: Formulário de feedback dos dados

C ESTRUTURA DO DOCUMENTO PROPOSTO

```
<?xml version="1.0" encoding="utf-8"?>


<!-- Equipe Desenvolvedora -->
<!-- Descreve a equipe que está sendo contratada para realizar o serviço: Seguimento de
<equipe_desenvolvedora>
  <descricao></descricao>
     <recursos>
        <!-- Descrimina a equipe da desenvolvedora :: Número de membros ilimitados -->
    <equipe>
      <membro>
        <nome></nome>
        <papel></papel>
        <email></email>
        <telefone></telefone>
      </membro>
    </equipe>
  </recursos>
</equipe_desenvolvedora>
<!-- Equipe Desenvolvedora -->

<!-- VISAO -->
<!-- Corresponde ao documento de visão de um sistema :: O DOCUMENTO DE VISÃO DE É ALTO
<visao>

  <!-- Apresente as principais funcionalidades do sistema -->
```

```
  <caracteristicas></caracteristicas>

  <!-- Número de funcionalidades ilimitados-->
  <funcionalidades>
    <funcionalidade>
    </funcionalidade>
  </funcionalidades>

 <!-- Apresente as principais restrições a que o produto está sujeito, principalmente c
relação a volume de informação, tempo de resposta, plataforma operacional, políticas de
homologação, segurança, auditoria, entre outros -->

  <restricoes></restricoes>

  <!-- Potenciais usuarios do sistema-->
  <usuarios>

    <caracterizacao></caracterizacao>

    <!-- Como o sistema impacta no trabalho do usuário-->
    <utilizacao_do_sistema></utilizacao_do_sistema>

    <!-- Mercado potencial do produto-->
    <mercado_alvo></mercado_alvo>

  </usuarios>

  <recursos>
```

```
    <!-- Descreve quais recursos são necessários para montar o sistema -->
  </recursos>

</visao>
<!-- VISAO -->

<!-- REQUISITOS -->
<requisitos>

  <funcionais>
    <requisito>
      <id></id> <!-- identificador -->
      <descricao></descricao> <!-- descricao geral -->
      <prioridade></prioridade> <!-- Prioridade desse requisito -->
      <tamanho></tamanho> <!-- numérica (fibonacci) / Scrum based -->
    </requisito>
  </funcionais>

  <nao_funcionais>
    <requisito>
      <id></id> <!-- identificador -->
      <descricao></descricao> <!-- descricao geral -->
      <prioridade></prioridade> <!-- Prioridade desse requisito -->
      <tamanho></tamanho> <!-- numérica (fibonacci) / Scrum based -->
    </requisito>
  </nao_funcionais>

</requisitos>
```

```
<!-- REQUISITOS -->

<!-- CASOS DE USO -->
<casos_de_uso>

  <caso_de_uso>
    <id></id>
    <nome></nome>
    <atores>
      <ator>
        <nome></nome>
        <tipo></tipo>
        <relacionamentos>
          <ator></ator>
          <descricao></descricao>
          <tipo></tipo>
        </relacionamentos>
      </ator>
    </atores>
    <descricao></descricao>
    <pre_condicao>
      <!-- lista de pré condições que devem ser satisfeitas para realizar o caso de uso
    </pre_condicao>
    <pos_condicao>
      <!-- lista de possíveis estados resultantes do caso de uso -->
    </pos_condicao>
    <relacionamentos>
      <relacionamento>
```

```
        <caso_de_uso></caso_de_uso>
        <descricao></descricao>
        <tipo></tipo>
      </relacionamento>
    </relacionamentos>
  </caso_de_uso>

</casos_de_uso>
<!-- CASOS DE USO -->


<!-- INTERFACES GRAFICAS -->
<interfaces>

  <inteface>
    <id></id>
    <nome></nome>
    <tipo></tipo>
    <descricao></descricao>
    <elementos>
      <elemento>
        <id></id>
        <nome></nome>
        <tipo></tipo>
        <descricao></descricao>
        <entrada_saida></entrada_saida>
      </elemento>
    </elementos>
```

```
    <fluxogramas>
      <fluxograma>
        <id></id>
        <nome></nome>
        <descricao></descricao>
        <acoes>
          <acao>
            <nome></nome>
            <elementos></elementos>
            <ordem></ordem>
          </acao>
        </acoes>
      </fluxograma>
    </fluxogramas>
  </inteface>

</interfaces>
<!-- INTERFACES GRAFICAS -->

<!-- REGRA DE NEGOCIO -->
<regras>

  <regra>
    <id></id>
    <nome></nome>
    <descricao></descricao>
    <entrada>
      <propriedade id="" nome="" tipo=""/>
```

```
    </entrada>
    <saida>
      <propriedade id="" nome="" tipo=""/>
    </saida>
    <fluxogramas>
      <fluxograma>
        <id></id>
        <nome></nome>
        <descricao></descricao>
        <acoes>
          <acao>
                        <id></id>
                        <descricao></descricao>
            <entradas>
                <entrada></entrada>
            </entradas>
            <saidas>
                <saida></saida>
            </saidas>
            <ordem></ordem>
          </acao>
        </acoes>
      </fluxograma>
    </fluxogramas>
  </regra>

<regras>
<!-- REGRA DE NEGOCIO -->
```

```
<!-- ARMAZENAMENTO -->
<armazenamento>

  <banco_de_dados>
    <tabelas>
      <tabela>
        <id></id>
        <descricao></descricao>
        <campos>
          <campo>
            <nome><nome>
            <tipo></tipo>
            <tamanho></tamanho>
            <descricao></descricao>
            <null></null>
            <primary_key></primary_key>
            <foreign_keys>
              <foreign_keys>
                <tabela></tabela>
                <campo></campo>
              </foreign_keys>
            </foreign_keys>
          <campo>
        </campos>
      </tabela>
    </tabelas>
  </banco_de_dados>
```

```
  <arquivos>
    <arquivo>
      <nome></nome>
      <mime_type></mime_type>
      <endereco></endereco>
      <descricao></descricao>
    </arquivo>
  </arquivos>

</armazenamento>
<!-- ARMAZENAMENTO -->

<!-- TESTES -->
<teste>

  <objetivo_do_teste></objetivo_do_teste>

  <plano_de_teste>
    <contextualizacao>
      <principal_funcao_do_software></principal_funcao_do_software>
      <listagem_funcionalidades></listagem_funcionalidades>
    </contextualizacao>
    <casos_de_teste>
      <caso_de_teste>
          <nome>
        <descricao></descricao>
        <fluxo></fluxo>
```

```
        <entrada></entrada>
        <saida_esperada></saida_esperada>
        <estorias>
            <funcionalidade>
                <cenario></cenario>  <!--Caso ilimitados de cenários-->
            <funcionalidade>
        </estorias>
      </caso_de_teste>
    </casos_de_teste>
  </plano_de_teste>

  <projeto_de_teste></projeto_de_teste>

  <relatorio_de_teste>
      <erros>
          <erro>
              <categoria></categoria >
              <resumo></resumo>
              <descricao></descricao>
              <atribuido_a></atribuido_a>
          </erro>
      </erros>
  </relatorio_de_teste>

</teste>
<!-- TESTES -->
```