MÁRCIA CRISTINA DA PONTE

MELHORIA DE PROCESSOS DE REQUISITOS E TESTES: APLICAÇÃO EM UMA INSTITUIÇÃO DE ENSINO

Monografia apresentada ao PECE – Programa de Educação Continuada em Engenharia da Escola Politécnica da Universidade de São Paulo como parte dos requisitos para conclusão do curso de MBA em Tecnologia de Software.

São Paulo

2014

MÁRCIA CRISTINA DA PONTE

MELHORIA DE PROCESSOS DE REQUISITOS E TESTES: APLICAÇÃO EM UMA INSTITUIÇÃO DE ENSINO.

Monografia apresentada ao PECE – Programa de Educação Continuada em Engenharia da Escola Politécnica da Universidade de São Paulo como parte dos requisitos para a conclusão do curso de MBA em Tecnologia de Software.

Área de Concentração: Tecnologia de Software

Orientador: Prof. Dr. Mauro de Mesquita Spinola

São Paulo

2014

DEDICATÓRIA

Dedico este trabalho a Deus sem o qual seria impossível chegar até aqui.

AGRADECIMENTOS

À Universidade de São Paulo (USP), à Escola Politécnica e ao Programa de Educação Continuada em Engenharia (PECE) que permitiram a realização desse sonho fornecendo todos os recursos possíveis para que eu pudesse desenvolver este trabalho e acreditaram em mim. À STI do Instituto de Química da USP pelo apoio e incentivo, especialmente do meu chefe Fabio Massami Yamamoto pelo apoio e compreensão.

Aos professores do PECE pela dedicação, atenção e por sempre estarem prontos a esclarecer minhas dúvidas, em especial a Profa. Dra. Selma Shin Shimizu Melnikoff e ao Prof. Dr. Kechi Hirama.

Ao meu orientador Prof. Mauro de Mesquita Spinola pela orientação na escolha de um tema importante para meu desenvolvimento acadêmico e profissional, por me direcionar na pesquisa quando eu não sabia onde começar, pela dedicação e compreensão, paciência e incentivo, apoio e principalmente por dividir comigo seu preciso tempo, conhecimento e experiência, muito obrigada.

A minha querida amiga e colega de trabalho Maria Inês Cardillo por revisar meu trabalho.

Aos meus colegas, amigos, irmão e mãezinha pela paciência, incentivo e por compreenderem tantas vezes a minha ausência junto a eles e ao Rildo, meu amor, que com carinho e paciência ficou ao meu lado nos momentos difíceis para que eu conseguisse chegar até aqui.

RESUMO

A engenharia de requisitos, através das atividades de elicitação e análise, especificação, validação de requisitos e gerenciamento de requisitos, melhora a comunicação entre clientes e desenvolvedores, garante a construção do produto certo de forma correta, permite detectar e corrigir problemas no início do projeto. Com base nesses benefícios a STI do IQUSP considerou justificável a sua implementação no processo de desenvolvimento que utilizava para melhorar a qualidade dos softwares e melhor atender às necessidades de seus clientes. Utilizando metodologia de quatro etapas: diagnóstico, revisão da literatura, desenvolvimento de novos processos e aplicação, o processo da STI foi estudado com base na engenharia de software, isso permitiu diagnosticar vários problemas e propor três novos processos: elicitação de requisitos, gestão de requisitos e validação do software que foram aplicados no projeto de um novo software para a Seção de Veículos do IQUSP. O resultado obtido, através do documento de requisitos de software elaborado, foi o melhor entendimento do software para o cliente e desenvolvedor; o cliente autorizou o seu desenvolvimento ciente da estrutura geral do software a ser construído e verificou que atendia suas necessidades. Os procedimentos para validação do software também foram definidos. A área de desenvolvimento ficou mais organizada e este padrão será aplicado aos novos projetos de software o que ajuda a reduzir o retrabalho de correção de problemas por falta de entendimento entre as partes, auxilia na manutenção dos produtos construídos além de agregar qualidade.

ABSTRACT

The requirements engineering, through the activities of elicitation and analysis, specification, and validation of requirements and management requirements, improves the communication between clients and developers. It guarantees the construction of the right product in the right way, and also allows you to detect and correct problems early in the project. Based on these benefits STI of IQUSP considered its implementation justifiable in the development process, which was used to improve software quality and to better meet the needs of its customers. Using the methodology of four steps: diagnosis, literature review, development and implementation of new processes, the process of STI was studied based on software engineering. This enabled us to diagnose various problems and to propose three new processes: requirements for elicitation, management requirements, and validation of the software. These were then applied in the design of the new software for the Vehicle Section IQUSP. As a result, through the elaborated software requirements document, the understanding of the software became clearer to the client and the developer. The customer authorized the development of the software, and was aware of the general structure of the software to be built and verified that it met his needs. The procedures for validating the software have also been defined. The development area also became more organized and as a result this standard will be applied to new software projects which will help to reduce the correction of problems due to lack of understanding between the parties. This new standard will also assist in the maintainability of products built as well as add quality.

LISTA DE ILUSTRAÇÕES

Pág.

Figura 1: Processo de engenharia de requisitos em uma espiral (SOMMERVILLE, 2011) 25
Figura 2: Notação Diagrama de Caso de Uso 29
Figura 3: Processo de engenharia de requisitos da STI 43
Figura 4: Processo de elicitação de requisitos. 53
Figura 5: Processo de gestão de requisitos 54
Figura 6: Processo de validação de software 55
Figura 7: Diagrama de casos de uso 68
Figura 8: Tela após ator executar o evento iniciador do caso de uso gerar solicitação de serviço. 79
Figura 9: Tela após execução do passo 1 do caso de uso 7, gerar solicitação de serviço. 81
Figura 10: Tela referente ao passo 2 do caso de uso 7, gerar solicitação de serviço. 82
Figura 11: Tela referente aos passos 3, 4 e 5 do caso de uso 7, gerar solicitação de serviço. 83
Figura 12: Formulário de serviços (exemplo) 115
Figura 13: Formulário de viagem (exemplo) 116

LISTA DE TABELAS

Pág.

Tabela 1: Detalhamento de caso de uso (PRESSMAN, 2011) 30
Tabela 2: Estrutura de um documento de requisitos (SOMMERVILLE, 2011) 31
Tabela 3: Informações sobre o software a ser construído (PRESSMAN, 2011) 57
Tabela 4: Requisitos funcionais 59
Tabela 5: Característica e subcaracterísticas dos requisitos de qualidade 62
Tabela 6: Requisitos não funcionais (escopo, valor e métrica). 63
Tabela 7: Listagem de Casos de Uso (PAULA FILHO, 2009) 65
Tabela 8: Descrição dos Atores (PAULA FILHO, 2009) 66
Tabela 9: Características dos usuários representados pelos atores (PAULA FILHO, 2009) 66
Tabela 10: Detalhamento caso de uso: gestão de administradores (PRESSMAN, 2011) 69
Tabela 11: Detalhamento caso de uso: gestão de áreas (PRESSMAN, 2011) 70
Tabela 12: Detalhamento caso de uso: gestão de usuários (PRESSMAN, 2011) 71
Tabela 13: Detalhamento caso de uso: gerar solicitação de serviço (PRESSMAN, 2011) 72
Tabela 14: Detalhamento caso de uso: gestão de serviços pendentes (PRESSMAN, 2011) 73
Tabela 15: Detalhamento caso de uso: gerar solicitação de viagem (PRESSMAN, 2011) 74
Tabela 16: Detalhamento caso de uso: gestão de viagens pendentes (PRESSMAN, 2011) 75
Tabela 17: Detalhamento caso de uso: emissão de relatórios (PRESSMAN, 2011). 77
Tabela 18: Requisitos não funcionais, propriedades (PAULA FILHO, 2009). 78
Tabela 19: Rastreabilidade Casos de uso x Requisitos não funcionais (SAYÃO & LEITE, 2005) 85
Tabela 20: Rastreabilidade Ator x Casos de uso (SAYÃO & LEITE, 2005) 86
Tabela 21: Casos de Uso (PAULA FILHO, 2009) 112
Tabela 22: Descrição dos atores (PAULA FILHO, 2009). 113

Tabela 23: Características dos usuários representados pelos atores (PAULA FILHO, 2009). 113

LISTA DE ABREVIATURAS E SIGLAS

STI	Seção Técnica de Informática
IQUSP	Instituto de Química da Universidade de São Paulo
IP	Internet Protocol

SUMÁRIO

Pág.

1. INTRODUÇÃO 15

1.1 Motivação 15

1.2 Objetivo 16

1.3 Justificativa 16

1.4 Metodologias de Pesquisa 17

1.5 Estruturas do Trabalho 17

2. FUNDAMENTAÇÃO TEÓRICA 19

2.1 Engenharia de requisitos 19

2.2 Requisitos de software 21

2.2.1 Requisitos funcionais 22

2.2.2 Requisitos não funcionais 22

2.3 Processos de engenharia de requisitos 24

2.3.1 Estudo da viabilidade do software 26

2.3.2 Elicitação e análise de requisitos 26

2.3.3 Verificação de requisitos e casos de uso 27

2.3.4 Especificação de requisitos 27

2.3.5 Técnica de casos de uso 28

2.3.6 Documentação de requisitos 30

2.3.7 Validação de requisitos 32

2.4 Gestão de requisitos 34

2.5 Validação do software 35

2.5.1 Planejamento dos testes do software 35

2.5.2 Execução e registro de testes 37

2.6 Considerações finais 37

3. DIAGNÓSTICO 38

3.1 Contexto 38

3.2 Processo de desenvolvimento de software da STI 40

3.3 Diagnóstico 42

3.3.1 Estudo da viabilidade do software 44
3.3.2 Elicitação e análise de requisitos 44
3.3.3 Verificação de requisitos e casos de uso 44
3.3.4 Especificação de requisitos 44
3.3.5 Técnica de casos de uso 45
3.3.6 Documentação de requisitos 45
3.3.7 Validação de requisitos 45
3.3.8 Gerenciamento de requisitos 45
3.3.9 Validação do software 46

4. PROCESSOS DEFINIDOS 47

4.1 Processo de elicitação de requisitos 47

4.2. Processo de gestão de requisitos 49

4.3. Processo de validação do software 50

5. APLICAÇÃO DOS PROCESSOS DEFINIDOS 56

5.1 Estudo de viabilidade 56

5.2 Processo de elicitação de requisitos 56

5.2.1 1ª. Etapa: Identificar requisitos funcionais e não funcionais 56
5.2.2 2ª. Etapa: Especificar requisitos funcionais 65
5.2.3 3ª. Etapa: Especificar requisitos não funcionais 78
5.2.4 4ª. Etapa: Construir protótipo 79
5.2.5 5ª. Etapa: Validar requisitos 84

5.3 Processos de gerência de requisitos 84

5.4 Processos de validação do software 87

5.4.1 Planejamento de testes 87
5.4.2 Execução e registro 88

5.5. Avaliação dos Resultados 89

5.6 Considerações sobre a aplicação 90

6. CONSIDERAÇÕES FINAIS 91

6.1 Conclusões 91

6.2 Contribuições do Trabalho 92

6.3 Trabalhos Futuros 92

REFERÊNCIAS 93

GLOSSÁRIO 94

APÊNDICE A – Caso de teste – Caso de uso gerar solicitação de serviço 95

ANEXO A - NBR ISSO/IEC 9126-1 – Engenharia de software – Qualidade de produto – Pare 1: Modelo de qualidade – Modelo de qualidade para qualidade externa e interna 99

ANEXO B – Lista de questões sugeridas para verificação de requisitos 110

ANEXO C - Lista de questões para verificação de casos de uso 111

ANEXO D – Tabelas de casos de uso 112

ANEXO E – Formulários utilizados pela Seção de Veículos 115

1. INTRODUÇÃO

A engenharia de requisitos tem como finalidades a constituição de um documento com requisitos de software, que servirá de referência para seu desenvolvimento após aprovação do cliente e a gestão de requisitos de forma controlada quando alterações se fizerem necessárias. A especificação de requisitos serve também para a validação do software, ou seja, para que o usuário verifique se o software foi desenvolvido dentro das suas expectativas e necessidades.

Um processo de elicitação de requisitos permite a detecção e correção de incoerências e inconformidades no início do projeto, o que pode reduzir os custos e o tempo gasto com correções, pois é mais fácil, mais barato e menos crítico corrigir os problemas quando nada ainda foi codificado.

1.1 Motivação

A STI opera com uma equipe de dois analistas de sistemas e quatro técnicos de rede, sendo responsável por manter o funcionamento da rede computacional do IQUSP e seus principais serviços, desenvolvimento e manutenção de software e atendimento técnico de microinformática aos usuários.

Segundo seus analistas, responsáveis pelo desenvolvimento de software, o processo de desenvolvimento de software apresentava os seguintes problemas que demandam solução, e motivaram esse trabalho:

1. Ausência de padrão para a elicitação, aprovação e documentação de requisitos.
2. Dificuldade para identificar as reais necessidades do usuário.
3. Falta de formalização do aceite pelo usuário.
4. Falta de formalização para alterações nos requisitos.

1.2 Objetivo

O objetivo central deste trabalho é melhorar o processo de desenvolvimento de novos softwares da STI, implementado técnicas de engenharia de requisitos e de validação de software, buscando assim melhor atender às necessidades do cliente.

Após trabalho do processo de desenvolvimento de software da STI, foi proposto o desenvolvimento de novos processos baseados no estudo dos principais conceitos de engenharia de requisitos de software apresentados na seção 2, a implementar:

- Um novo padrão para a elicitação e verificação dos requisitos do software[1].
- O desenvolvimento de protótipos para validação e aceitação dos requisitos pelo cliente.
- O gerenciamento de requisitos para o controle de mudança de requisitos.
- Desenvolvimento de um planejamento de testes do software que permita a validação do software e a aceitação do cliente após o seu desenvolvimento.

1.3 Justificativa

A melhoria dos processos de requisitos e validação têm grande impacto sobre os serviços entregues ao cliente. Este trabalho priorizou esses dois processo com o objetivo de melhorar a qualidade em atividades com maior contato com o cliente e, por conseguinte, maior relevância para ele.

A aplicação deste trabalho se realizou em torno do processo de desenvolvimento de software da STI de um novo software para a Seção de Veículos do IQUSP, a ser executado pela STI, embora não envolvesse a construção do código do software em si. Também engloba testes do software para obter aceitação formal do software produzido pelo cliente.

Considerando que as implementações acima propostas contribuíram para evitar redundâncias de soluções, reduziram o retrabalho sendo que melhorou o

[1] A definição do termo encontra-se no glossário.

entendimento do software e das necessidades do cliente, aperfeiçoaram a qualidade do desenvolvimento, facilitando consideravelmente a tarefa de manutenção de software, consideramos que estas implementações são justificadas.

1.4 Metodologias de Pesquisa

A metodologia para o desenvolvimento deste trabalho envolveu quatro passos fundamentais descritos a seguir:

- Revisão da literatura. Foi realizado estudo de literatura voltada para os tópicos de requisitos, validação e planejamento de testes permitindo a obtenção de conhecimentos necessários para propor uma solução aos problemas apresentados pelos analistas.

- Diagnóstico. Realização de entrevistas com os analistas de softwares de uma instituição de ensino superior, que permitiu diagnosticar alguns problemas no processo de desenvolvimento de software utilizado na área.

- Desenvolvimento de novos processos. Esta solução consistiu no desenvolvimento de novos processos de elicitação de requisitos, gestão de requisitos e validação do software para os novos softwares, e no planejamento de teste de software para obtenção da aceitação final do sistema pelo cliente.

- Aplicação. A solução foi aplicada parcialmente em um novo software solicitado pelos clientes da área, permitindo a avaliação dos resultados.

1.5 Estruturas do Trabalho

O Capítulo 1 INTRODUÇÃO apresenta no contexto, o problema que se deseja solucionar, o objetivo e as motivações que estimularam o desenvolvimento desse trabalho, as justificativas consideradas relevantes e a estrutura do trabalho. Também apresenta a metodologia de pesquisa escolhida para o desenvolvimento deste

trabalho em quatro passos: diagnóstico, revisão da literatura, desenvolvimento de novos processos e aplicação.

O Capítulo 2 FUNDAMENTAÇÃO TEÓRICA apresenta conceitos necessários para o entendimento deste trabalho, dentre eles: engenharia de requisitos, tipos e classificação de requisitos, documentação de requisitos de software e sua estrutura, discorrendo sobre as principais atividades de um processo de engenharia de requisitos, gerenciamento de requisitos e validação de sistema.

O Capítulo 3 DIAGNÓSTICO apresenta um estudo do processo de desenvolvimento de softwares utilizado pela STI do IQUSP, como um todo, que identifica e documenta o contexto geral do ambiente e os problemas inerentes, com base nos conceitos de engenharia de requisitos.

O Capítulo 4 PROCESSOS DEFINIDOS apresenta três novos processos, desenvolvidos com base na fundamentação teórica, para melhorar o processo de desenvolvimento da STI, sendo: processo de elicitação de requisitos, processo de gestão de requisitos e processo de validação do sistema.

O Capítulo 5 APLICAÇÃO descreve a aplicação dos processos em um novo software solicitado a STI, avaliação dos resultados obtidos e algumas conclusões finais.

O Capítulo 6 CONSIDERAÇÕES FINAIS apresenta as considerações finais obtidas pelo autor, ao término deste trabalho, qualificando como uma melhor contribuição às próximas etapas profissionais, bem como subsidiando a criação de projetos.

2. FUNDAMENTAÇÃO TEÓRICA

Este trabalho foi desenvolvido principalmente com base no modelo de gestão de requisitos proposto por Ian Sommerville (2011). Esse modelo é largamente reconhecido na comunidade de engenharia de software. Assim, foi utilizado como um guia, fornecendo as diretrizes para formulação dos processos, Além desse modelo essencial, o trabalho se utilizou de outras contribuições importantes, que serviram para complementar conceitos, ideias, sugestões e técnicas. Entre estes, destacam-se PRESSMAN (2011), Kotonya & Sommerville (2002), Leffingwell & Widrig (2003), Paula Filho (2009) e Jennifer, Rogers & Sharp (2005).

Este capítulo contribuirá para melhor compreensão do presente trabalho, conceitos sobre engenharia de requisitos, o que são e quais seus tipos, as principais atividades que compõem o processo, técnicas para auxiliar na elicitação e especificação de requisitos, uma técnica de validação de requisitos. Também aborda os temas: gestão de requisitos - sua importância e como ela pode ser feita, e validação do sistema - sua finalidade, estratégia de validação adotada neste trabalho e tipos de testes.

2.1 Engenharia de requisitos

Segundo Sommerville (2011), a engenharia de requisitos é "o processo de descobrir, analisar, documentar e verificar serviços e restrições..." e faz parte do âmbito da disciplina de engenharia de software ministrada pelo PECE. Kotonya & Sommerville (2002) consideram a gestão de requisitos como parte do processo. Estas são atividades de alto nível que têm como finalidade analisar a viabilidade do software, elicitar os requisitos e descrevê-los de uma forma padrão, completa e consistente para serem compreendidos pelos envolvidos[2]. Para Pressman (2011), eles devem ser descritos de modo que permitam sua validação, isto é, deve ser possível checar se o produto atenderá às necessidades do cliente, compreender seus impactos no negócio e a interação dos usuários com ele. Já Leffingwell &

[2] Envolvidos são os clientes, usuários, membros que formam a equipe que desenvolverá o projeto e qualquer um que seja direta um indiretamente afetado pelo software.

Widrig (2003) define que a engenharia de requisitos é um processo de elicitar de forma sistemática, organizar e documentar os requisitos para um software complexo.

Antes de iniciar o desenvolvimento do software é importante que todos os envolvidos no projeto tenham o mesmo entendimento, ou seja, tenham a mesma compreensão dos requisitos do problema e do software a ser construído. Esta é considerada uma das atividades mais difíceis de serem realizadas pelo engenheiro de software por Pressman (2011) e um grande desafio sob a ótica de Leffingwell & Widrig (2003).

A elicitação de requisitos é uma atividade crítica que deve ser realizada no início do projeto para elicitar, revisar, corrigir e remover os requisitos desnecessários ou ambíguos, resolver conflitos de entendimento entre os envolvidos quanto aos requisitos, estabelecer as prioridades e definir um conjunto de requisitos estáveis, possíveis de ser implementados, testáveis e aceitos por todos os envolvidos. Erros nos requisitos são comuns custam caro para consertar (LEFFINGWELL & WIDRIG, 2003), por isso precisam ser identificados antes do desenvolvimento do software. Esses requisitos serão usados como base para nortear o desenvolvimento do produto correto (SOMMERVILLE, 2011)

No entanto, segundo Sommerville (2011) e Pressman (2011), apesar da busca pelos requisitos estáveis, uma característica peculiar é que eles podem sofrer alterações em qualquer fase do projeto e essas mudanças podem impactar no projeto como um todo, custar caro para o cliente e até provocar a inutilização do software. Então, ao se estabelecer um conjunto de requisitos estáveis, também se deve implementar um processo de gerenciamento de mudanças de requisitos, principalmente em softwares complexos. O sucesso do software com qualidade depende de um gerenciamento de requisitos eficaz (LEFFINGWELL & WIDRIG, 2003).

2.2 Requisitos de software

De acordo com Sommerville (2011) e Kotonya & Sommerville (2002), requisito de um software é a descrição do que ele deve fazer e dos serviços que deve oferecer, também deve descrever as limitações do seu funcionamento além de exprimir às necessidades do cliente.

Os requisitos podem ser descritos como requisitos de usuário ou requisitos de software (SOMMERVILLE, 2011) conforme detalhado a seguir:

Requisitos de usuário expressam às necessidades do cliente em alto nível utilizando linguagem natural e diagramas. Pode ser a descrição de um serviço que o software deve prover e/ou restringir, uma visão de negócio etc. Não se define a solução ou como o software irá operar (SOMMERVILLE, 2011)

Em Kotonya & Sommerville (2002) e Pressman (2011), requisitos de software detalha uma determinada função, serviço ou restrição operacional do software, de modo que o cliente possa entender e validar o que o software fará.

A descrição dos requisitos de diferentes formas permite transmitir informações do software a ser construído aos diferentes tipos de leitores (SOMMERVILLE, 2011) Por exemplo: gerentes podem estar interessados em saber como o serviço oferecido pelo software apoiará o seu processo de negócio, desenvolvedores precisam entender o funcionamento detalhado do software e suas operações para construir o software corretamente, os usuários finais precisam conhecer o seu funcionamento para operá-lo, etc.

Os requisitos de software podem ser classificados em requisitos funcionais e não funcionais e são dependentes mútuos; por isso um requisito pode gerar restrições a outros requisitos (SOMMERVILLE, 2011).

2.2.1 Requisitos funcionais

Os requisitos funcionais devem ser descritos de forma completa e consistente para assegurar que o produto entregue esteja correto, isto é, todos os serviços solicitados pelo cliente devem ser detalhadamente descritos, sem contradições ou ambiguidades. Dependendo do software que se pretende produzir, usuários e abordagem geral adotada pela empresa, os requisitos podem ser descritos em diferentes níveis de detalhamento (SOMMERVILLE, 2011)

Pode ser uma descrição detalhada de como o software deve proceder mediante a entrada ou saída específica ou em situação determinada, o que ele deve ou não fazer, o detalhamento de uma função, detalhamento de um serviço do software etc. Também podem ser descritos de uma forma geral, como requisitos de usuário, para que os clientes possam compreender o que software fará e recursos que proverá (SOMMERVILLE, 2011)..

2.2.2 Requisitos não funcionais

Requisitos não funcionais normalmente são aqueles aplicados ao software como um todo (SOMMERVILLE, 2011) e abrangem principalmente os requisitos de desempenho e atributos de qualidade (PAULA FILHO, 2009).

Os requisitos não funcionais devem ser identificados e descritos de maneira a evitar interpretações diversificadas, de forma quantitativa para que seja possível medi-los e submetê-los a testes de aceitação (SOMMERVILLE, 2011)

Cabe ressaltar que o custo para testar requisitos não funcionais pode ser elevado e, por isso, a empresa pode considerar injustificável tal investimento. Uma atenção especial deve se dada aos requisitos não funcionais, pois, o não atendimento de um requisito pode ocasionar a inutilização do software (SOMMERVILLE, 2011)

Os requisitos não funcionais são obtidos a partir dos requisitos de usuário e software e podem ser classificados assim (SOMMERVILLE, 2011):

- Requisitos de produto: Definem ou restringem o comportamento do software.

- Requisitos organizacionais: São os requisitos gerais provenientes de políticas e procedimentos da empresa ou desenvolvedores.

- Requisitos externos: Originados de fatores externos ao software como é o caso dos requisitos definidos por um órgão regulador ou leis, sem o qual o software não terá autorização para operar.

Os requisitos não funcionais de qualidade podem ser organizados com base no modelo de qualidade de software proposto pela NBR ISO/IEC 9126 (2003). Esta norma foca na qualidade do produto de software e se enquadra no modelo de qualidade das normas da família 9000. A qualidade aqui mencionada é aquela observada e percebida pelo usuário.

A norma NBR ISO/IEC 9126 (2003) organiza os atributos de qualidade em seis características (funcionalidade, confiabilidade, usabilidade, eficiência, manutenibilidade e portabilidade) que são subdivididas em subcaracterísticas conforme abaixo (mais detalhes são apresentados no Anexo A).

- Funcionalidade: adequação, acurácia, interoperabilidade, segurança de acesso, conformidade à funcionalidade.

- Confiabilidade: maturidade, tolerância a falhas, recuperabilidade, conformidade, conformidade à confiabilidade.

- Usabilidade: inteligibilidade, apreensibilidade, operacionalidade, atratividade, conformidade relacionada à usabilidade.

- Eficiência: comportamento em relação ao tempo, utilização de recursos, conformidade relacionada à eficiência.

- Manutenibilidade: analisabilidade, modificabilidade, estabilidade, testabilidade, conformidade à manutenibilidade.

- Portabilidade: adaptabilidade, capacidade para ser instalada, coexistência, capacidade para substituir, conformidade relacionada à portabilidade.

Os requisitos de desempenho são estáticos ou dinâmicos (PAULA FILHO, 2009).

- Estáticos: são os requisitos que desconsideram o fator temporal como, por exemplo: o número de sessões permitidas por usuário deve ser de, no máximo quatro.

- Dinâmicos: são os requisitos que trabalham com o tempo, por exemplo, após a digitação de usuário[3] e senha o software deve apresentar a tela inicial em menos de quatro segundos.

Requisitos de persistência são aqueles cujo valor continua a existir após a execução do software, normalmente dados armazenados em meio físico (PAULA FILHO, 2009). Por exemplo, dados que precisam ser consultados antes que uma operação seja realizada.

Requisitos técnicos são aqueles provenientes do cliente ou de autoridades externas (PAULA FILHO, 2009) como, por exemplo, o software deve ser implementado em PHP com base de dados MySql, leis etc.

2.3 Processos de engenharia de requisitos

O processo de engenharia de requisitos é iterativo e possui quatro atividades de alto nível: estudo de viabilidade, elicitação e análise, especificação e validação. Essas atividades são organizadas em espiral como na figura 1, que mostra a intercalação entre elas, cujo produto final é um documento de requisitos de software (SOMMERVILLE, 2011).

[3] Usuário neste caso trata-se do nome de acesso ao software de uma determina pessoa ou do próprio software.

Observe que a espiral é divida em três partes e o estudo de viabilidade é realizado logo no início, para verificar se o software solicitado vai resolver o problema do cliente. Em seguida, é feita a especificação de requisitos de negócio também em linguagem natural.

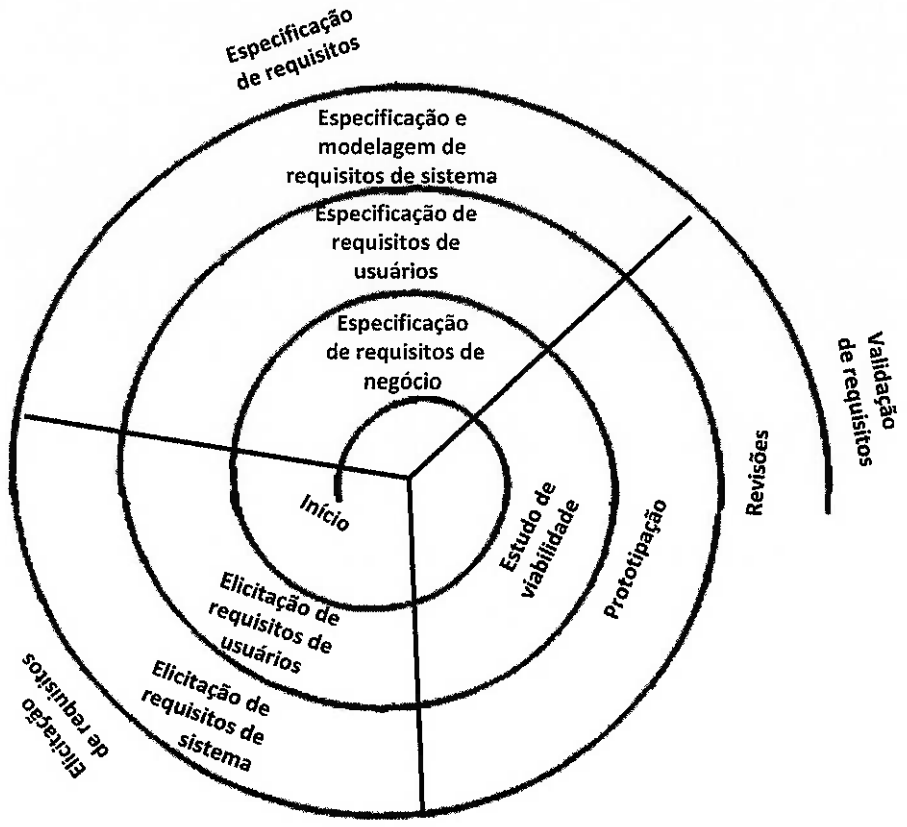

Figura 1: Processo de engenharia de requisitos em uma espiral (SOMMERVILLE, 2011).

Na sequencia, é feita a elicitação de requisitos de usuários e a especificações dos requisitos em linguagem natural. Também é feita a prototipação para que o usuário possa validar se os requisitos solicitados estão presentes.

Depois, a elicitação e especificação dos requisitos do software que são realizadas para descrever de forma detalhada o que o software realmente fará.

Na figura 1 os requisitos são organizados em diferentes níveis de detalhamento e cada requisito poderá passar pela espiral quantas vezes for necessário até que esteja completamente claro (SOMMERVILLE, 2011).

2.3.1 Estudo da viabilidade do software

É preciso estudar a viabilidade do desenvolvimento do software junto aos clientes, para conhecer o problema que ele deseja solucionar. É possível que a solução não seja o desenvolvimento de um software. Neste ponto o comprometimento entre os envolvidos é essencial para melhor compreensão dos requisitos de negócios, que precisam ser identificados e relacionados. Podem-se utilizar, no caso, recursos como reuniões, onde perguntas iniciais são feitas aos clientes e documentadas para análise de viabilidade (PRESSMAN, 2011).

2.3.2 Elicitação e análise de requisitos

Confirmando que o software precisa ser desenvolvido, inicia-se o processo de elicitação e análise de requisitos como propõe Sommerville (2011), onde são executadas as atividades de identificar, classificar, priorizar e especificar os requisitos. Neste ponto também é relevante o envolvimento de todas as pessoas que irão interagir com o software a ser construído, para que se obtenha o melhor entendimento possível; é necessário reunir informações que apresentem seu perfil. A seguir são descritas as atividades do processo de elicitação e análise de requisitos (SOMMERVILLE, 2011):

- Elicitar: relacionar, através de entrevistas ou reuniões, o máximo possível de informações sobre os softwares existentes, o que se deseja para o novo software solicitado, o ambiente em que deverá operar e suas restrições e identificar os requisitos. Essas informações podem ser obtidas através de documentos, domínio em que o software vai operar softwares em funcionamento e qualquer pessoa que seja afetada de alguma forma pelo software, ou seja, os envolvidos.

- Classificar: classificar os requisitos como requisitos de usuários e requisitos de software (funcionais e não funcionais).

- Priorizar: identificar juntamente com os envolvidos os requisitos de maior prioridade que sejam possíveis de serem atendidos (Pode-se utilizar: baixa, média, alta ou outro padrão).

- Especificar: detalhar no documento de requisitos de software os requisitos de software funcionais e não funcionais.

2.3.3 Verificação de requisitos e casos de uso

Uma verificação de requisitos e casos de uso pode ser realizada pelo desenvolvedor para localizar erros e problemas antes que o projeto avance demais e o investimento para fazer as correções seja muito alto. Ela pode melhorar significativamente a qualidade do software, mitigar a possibilidade de erros insignicantes tornanem-se grandes dores de cabeça e pode ser aplicada no código, requisitos, dados de teste etc. Quando nenhuma verificação é feita, a correção de erros pode ficar, além de mais cara, difícil e demorada (PRESSMAN, 2011).

Pressman (2011) sugere que a verificação seja feita no início da elicitação de requisitos e que cada requisito seja examinado com base num conjunto de questões pré-definidas, respondidas para cada requisito conforme eles vão sendo descobertos. O intuito é identificar inconsistências, incoerências e erros. Algumas questões sugeridas por ele são apresentadas nos Anexos B e C.

2.3.4 Especificação de requisitos

A especificação dos requisitos dever ser documentada de forma que auxilie os envolvidos a compreender o software, mesmo que os requisitos ainda estejam incompletos. É possível que, nesse ponto, novos requisitos surjam (SOMMERVILLE, 2011).

Os requisitos devem ser elicitados e descritos de modo que possam ser medidos ou quantificados. É necessário já visualizar como e quais testes serão realizados, quais as métricas poderão ser utilizadas para testar cada requisito, preparando-se assim para criar um planejamento de testes (PAULA FILHO, 2009).

Quando não for possível ou muito difícil a composição de testes, é preciso reconsiderar o requisito, pois pode ser inviável, ou seja, talvez ele seja impossível de ser implementado (PAULA FILHO, 2009). É preferível que isso seja descoberto no início do projeto, quando o investimento de recursos é ainda reduzido, para fazer as correções (SOMMERVILLE, 2011).

2.3.5 Técnica de casos de uso

A modelagem de casos de uso é uma técnica de descoberta de requisitos que pode ser usada para representar os requisitos de usuário em alto nível assim como, os requisitos funcionais, onde cada requisito funcional pode ser representado por um caso de uso. Cada caso de uso detalha passo a passo como será a interação de um usuário com o software em circunstâncias específicas (SOMMERVILLE, 2011)

A modelagem dos requisitos funcionais com casos de uso deve representar todas as funções do software, atores e relacionamentos de forma completa, detalhada e permitir gerar algum benefício para o cliente após sua execução (SOMMERVILLE, 2011)

Os seguintes passos sugeridos por Paula Filho (2009), podem ser utilizados para auxiliar o processo de identificação de: casos de uso, atores, identificação de características de usuários, representação gráfica dos relacionamentos entre os casos de uso e atores num Diagrama de Casos de Uso e o seu detalhamento.

a) Identificar todos os casos de uso com base nos requisitos de software que se pretende construir definindo: nome e descrição resumida vide exemplo Anexo D.

b) Identificar todos os atores, considerar que cada tipo de ator representa um grupo de usuários e que ator é toda pessoa, hardware ou software que irá interagir com o software. Para cada tipo de ator defina um nome e uma descrição sucinta vide exemplo Anexo D.

c) Descrever as características dos usuários que são representadas pelos atores, vide exemplo Anexo D.

d) Representar os relacionamentos que indicam a existência de comunicação entre os casos de uso e os casos de uso e atores, gerando o Diagrama de Casos de Uso utilizando a notação proposta na figura 2:

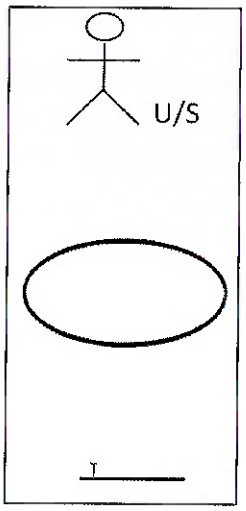

Figura 2: Notação Diagrama de Caso de Uso

Representação do ator que é qualquer pessoa (U) ou software (S) envolvido nas interações.

Cada caso de uso dever ser único, existindo um cenário para cada um e representado por uma elipse. Um caso de uso pode estar relacionado a um ou mais atores.

As interações serão representadas por uma linha contínua.

e) Identificar cenários para cada caso de uso

Cada caso de uso consiste de um fluxo de eventos cenário principal (fluxo de eventos principal, normalmente aquele que é mais utilizado). Este cenário principal pode ter variações dependo da entrada fornecida pelo seu ator, nesse caso são chamados de cenários alternativos ou fluxo alternativo. Também há os cenários que podem surgir quando algo não funciona bem, são denominados exceções. É necessário identificar todos os caminhos possíveis para cada caso de uso.

f) Detalhar cada caso de uso identificando pelo menos os campos apresentados na tabela 1 a seguir.

Tabela 1: Detalhamento de caso de uso (PRESSMAN, 2011)

Nome	Nome do caso de uso
Descrição	Descrição resumida do caso de uso
Ator(es)	Pessoas, hardware ou softwares que fará a interação com o software.
Pré-condições	Para a execução do caso de uso em questão, identificar quais condições precisa estar satisfeitas.
Evento iniciador	Evento que iniciará a execução do caso de uso.
Fluxo de Evento	Descrição do fluxo principal do caso de uso, o passo a passo, caminho mais comum ou o caminho feliz, trata-se do comportamento do caso de uso. Sua escrita deve ser simples e sem termos técnicos para atender os fins deste trabalho, uma vez que os softwares desenvolvidos são de pequeno porte.
Pós-Condições	Condição esperada após a execução do caso de uso gera resultado que proporcione algum benefício ao usuário. Trata-se de um elemento importante para a elaboração dos testes.
Fluxo alternativo	Após a definição do caminho mais comum representado no fluxo de evento e atendida as pré-condições, o fluxo alternativo é uma questão de convenção para se tratar e detalhar as situações menos comuns.
Extensões	Comportamento adicional (como uma exceção ou opcional) que seja mais complexo que um fluxo alternativo.

g) Definir casos de teste

Os casos de teste são construídos para exercitarem as estruturas do software; espera-se que, para cada entrada especifica, se obtenha saídas específicas. Seu objetivo é garantir que todas as funcionalidades tenham sido devidamente atendidas. Para cada caso de uso, normalmente, é possível gerar um conjunto de casos de teste que podem ajudar a localizar erros.

2.3.6 Documentação de requisitos

Agora que os requisitos e a técnica de casos de uso foram apresentados e que o processo de engenharia de requisitos já é conhecido, as informações coletadas junto aos envolvidos podem ser relacionadas e organizadas em um documento que pode ser denominado "Documento de Requisitos de Software" (SOMMERVILLE, 2011).

Este documento tem como finalidade formalizar a comunicação com o cliente, detalhar os requisitos para os desenvolvedores e testadores. Definindo o que os desenvolvedores devem implementar no software, os requisitos de usuários e os requisitos de software (SOMMERVILLE, 2011).

Como esse documento é utilizado para leitura de diversas pessoas tais como clientes, desenvolvedores e testadores, portanto, ele deve estar atualizado com os registros de todas as alterações de requisitos e versões identificadas. Seu uso pode evitar ações restritivas por parte dos desenvolvedores que poderiam inviabilizar o software impedindo-o de evoluir, além de evitar o esquecimento dos requisitos que afetam o software de forma geral ou a interpretação incorreta das necessidades do cliente. Também pode ser usado para facilitar o trabalho dos profissionais que precisam fazer manutenção no software. Enfim, muitas são as utilidades do Documento de Requisitos de Software (SOMMERVILLE, 2011).

No entanto, o detalhamento e a precisão deste documento dependerão da criticidade do software a ser desenvolvido (SOMMERVILLE, 2011). A tabela 2 apresenta uma possível estrutura que pode ser seguida para a sua criação e adaptada se necessário.

Tabela 2: Estrutura de um documento de requisitos (SOMMERVILLE, 2011)

Capítulo	**Descrição**
Prefácio	Deve definir os possíveis leitores do documento e descrever seu histórico de versões, incluindo uma justificativa para a criação e um resumo das mudanças feitas em cada versão;
Introdução	Deve descrever a necessidade para o software. Deve descrever brevemente as funções do software e explicar como ele vai funcionar com outros softwares. Também deve descrever como o software atende aos objetivos globais de negócio ou estratégicos da organização que encomendou o software.
Glossário	Deve definir os termos técnicos usados no documento. Não se devem fazer suposições sobre a experiência ou o conhecimento do leitor.
Definição de requisitos de usuário	Deve descrever os serviços fornecidos ao usuário. Os requisitos não funcionais de software também devem ser descritos nessa seção. Essa descrição pode usar a linguagem natural, diagramas ou outras notações compreensíveis para os

Capítulo	Descrição
	clientes, normas de produtos e processos que devem ser seguidas devem ser especificadas.
Arquitetura dos softwares	Deve apresentar uma visão geral em alto nível da arquitetura do software previsto, mostrando a distribuição de funções entre os módulos do software. Componentes de arquitetura que são reusados devem ser destacados.
Especificação de requisitos do software	Deve descrever em detalhes os requisitos funcionais e não funcionais. Se necessário, também podem ser adicionados mais detalhes aos requisitos não funcionais. Interfaces com outros softwares podem ser definidas.
Modelos do software	Pode incluir modelos gráficos do software que mostram os relacionamentos entre os componentes do software, o software e seu ambiente. Exemplos de possíveis modelos são modelos de objetos, fluxo de dados ou modelos semânticos de dados.
Evolução do software	Deve descrever os pressupostos fundamentais em que o software se baseia, bem como quaisquer mudanças previstas, em decorrência da evolução de hardware, de mudanças nas necessidades do usuário etc. Essa seção é útil para projetistas de softwares, pois pode ajudá-los a evitar decisões capazes de restringir possíveis mudanças futuras no software.
Apêndices	Deve fornecer informações detalhadas e específicas relacionadas à aplicação em desenvolvimento, além de descrições de hardware e banco de dados, por exemplo. Os requisitos de hardware definem as configurações mínimas ideais para o software. Requisitos de banco de dados definem a organização lógica dos dados usados pelo software e os relacionamentos entre esses dados.
Índices	Vários índices podem ser incluídos no documento. Pode haver, além de um índice alfabético normal, um índice de diagramas, de funções, entre outros pertinentes.

2.3.7 Validação de requisitos

A validação de requisitos é uma atividade feita para verificar se os requisitos definidos para o novo software estão consistentes, completos e precisos (KOTONYA & SOMMERVILLE, 2002) com as necessidades do cliente. Deve ser realizada antes que o software seja encaminhado para o desenvolvimento e com o que o cliente espera receber, além de, permitir a localização e correção de problemas ainda na fase inicial. Problemas encontrados no projeto, antes que ele seja encaminhado para o desenvolvimento ou entre em produção, poderão ser corrigidos rapidamente evitando o retrabalho que aumentaria consideravelmente o seu custo (PRESSMAN, 2011).

A validação de requisitos se inicia depois que os requisitos foram estabelecidos. Pode consistir no desenvolvimento de um protótipo que permita a interação dos envolvidos com o software desejado. Em que o cliente pode testar, explorar e verificar se o software atende suas necessidades, além de, esclarecer dúvidas quanto ao entendimento do que o produto irá oferecer, detectar problemas ou novos requisitos antes que qualquer linha de código seja construída além de facilitar a comunicação entre os envolvidos (JENNIFER, ROGERS, & SHARP, 2005).

Para Paula Filho (2009), a aplicação dessa técnica auxilia na elicitação de requisitos, pode tornar o processo mais rápido com a colaboração dos interessados e deve focar nos pontos menos compreendidos. O importante na construção de um protótipo é a rapidez com que ele é produzido (PAULA FILHO, 2009) e em quão fácil é modificá-lo (JENNIFER, ROGERS, & SHARP, 2005). Assim, o ideal é que se escolha o meio mais conveniente para desenvolver o protótipo do projeto.

Jennifer, Rogers, & Sharp (2005) diz que o protótipo pode ser de baixa-fidelidade ou de alta-fidelidade. O protótipo de baixa-fidelidade é simples, barato, rápido de fazer, auxilia na descoberta de requisitos e é descartado ao final. Já o de alta-fidelidade é mais elaborado e, por isso, é mais demorado para construir e mais caro e ao final, espera-se que ele seja utilizado.

Protótipo de baixa-fidelidade pode ser construído utilizando programas específicos de prototipagem que permitam a navegação entre as telas, desenvolvido em HTML com hiperlinks; também pode ser desenhado em fichas de modo que os envolvidos possam ordenar da melhor maneira, ou obtido através de editores de textos com ferramentas que produzam relatórios etc. O de alta-finalidade é construído utilizando ferramentas de software mais completas e poderosas (JENNIFER, ROGERS, & SHARP, 2005).

Um protótipo deve permitir a simulação de um cenário em que o usuário possa realizar testes e apontar os problemas rapidamente. Um protótipo muito elaborado e cheio de recursos visuais pode ser considerado pelo cliente como parte da solução final; portanto é importante esclarecer qual a finalidade do mesmo, de modo que o cliente esteja de acordo com o descarte (JENNIFER, ROGERS, & SHARP, 2005).

2.4 Gestão de requisitos

Os requisitos de software podem sofrer alterações ao longo de sua vida, ou novos requisitos podem surgir em qualquer fase do projeto. Por isso, a gestão de requisitos deve ser feita durante de todo o processo de desenvolvimento para analisar, documentar e implementar a mudança ou inclusão do novo requisito quando esta for justificada (PRESSMAN, 2011).

A gerência de requisitos está associada à qualidade do software produzido e é baseada na rastreabilidade de requisitos, uma vez que, qualquer alteração deverá ser analisada para verificar sua viabilidade. A rastreabilidade permite apresentar ao cliente que os requisitos solicitados e aprovados por ele, foram atendidos pelo software e/ou identificar inconsistências e problemas entre requisitos, planos e artefatos (SAYÃO & LEITE, 2005).

Para apoiar o processo de gestão de requisitos, na fase inicial do projeto é preciso definir um padrão de rastreabilidade para o software a ser produzido assim como o que deverá ser rastreado (SAYÃO & LEITE, 2005) em comum acordo entre cliente e desenvolvedor.

O rastreamento de requisitos é uma técnica que provê os "relacionamentos entre requisitos, arquitetura e implementação final do software" (SAYÃO & LEITE, 2005). Ela deve permitir localizar todos os itens que precisam ser alterados para que uma mudança ou a inclusão de um novo requisito possa ser efetivada. Artefatos, documentos, diagramas, código, testes de aceitação para a validação do software e protótipos em HTML podem ser utilizados para rastreabilidade.

Basicamente, o processo de gerenciamento de requisitos consiste nas atividades de análise de problema e especificação de mudanças, análise de mudanças e custo e na implementação de mudança (SOMMERVILLE, 2011) e deve iniciar a partir de primeira versão do Documento de Requisitos de Software aprovada. Tem-se a seguir a descrição das atividades:

- Análise de problema e especificação de mudanças: identificar o problema ou mudança nos requisitos atuais e a alteração a ser feita nos requisitos do software ou novo requisito a ser implementado.
- Análise de mudanças e custo: analisar os impactos e riscos nos requisitos do software no projeto como um todo, considerando variáveis como: prazo, esforço, custo, recursos materiais e riscos para o projeto. Isso se aplica também quando a solicitação é para inclusão de um novo requisito.
- Implementação de mudança: se for justificada a mudança, elas devem ser realizadas em todo o projeto.

Qualquer mudança no ambiente técnico ou de negócio que afete os requisitos de software, como alteração de prioridades de negócio, inclusão de novos equipamentos que necessitem interagir com o software e alterações nas leis e outros, devem ser formalizados (SOMMERVILLE, 2011).

2.5 Validação do software

A validação do software consiste na adoção de alguma estratégia que pode ser, por exemplo, a elaboração de um planejamento dos testes com desenvolvimento de casos de teste, sua execução, registro e análise dos dados resultantes (PRESSMAN, 2011). Seu objetivo é permitir que o usuário execute testes no software e em suas funcionalidades para encontrar defeitos e validar se os requisitos definidos foram atendidos. Lembre-se que testes podem ajudar a localizar erros, o que não garante que o software esteja livre deles se nenhum erro for localizado (SOMMERVILLE, 2011).

2.5.1 Planejamento dos testes do software

O planejamento de testes do software deve ser desenvolvido pela equipe de software e com o consentimento do cliente, antes que qualquer codificação seja feita, para que, seja possível verificar se os testes são possíveis de serem concebidos, realizados e aceitos. Seu objetivo é a obter a aceitação formal do software pelo cliente (SOMMERVILLE, 2011)

Os testes devem ser executados, após a conclusão do desenvolvimento do software ou partes possíveis de testar. Podem ser realizado em ambiente de desenvolvimento, no local de trabalho ou em ambos, conforme o que for acordado com o cliente. Sendo que, os dados e resultados obtidos devem ser registrados pelo desenvolvedor (SOMMERVILLE, 2011)

Os testes envolvem o fornecimento de dados pelos clientes, assim como opiniões referentes ao teste de software. Os testes de usuário são fundamentais, pois, devido à sua experiência, conhecimento e influência no ambiente de trabalho, ele pode contribuir para aumentar a confiabilidade, desempenho, usabilidade e robustez do produto criado. O usuário pode fazer testes que jamais poderão ser realizados em um ambiente de testes criado pelo desenvolvedor (SOMMERVILLE, 2011).

Os testes de usuário podem ser do tipo alpha, beta ou de aceitação abaixo descritos (SOMMERVILLE, 2011):

- Teste alpha: Os usuários do software realizam testes na área de desenvolvimento, testando o software conforme ele vai sendo desenvolvido, identificando erros e problemas de uso, fornecendo informações e sugestões que não são visíveis aos desenvolvedores.
- Teste beta: Os testes são realizados pelos usuários de forma livre em uma versão do software para avaliação. Normalmente seleciona-se um grupo de pessoas que serão os primeiros usuários do software. Problemas de interação do software e o ambiente são comuns de serem detectados nesse tipo de teste.
- Teste de aceitação: Trata-se de um teste formal de aceite pelo cliente e, para sua realização, é necessário: definir os critérios de aceitação; planejar testes de aceitação; derivar testes de aceitação; executar testes de aceitação; negociar resultados de teste e rejeitar/aceitar o software.

O aceite formal do software pelo cliente se dará após as correções dos defeitos por ele apontados nos testes realizados e após comunicar por escrito que o software cumpriu os seus propósitos, atendendo aos requisitos estabelecidos e que pode permanecer em produção.

2.5.2 Execução e registro de testes

Baseado em Sommerville (2011) a execução dos testes pode envolver três etapas, na primeira o usuário irá seguir uma sequência de testes pré-definidos pelo desenvolvedor (ex.: casos de teste) no teste alpha, que servirá para testar as funcionalidades do software e validar os requisitos. O teste será realizado em ambiente de desenvolvimento e acompanhado pelo desenvolvedor que anotará os defeitos para posterior correção

Na segunda, o teste beta pode ser realizado no local de trabalho do usuário, mas apenas quando o software ou parte dele estiver finalizado, como se fosse um treinamento para um pequeno grupo de usuários. Os erros deverão ser reportados ao desenvolvedor para posterior correção.

A quantidade de usuários que executarão os testes alpha e beta dependerá do software.

A terceira etapa consistirá num teste de aceitação que consistirá na disponibilidade do software no local de trabalho dos usuários, para o maior número possível de usuários, de modo que possam realizar testes com dados reais. Todos os defeitos devem ser reportados ao desenvolvedor em intervalos regulares por meio de preenchimento de formulário a ser definido de acordo com o software.

2.6 Considerações finais

O estudo dos conceitos de engenharia de requisitos, gestão de requisitos e validação do software desta seção, permitiu: documentar o processo de desenvolvimento da STI, gerar um diagnóstico mais completo dos problemas existentes no processo com base nesses conceitos; proporcionar conhecimentos necessários para que novos processos sejam desenvolvidos e aplicados para a melhoria do processo, satisfação do cliente e organização da STI.

3. DIAGNÓSTICO

Este capítulo apresenta o processo de desenvolvimento do software anteriormente utilizado pela STI, o contexto geral no qual a Seção está inserida e os serviços que oferece. Após o entendimento deste processo, foi gerado o diagnóstico que permitiu identificar problemas no processo e o desenvolvimento de uma possível solução baseada na fundamentação teórica deste trabalho.

3.1 Contexto

A STI do IQUSP é responsável por manter o funcionamento da rede computacional para que a comunidade IQUSP realize seus trabalhos administrativos e acadêmicos. Também é responsável pelo desenvolvimento de softwares para área administrativa e atendimento técnico de microinformática. É formada por uma equipe de dois analistas de sistemas, quatro técnicos de informática e esporadicamente contrata estagiários com o objetivo de obter auxilio em trabalhos específicos da área de informática.

Inicialmente as solicitações de desenvolvimento de software à STI eram para a construção de softwares simples e, por isso, atendida prontamente. Com o passar dos anos observou-se a necessidade de desenvolver softwares com mais qualidade e melhorar a compreensão das necessidades dos clientes. Assim sendo, os analistas perceberam a necessidade de melhorar o processo de desenvolvimento de software para atender a essas necessidades, resolvendo os problemas por eles apontados:

1. Ausência de padrão para a elicitação, aprovação e documentação de requisitos.
2. Dificuldade para identificar as reais necessidades do usuário.
3. Falta de formalização do aceite pelo usuário.
4. Falta de formalização para alterações nos requisitos.

Aos analistas de sistemas cabem as seguintes tarefas: instalar, configurar, administrar e monitorar equipamentos de rede e servidores; manter funcionamento e segurança da rede computacional e dos principais serviços que atendem a comunidade IQUSP4; gerir o serviço de suporte técnico de rede e microinformática para área acadêmica e administrativa; realizar o serviço de desenvolvimento de software para automatizar as tarefas da área administrativa.

Aos técnicos de rede cabe a execução de tarefas de suporte técnico de microinformática e rede aos usuários tais como: manutenção de hardware; instalação de softwares operacionais e aplicativos; instalação e conexão de pontos de rede.

As tarefas dos analistas são priorizadas de acordo com sua criticidade5; portanto, o serviço de desenvolvimento de software deve ser realizado de modo que não prejudique as tarefas mais criticas.

A rede computacional do IQUSP tem aproximadamente 2500 pontos de rede, 90 switches gerenciáveis, 150 pontos de acesso sem fio e 18 servidores com software operacional Linux sendo: servidores de backup, espelho Linux Debian, servidor de arquivos, registro de logs, DNS, DHCP, VPN, serviço centralizado de antivírus, listas, servidores de softwares internos e externos desenvolvidos pela STI, HTTP/HTTPS (páginas de docentes e seções), TFTP, monitoramento de rede e firewall/gateway.

Neste trabalho foi detalhada apenas a configuração dos dois servidores de softwares, que hospedam os softwares de uso interno e externo desenvolvidos pela STI.

O servidor de softwares internos hospeda todos os softwares que tem a finalidade de auxiliar na administração das atividades internas executadas pela STI tais como: atendimento de chamados técnicos aos usuários, controle e cadastro de

[4] Comunidade IQUSP: são os docentes, funcionários e alunos.
[5] É considerada tarefa crítica qualquer tarefa que, quando não executada, pode interromper do funcionamento da rede e serviços para a comunidade IQ.

equipamentos para acesso a rede, controle de uso de IP, administração de contas de e-mails, administração de páginas etc.

O servidor externo hospeda os softwares desenvolvidos para uso das seções e setores da área administrativa tais como: software de telefonia, software de reserva de salas de aula e reuniões, softwares de ocorrências etc.

Encontra-se a seguir, a configuração de hardware e software utilizados por estes servidores atualmente:

Servidor de softwares internos e externos: Power Edge R620, CPU Intel Inside Xeon, 68GHZ de RAM e 600GB de HD. Software Operacional Linux Distribuição Debian, Apache, MySql e PHP.

O parque computacional do IQUSP tem cerca de 2000 equipamentos entre computadores, notebooks, impressoras, tablets, palms, celulares, nobreaks, telefones VOIP e aparelhos de videoconferência. A maioria dos computadores e notebooks opera com sistema operacional Microsoft Windows em diversas versões.

O quadro de servidores do IQUSP é formado por 115 docentes e 267 funcionários administrativos e o corpo discente por 1088 alunos, aos quais a STI presta algum tipo de serviço.

3.2 Processo de desenvolvimento de software da STI

Como dito anteriormente, o processo de desenvolvimento de software é realizado por dois analistas de sistemas na área de informática do IQUSP, em ambiente Linux, distribuição Debian, linguagem PHP, Banco de Dados MySQL, CSS e HTML, ambiente esse que deverá ser mantido.

Os softwares desenvolvidos pela STI, em sua maioria, são de pequeno porte e destinados ao uso dos servidores6 e discentes7. Nos softwares desenvolvidos é

[6] Servidores: funcionários docentes
[7] Discentes: alunos de graduação e pós-graduação

comum existir a necessidade de se estabelecer comunicação com outros softwares desenvolvidos e em produção, a fim de obter informações atualizadas e/ou autenticação. Eles são disponibilizados para acesso via web (intranet e Internet) através de qualquer navegador e dependendo da necessidade há restrições de quem e de quais computadores e redes poderão utilizá-los.

Para cada solicitação de um novo software o seguinte processo era realizado:

- Solicitar novo software: O cliente solicita verbalmente o desenvolvimento do software ao chefe da STI.

- Analisar viabilidade: O chefe da STI analisa a viabilidade com base na experiência adquirida ao longo dos anos; se confirmada a viabilidade, o software é incluído na rotina da STI, caso contrário a chefia deve informar inviabilidade para o cliente.

- Designar analista: Confirmada a viabilidade, o chefe da STI designa um analista para realizar projeto.

- Coletar informações e esclarecer dúvidas: O analista responsável pelo projeto realiza uma ou mais reuniões com os usuários para coletar informações e identificar suas necessidades e expectativas além de esclarecer dúvidas. As anotações são feitas no caderno do analista, sendo que cada analista tem seu próprio caderno. Não havendo mais dúvidas o software é construído pelo analista. O desenvolvimento software em si, a sua codificação, não é abordada nesse trabalho.

- Disponibilizar software para teste: Após a conclusão da construção do software, o analista disponibilizava o software ou parte dele para que o usuário realize testes com o objetivo de detectar possíveis problemas.

- Testar software: Cliente testa o software e, se houver erros, estes eram reportados verbalmente ao analista para providenciar as correções.

- Finalizar: Após as correções, o analista disponibilizava o software para os usuários executarem novamente os testes. Quando não houver mais erros, o usuário autorizava verbalmente ou por e-mail colocar o software em produção.

- Reportar erros: Após as correções, o analista disponibilizava o software para os usuários executarem novamente os testes. Os erros eram novamente reportados pelos usuários ao analista para providenciar as correções.

Durante algumas semanas, o analista responsável pelo software acompanhava o seu funcionamento em busca de eventuais problemas e erros que possam surgir.

A figura 3 é a representação gráfica do processo de desenvolvimento de novos softwares atualmente utilizado pela STI. Esta representação se faz necessária a fim de visualizar o processo como um todo. Observe que o desenvolvimento não é mencionado na figura, pois o foco deste trabalho são as atividades de maior contato com o cliente.

3.3 Diagnóstico

Apesar de não haver um processo formal de engenharia de requisitos para o desenvolvimento de software da STI, após algumas reuniões com os analistas foi possível obter informações para mapear o seu processo de desenvolvimento (seção 3.2).

Analisando o processo de engenharia de requisitos atual com base nos conceitos de engenharia de requisitos apresentados, foi possível elaborar um diagnóstico contendo mais alguns problemas, além daqueles apresentados pelos analistas.

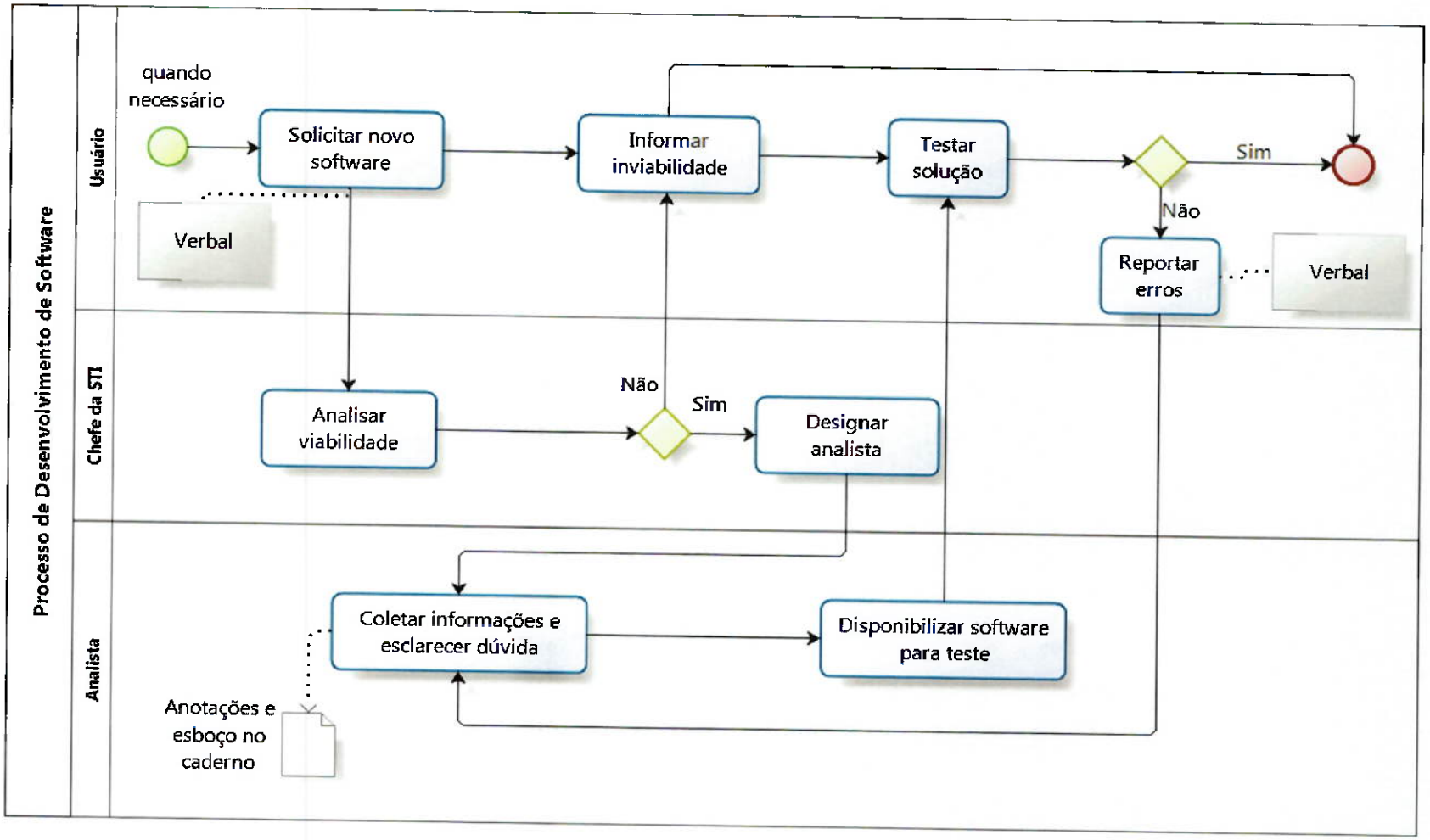

Figura 3: Processo de engenharia de requisitos da STI

3.3.1 Estudo da viabilidade do software

O estudo da viabilidade feito pelo chefe da STI que, tendo como base sua experiência no desenvolvimento de software, autoriza ou não o desenvolvimento do software. Não existe uma base de projetos anteriores, documentados de forma padronizada, para consulta.

3.3.2 Elicitação e análise de requisitos

Um dos problemas apontados pelos analistas é falta de padrão para a elicitação(1), aprovação e documentação de requisitos, aprovação e documentação de requisitos.

A ausência de um padrão nas atividades de requisitos, pode acarretar em dificuldade na compreensão do software tanto para o desenvolvedor como para o cliente, além de requisitos importantes ou críticos passarem despercebidos ou serem mal interpretados.

Além disso, a falta de documentação de requisitos associada à falta de aprovação dos requisitos pelo cliente podem ocasionar desentendimentos entre o que foi pedido e o que foi construído, além de, gerar conflitos entre os envolvidos, o quê pode culminar no insucesso do projeto.

3.3.3 Verificação de requisitos e casos de uso

Nenhuma técnica para verificação de requisitos ou caso de uso era utilizada.

3.3.4 Especificação de requisitos

Os requisitos não são especificados e documentados de modo detalhado e padronizado para o entendimento geral dos envolvidos.

A não produção de uma documentação de requisitos que descreva o software detalhadamente pode acarretar no esquecimento de requisitos importantes que devam ser implementados.

3.3.5 Técnica de casos de uso

Nenhuma técnica para detalhamento das funções e serviços, que serão oferecidas pelo software, é utilizada para documentar de forma padronizada para o entendimento geral dos envolvidos.

3.3.6 Documentação de requisitos

A documentação dos requisitos é feita informalmente e sem padrão no caderno do analista responsável pelo desenvolvimento e muitas vezes não é preservada.

3.3.7 Validação de requisitos

A dificuldade para identificar as reais necessidades do cliente, faz parte dos problemas apresentados pelos analistas, ou ainda a dificuldade do usuário em expô-las pode provocar o fracasso do projeto de software, caso o produto seja construído de modo errado ou o produto errado seja construído.

Não existia uma padronização para documentar requisitos de software e então apresentá-la ao usuário para aceite.

Não utilizava mecanismo padrão para apresentar ao cliente o protótipo de como será o software, de modo que pudesse verificar a presença dos requisitos solicitados e então obter seu aceite. Apenas rascunhos para representação de algumas telas eram usados quando necessário.

3.3.8 Gerenciamento de requisitos

Falta de formalização para alterações nos requisitos, faz parte dos problemas apresentados pelos analistas, pode dificultar o rastreamento das mudanças que foram realizadas porque as solicitações de alterações não eram documentadas de forma padronizada.

3.3.9 Validação do software

A ausência de formalidade de aceite por parte do usuário, um dos itens constantes nos problemas listados pelos analistas, se resumia no procedimento descrito a seguir: os testes eram realizados pelos usuários em ambiente real, após a construção do software; no entanto sem um planejamento de testes que fosse aprovado por ambos e que pudesse ser utilizado como uma aceitação formal pelo usuário, resumindo-se apenas a uma confirmação por e-mail ou verbal. A não formalização de um aceite pode gerar desentendimentos com relação ao que foi combinado e comprometer o projeto.

4. PROCESSOS DEFINIDOS

Foram desenvolvidos os processos de elicitação de requisitos, gestão de requisitos e validação do software, utilizando a ferramenta de modelagem Bizagi, conforme notação da metodologia de Notação de Modelagem de Processos de Negócio (BPMN) e a seção 2 deste trabalho.

4.1 Processo de elicitação de requisitos

Com base nos conceitos expostos, nas características dos produtos desenvolvidos pela STI e na quantidade de profissionais disponíveis para executar esta atividade, foi elaborada uma proposta de um novo padrão para o processo de elicitação de requisitos para o desenvolvimento de novos softwares a ser utilizado pela STI.

Esse processo de elicitação de requisitos deve ser executado, após estudo de viabilidade do software e parecer favorável do chefe da área ao seu desenvolvimento, pelo analista designado para o projeto. Ele foca nos requisitos do software com o intuito de agilizar o processo de elicitação, sendo que cada etapa deve ser registrada no Documento de Requisitos de Software ou equivalente.

1ª. Etapa: Identificar requisitos funcionais e não funcionais

a) Coletar dados: realizam-se reuniões para coleta de informações sobre as necessidades dos clientes, serviços e funções desejadas, materiais que possam auxiliar no entendimento do software desejado (documentos impressos, informações de outros softwares, restrições, nome de outras pessoas que precisam ser consultadas, necessidades e expectativas dos clientes etc.).

b) Definir requisitos funcionais

c) Definir requisitos não funcionais

2ª. Etapa: Especificar requisitos funcionais

Identificam-se os requisitos os funcionais: casos de uso para os requisitos funcionais (seção 2.2.1). As informações foram registradas na tabela 3.

a) Listar casos de uso: Com base nos requisitos funcionais, listam-se os casos de uso, os atores e as características de atores (seção 2.3.5).

b) Gerar diagrama de casos de uso: Para representar as relações entre casos de uso e atores e casos de uso entre si (seção 2.3.5).

c) Detalhar casos de uso: Para cada caso de uso faz-se o detalhamento conforme propõe a fundamentação teórica deste estudo (seção 2.3.5).

3ª. Etapa: Especificar requisitos não funcionais.

Identificam-se os requisitos não funcionais: descrição dos requisitos não funcionais, classificando-os de acordo com suas características e subcaracterísticas apresentadas na seção 2.2.2. As informações foram registradas na tabela 3.

4ª. Etapa: Construir protótipo

Elabora-se o protótipo em HTML que esteja disponível ao usuário de modo que ele possa navegar e conhecer o funcionamento e a estrutura do software que será construído (seção 3.3.7).

5ª. Etapa: Validar requisitos

Após a conclusão das etapas anteriores, todos os requisitos do software e o protótipo deverão ser apresentados ao cliente a fim de obter sua aprovação ou identificar possíveis problemas. Deve-se solicitar aprovação do cliente (seção 3.3.7).

No caso de identificação de problemas na especificação, esses deverão ser corrigidos imediatamente de modo que se adequem às necessidades do cliente.

A figura 4 representa o novo processo de elicitação de requisitos.

4.2. Processo de gestão de requisitos

Após a conclusão da primeira versão do Documento de Requisitos de Software e definição do que deve ser rastreado, ambos de comum acordo com o cliente, as solicitações de mudança ou inclusão de novos requisitos deverão obedecer quatro etapas do processo de gestão de requisitos.

1ª. Etapa: Solicitar alteração ou inclusão.

Qualquer alteração ou inclusão de requisitos deve ser feita formalmente através do preenchimento do "Formulário de solicitação ou inclusão de requisito"

Neste formulário deverão constar pelo menos os seguintes campos preenchidos: nome do solicitante, nome da chefia imediata, nome do software, data, área, telefone, e-mail, motivo da solicitação, descrição detalhada da mudança, justificativa, assinatura do solicitante, assinatura do responsável pela área solicitante, parecer do analista sobre a possibilidade da alteração (apenas para uso do analista como registro de conclusão) e data.

2ª. Etapa: Analisar solicitação

Cada solicitação deverá ser analisada com base na primeira versão aprovada do Documento de Requisitos de Software e nas matrizes de rastreabilidade, bem como, os riscos do ponto de vista da estrutura, riscos e impactos para a STI, e tempo disponível dos analistas para a execução.

3ª. Etapa: Comunicar inviabilidade

Se após análise da solicitação for constatado que a solicitação é inviável o cliente é comunicado.

4ª. Etapa: Implementar alteração

Se após análise da solicitação for constatado que a solicitação é viável a implementação das alterações devem ser registradas no Documento de Requisitos de Software. Mais especificamente, a atualização dos seguintes artefatos: matriz(es) de rastreabilidade, listagem de casos de uso, listagem dos atores, listagem das característica dos usuários, detalhamento de casos de uso, protótipo em HTML, casos de teste e especificação de requisitos não funcionais. A figura 5 representa o processo de gestão de requisitos.

4.3. Processo de validação do software

O processo de validação do software definido destina-se a obter a aceitação do cliente do produto construído. Este processo é formado pelas etapas seguintes que pretendem permitir ao usuário realizar testes, com o objetivo de identificar problemas e reportá-los ao desenvolvedor para correção.

1ª. Etapa: Gerar casos de teste alpha

O desenvolvedor cria um conjunto de casos de testes baseados nos casos de uso das funcionalidades que precisam ser testadas.

2ª. Etapa: Executar casos de teste

Os casos de teste serão executados pelo(s) cliente(s) pré-determinados, na área de desenvolvimento e com o acompanhamento do analista responsável pelo projeto.

3ª. Etapa: Corrigir

Se o(s) cliente(s) localizar(em) erros, os mesmos serão registrados e, se possível, corrigidos imediatamente. Caso contrário, o(s) cliente(s) confirma o funcionamento e o sucesso do teste aprovando a primeira versão do software. Passa-se para a etapa seguinte.

4ª. Etapa: Gerar teste beta

As configurações necessárias para o funcionamento do software em ambiente real são providenciadas, assim como a instalação no local de trabalho de alguns clientes pré-selecionados. O software é liberado por tempo determinado para execução dos testes.

5ª. Etapa: Executar teste beta

Os clientes executam testes no software em ambiente real com dados reais.

6ª. Etapa: Corrigir

Se o(s) cliente(s) localizar(em) erros, os mesmos serão registrados e, se possível, corrigidos imediatamente. Caso contrário, o(s) cliente(s) confirma o funcionamento e o sucesso do teste aprovando a primeira versão do software. Passa-se para a etapa seguinte.

7ª. Etapa: Gerar teste de aceitação

O software é disponibilizado para que grande parte dos usuários faça sua utilização alimentando-o com dados reais, a informação de que se trata de uma versão em fase de testes por tempo determinado deve acompanhar esta liberação.

8ª. Etapa: Executar teste de aceitação

Os clientes executam testes no software e erros são reportados ao analista responsável pelo projeto que providenciará as correções.

9ª. Etapa: Corrigir

Se o(s) cliente(s) localizar(em) erros, os mesmos serão registrados e, se possível, corrigidos imediatamente. Caso contrário, o(s) cliente(s) confirma o funcionamento e o sucesso do teste aprovando a primeira versão do software. Caso contrário, o cliente confirma o funcionamento e o sucesso do teste aprovando a segunda versão do software e autoriza colocá-lo em produção.

Para cada teste é acordado, entre o desenvolvedor e o cliente, um prazo determinado para a sua realização. A figura 6 representa o processo de elicitação de requisitos.

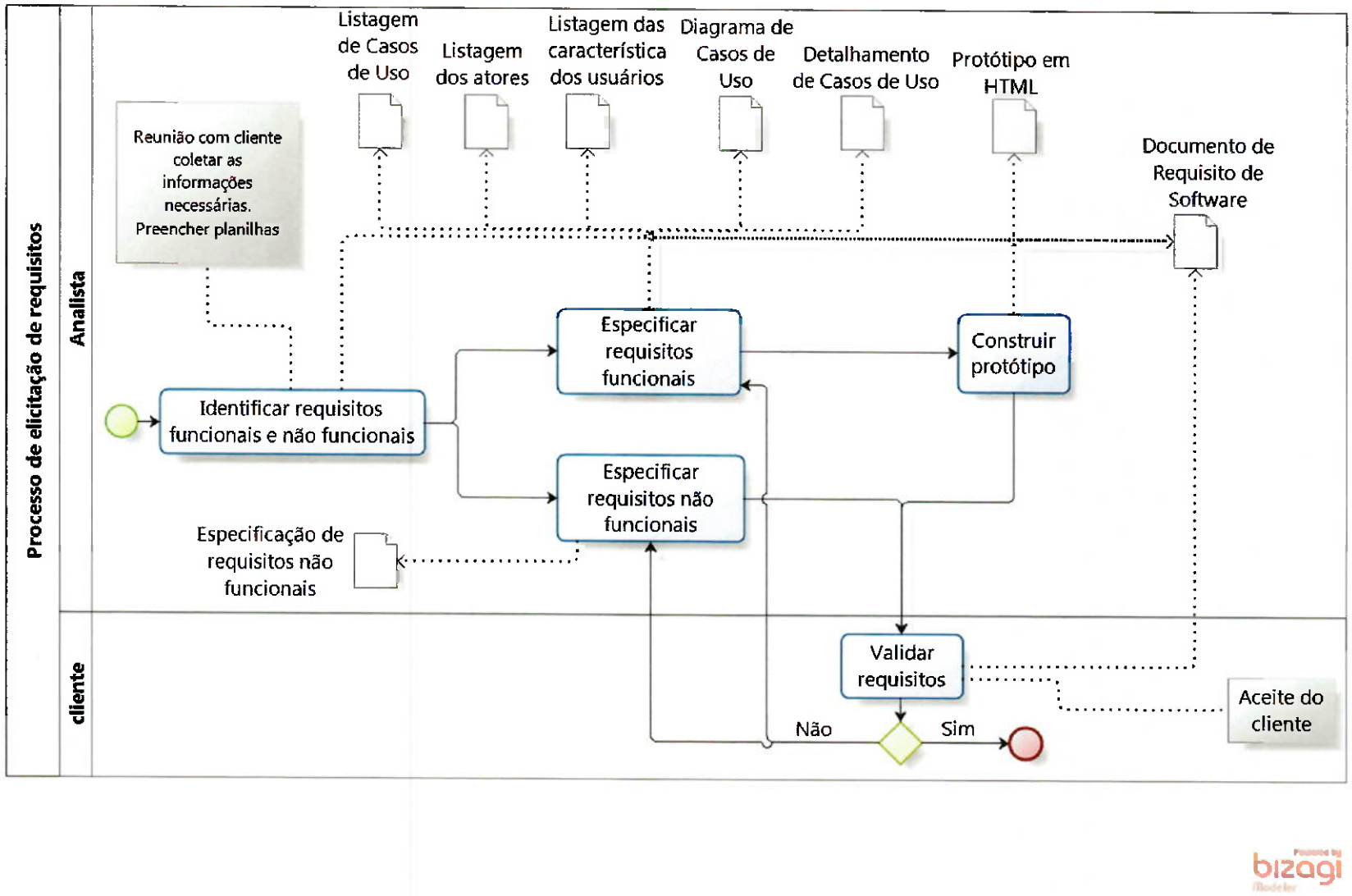

Figura 4: Processo de elicitação de requisitos.

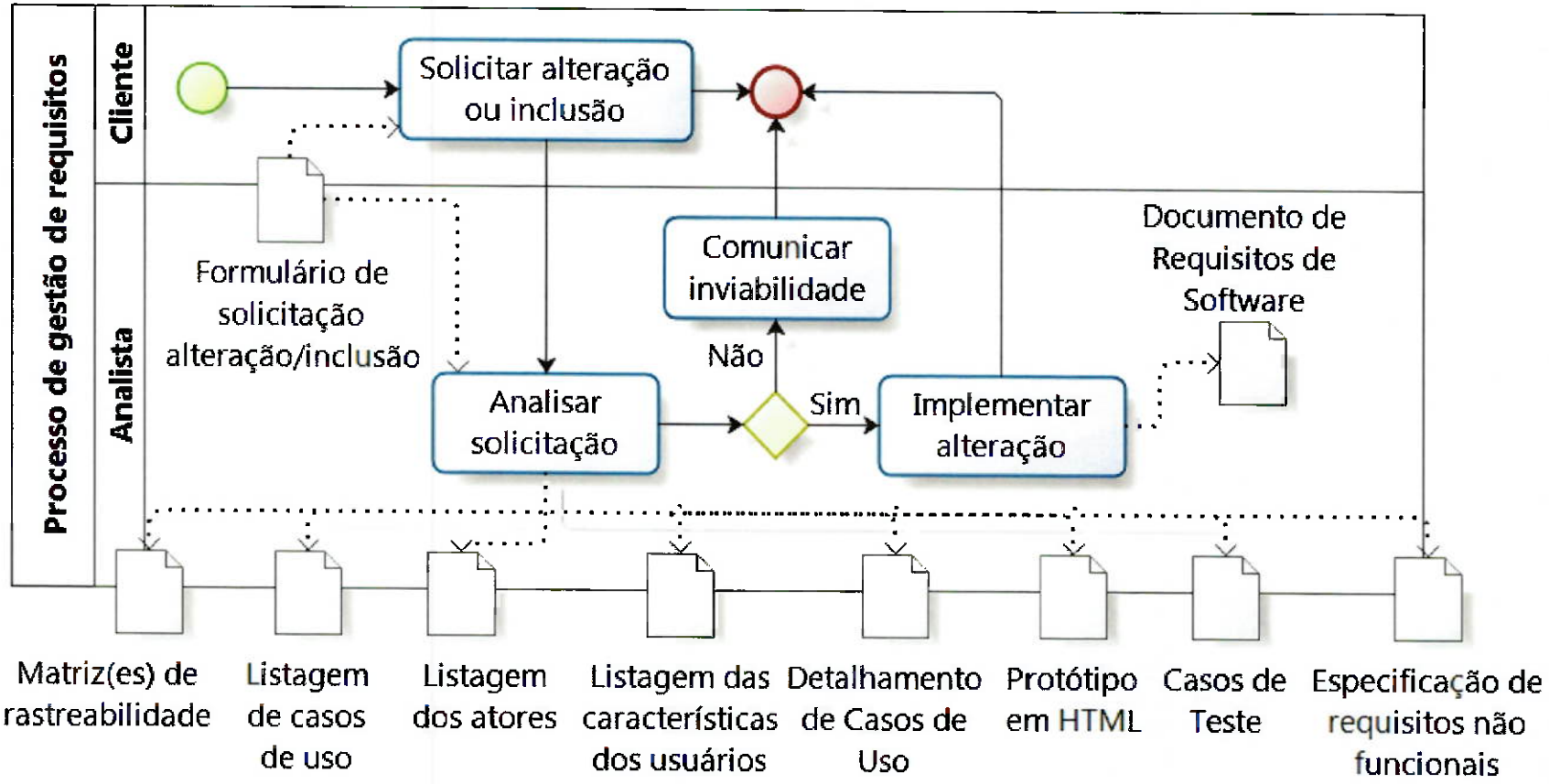

Figura 5: Processo de gestão de requisitos

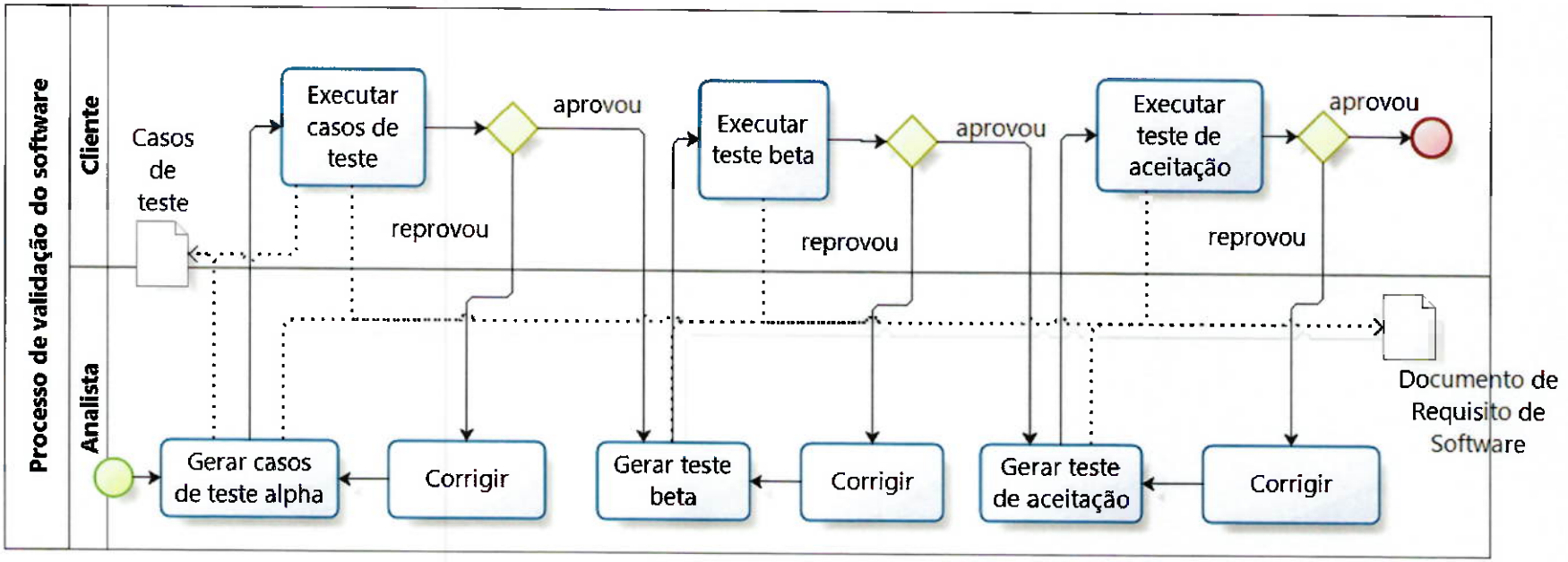

Figura 6: Processo de validação de software

5. APLICAÇÃO DOS PROCESSOS DEFINIDOS

Este capítulo descreve a aplicação dos conceitos e processos apresentados, no projeto de desenvolvimento de um software, solicitado pela área administrativa para melhor organizar e administrar as tarefas executadas pela Seção de Veículos do IQUSP, para facilitar a tomada de decisões administrativas e de negócio.

5.1 Estudo de viabilidade

O estudo da viabilidade foi realizado pela chefia da STI que concluiu que o desenvolvimento do software era necessário. Por se tratar de um software de pequeno porte, não crítico, possível de ser construído, o projeto pode se acomoda à rotina da STI sem prejuízo de outras atividades. Sendo assim, designou um analista para o desenvolvimento do mesmo.

5.2 Processo de elicitação de requisitos

O levantamento de requisitos realizado para o Software de Veículos foi realizado aplicando o processo criado na seção 4.1.

5.2.1 1ª Etapa: Identificar requisitos funcionais e não funcionais

a) Coletar dados

A Seção de Veículos presta serviços gerais de transporte e viagens e para organizar suas tarefas utiliza formulário (ANEXO E) impresso que é preenchido manualmente por seus funcionários. Quando o administrador geral solicita, relatórios são desenvolvidos pela chefia da Seção em editor de texto com base nos formulários preenchidos. Normalmente as solicitações de serviços são feitas pelo usuário, pessoalmente ou por telefone.

Após reuniões informais com o administrador geral e posteriormente com os funcionários da Seção, as informações necessárias para elaboração deste projeto de software foram obtidas e disponibilizadas na tabela 3.

Tabela 3: Informações sobre o software a ser construído (PRESSMAN, 2011)

Problema	Falta de precisão nas informações referentes às atividades realizadas na Seção de Veículos e por isso dificuldade na tomada de decisões administrativas e de negócio.
Solicitante	Administrador geral
Afeta	Administrador geral, motoristas e usuários finais.
Necessidades	Automatizar as solicitações de serviços e viagens com a criação de formulário. Implementar mecanismo que permita somente aos usuários cadastrados fazer solicitações de serviços e viagens. Listagens atualizadas com as solicitações de serviços e viagens pendentes, encerradas no mês e por período. Relatórios diversos de viagens e serviços: • Relatórios de serviços e viagens executados em determinado período, organizados por nome do solicitante e com a quantidade de serviço e viagens. • Relatório de serviços e viagens executados em determinado período, organizados por número do controle de tráfego e com a quantidade de serviço e viagens. • Relatório de serviços e viagens cancelados em determinado período, organizados por nome de solicitante e com a quantidade de serviço e viagens. Acesso em diferentes níveis para administradores, motoristas e usuários finais. Todos os acessos e operações no software devem gerar registros (logs). Permitir utilizar dados de uma solicitação encerrada para abrir nova solicitação de serviço ou viagem.

	Possibilidade criar novos relatórios e listagens a partir dos dados armazenados. Quando uma solicitação for aberta, encerrada ou alterada, o administrador, motorista e o solicitante devem ser informados por e-mail.
Objetivos	Precisa de informações atualizadas, organizadas e detalhadas sobre os serviços e viagens executados pela Seção para tomada de decisões administrativas e de negócio.
Escopo	O software permitirá: • Criar, encerrar e cancelar novas solicitações de serviços e viagens. • Emitir relatórios diversos. • Exibir listagens diversas. • Cadastrar usuário, local, área, tipo de serviço, grupos de usuários e cadastro de administradores do software.
Contexto	O software será utilizado pela Administração, funcionários da Seção de Veículos e por todos os usuários que desejam solicitar os serviços oferecidos por ela.
Limites	O software não proverá interface com área financeira. Os cadastros necessários ao funcionamento do software serão alimentados pelos administradores e não pela equipe de desenvolvimento da STI.
Restrições	O acesso ao software deverá ser feito por usuário que tenham e-mail IQUSP, de qualquer computador disponível na rede do IQUSP, através de navegadores como Internet Explorer, Firefox, Chromium e Mozilla, software operacional Windows 95 ou superior.

b) Definir requisitos funcionais

Os requisitos relacionados na tabela 4 consistem na identificação e descrição detalhada, em linguagem natural, de cada requisito e dos casos de teste que foram elaborados para serem utilizados na validação do software pelo cliente após a conclusão do desenvolvimento do software.

Tabela 4: Requisitos funcionais

Requisito (Caso de uso)	**Descrição detalhada dos requisitos**	**Teste**
Gestão de administradores	Controlar quem poderá administrar o software e o nível de acesso. Permite criação, alteração, ativação, desativação e o nível de acesso. Os registros não poderão ser removidos. Dados necessários: nome, e-mail, permissão.	Gerar caso de teste para criação, alteração, ativação e desativação de administradores.
Gestão de tipos de serviços	Considerar serviço todas das saídas de veículos para fazer serviços internos ou externos à USP que possam ser realizados em um dia (ex.: retirada de documentos, entrega de correspondência etc.). Permite criação de tipos de serviços que serão disponibilizados para novas solicitações de serviço, alteração, ativação e desativação. Os registros não poderão ser removidos. Os registros serão utilizados para as solicitações de serviços. Dados necessários: descrição do tipo de serviço.	Gerar caso de teste para criação, alteração, ativação e desativação de serviço.
Gestão de áreas	Considerar áreas atendidas que poderão solicitar serviços e viagens (seções, setores, departamentos etc.). Permite criação e alteração. Os registros não poderão ser removidos. Dados necessários: sigla e descrição.	Gerar caso de teste para criação, alteração de áreas.
Gestão de locais	Considerar locais físicos aos quais pertencem os solicitantes de serviços e viagens (Ex.: bloco 01). Permite criação e alteração. Os registros não poderão ser removidos. Dados necessários: descrição.	Gerar caso de teste para criação, alteração de local.
Gestão de grupos	Considerar grupos aos quais pertencem os solicitantes de serviços e viagens (Ex.: funcionários, docentes, estagiário, etc.). Permite criação e alteração Os registros não poderão ser removidos. Dados necessários: descrição.	Gerar caso de teste para criação, alteração de grupos.
Gestão de usuários	Controla os usuários que poderão acessar o software. Permite criação, alteração, ativação e desativação. Os usuários armazenados no software não poderão ser	Gerar caso de teste para criação, alteração, ativação e desativação

Requisito (Caso de uso)	Descrição detalhada dos requisitos	Teste
	excluídos. Dados necessários: nome, e-mail, telefone, bloco, área e grupo do usuário.	de usuário.
Gerar solicitação de serviço	Permite criação de uma solicitação de serviço. • Processamento para criar uma nova solicitação de serviço no software das 08h00 à 17h00, de segunda a sexta-feira. • O serviço deverá ser solicitado com 02 dias úteis de antecedência sendo necessário que o software verifique isso. • O serviço somente poderá ser solicitado pelos usuários cadastrados no software e com e-mail @iq.usp.br ativo no servidor de e-mail do IQUSP. • Dados necessários: nome, e-mail, grupo, telefone, bloco e área do solicitante, docente vinculado, nome do local, telefone, endereço e bairro cidade do destino, tipo de serviço, descrição do serviço (detalhamento do serviço que deverá ser feito), data para execução, hora para execução e observações.	Gerar caso de teste para solicitar novos serviços
Gestão de serviços pendentes	Permite edição de uma solicitação de serviço em aberto. Processamento para edição de uma solicitação de serviço em aberto • Designar ou alterar o motorista que irá realizar o serviço. • Encerrar uma solicitação de serviço preenchendo os dados: data da execução, hora da execução. • Cancelar a solicitação de serviço preenchendo os dados: motivo do cancelamento e data do cancelamento.	Gerar casos de teste para editar solicitações pendentes e executar designação de motorista, encerramento de solicitação e cancelamento.
Gerar solicitação de viagem	Permite criação de uma solicitação de viagem. Processamento para criar uma nova solicitação de viagem no software das 08h00 à 17h00, de segunda a sexta-feira. • A viagem deverá ser solicitada com 02 dias úteis de antecedência sendo necessário que o software verifique isso. • A viagem somente poderá ser solicitada pelos usuários cadastrados no software e com e-mail @iq.usp.br ativo no servidor de e-mail do IQUSP. • Dados necessários: nome, e-mail, grupo, telefone, bloco,	Gerar caso de teste para solicitar novas viagens.

Requisito (Caso de uso)	Descrição detalhada dos requisitos	Teste
	e área do solicitante, docente vinculado, nome do local, telefone, endereço, bairro, cidade e estado do destino, descrição/finalidade da viagem, data de saída, hora de saída, data de retorno, hora de retorno, informar por conta de quem serão as despesas.	
Gestão de viagens pendentes	Processamento para edição de uma solicitação de viagem em aberto • Editar dados referentes às despesas: valor estimado em reais, valor utilizado em reais e a totalização de cada um. • Editar dados referentes à quilometragem: quilometragem inicial, quilometragem final e a quilometragem percorrida que deve ser calculada no momento da exibição. • Designar ou alterar o motorista que irá realizar a viagem. • Encerrar uma solicitação de viagem preenchendo os dados: data da execução, hora da execução e número do controle de tráfego. • Cancelar a solicitação de viagem preenchendo os dados: motivo do cancelamento e data do cancelamento.	Gerar casos de teste para editar solicitações pendentes e executar edição de despesas, designação de motorista, encerramento de solicitação e cancelamento.
Emissão de relatórios	Gerar relatórios que possam ser impressos, obtendo dados armazenados na base do software, por período que será informado pelo administrador: • Pesquisa por data de execução, ordenada por nome de usuário, totalizando serviços e viagens. • Pesquisa por data de execução, ordenada por controle de tráfego, totalizando serviços e viagens. • Pesquisa as solicitações cancelada, por data de solicitação, ordenada por nome de usuário.	Gerar caso de teste de acordo com descrição: Relatório por nome de usuário com totais de serviços e viagens (por data de execução); Relatório por controle de tráfego (por data de execução); Relatório de solicitações canceladas (por data da solicitação).

c) Definir requisitos não funcionais

Para o preenchimento das características e subcaracterísticas dos requisitos de qualidade considere a tabela 5; mais informações podem ser obtidas no Anexo A.

Tabela 5: Característica e subcaracterísticas dos requisitos de qualidade

	Características					
	(F) funcionalidade	(C) confiabilidade	(U) usabilidade	(E) eficiência	(M) manutenibilidade	(P) portabilidade
Subcaracterísticas	- Adequação - Acurácia - Interoperabilidade - Segurança de acesso - Conformidade relacionada à funcionalidade	- Maturidade - Tolerância a falhas - Recuperabilidade - Conformidade relacionada à confiabilidade - Conformidade relacionada à usabilidade	- Inteligibilidade - Apreensibilidade - Operacionalidade - Atratividade - Conformidade relacionada à usabilidade	- Comportamento em relação ao tempo - Utilização de recursos - Conformidade relacionada à eficiência	- Analisabilidade - Modificabilidade - Estabilidade - Testabilidade - Conformidade relacionada à manutenibilidade	- Adaptabilidade - Capacidade para ser instalado - Coexistência - Capacidade para substituir - Conformidade relacionada à portabilidade

Detalhados na tabela 6 estão os requisitos não funcionais identificados, os casos de uso aos quais eles se relacionam e o valor que agregam ao software e os possíveis testes e métricas.

Os requisitos não funcionais 1 e 2 são mecanismos de segurança que deverão ser implementados para evitar acesso indevido; o requisito 3 deve testar cada campo digitado, antes que seja armazenado em banco de dados e, caso o usuário digite algo inconsistente, o software deve avisar e manter os dados na tela para que ele faça a correção evitando o retrabalho; o requisito 4 significa que o software deve ser desenvolvido de maneira simples, para que funcione corretamente nos navegadores mais utilizados pelos usuários.

Tabela 6: Requisitos não funcionais (escopo, valor e métrica).

Nº	**Requisito**	**Casos de uso**[8]	**Valor**	**Teste / métrica**
1	Acesso com senhas	Gestão de usuários Gestão de administradores	Restrição de acesso às funções por senha e grupo.	Tentar acesso com usuários não permitidos, senhas erradas. Testar o acesso para os diversos tipos de usuários, verificar se as funções corretas são exibidas e funcionam para cada tipo de usuários.
2	Restrição de acesso à rede IQUSP	Todos	Restrição de acesso	Testar acesso ao software de equipamento fora da rede IQUSP.

[8] Casos de uso aos quais se aplica.

N°	Requisito	Casos de uso[8]	Valor	Teste / métrica
3	Evitar erros de digitação	Gestão de áreas Gestão de locais Gestão de tipo de serviço Gestão de grupo Gestão de Usuário Gestão de administradores Gestão de viagens Gestão de serviços	Evitar erros de digitação por parte do usuário.	Verificar o tempo gasto que os usuários levam preencher e entender as mensagens de erros emitidas pelo software. No caso de erro, emitir mensagem de erro e o conteúdo digitado. Verificar se os dados foram armazenados corretamente.
4	Funcionar nos navegadores Internet Explorer e Mozilla	Todos	Compatibilidade com navegadores	Relacionar navegadores onde o software funcionou corretamente.

5.2.2 2ª Etapa: Especificar requisitos funcionais

Para a especificação de requisitos deste trabalho, na modelagem dos requisitos funcionais, a técnica de casos de uso foi aplicada para detalhar cada requisito funcional.

Nesta seção, é criado um índice para enumerar os casos de uso, sendo que cada requisito funcional corresponderá a um caso de uso. Também são identificados os atores e suas características.

a) Listar casos de uso

Após análise dos requisitos funcionais, foram elencados na tabela 7 os casos de uso.

Tabela 7: Listagem de Casos de Uso (PAULA FILHO, 2009)

N°	Caso de Uso	Descrição Resumida
1	Gestão de administradores	Permite criar administrador para acesso ao software e alterar seus dados e permissões.
2	Gestão de tipos de serviços	Permite criar tipo de serviço e alterar seus dados.
3	Gestão de áreas	Permite criar área e alterar e alterar seus dados.
4	Gestão de locais	Permite criar local e alterar seus dados.
5	Gestão de grupos	Permite criar grupo e alterar seus dados.
6	Gestão de usuários	Permite criar usuário, ativá-lo, desativá-lo e alterar seus dados.
7	Gerar solicitação de serviço	Permite criar uma nova solicitação de serviço.
8	Gestão de serviços pendentes	Permite visualizar e editar uma solicitação de serviço pendente para: designar motorista, encerrar, cancelar ou imprimir uma solicitação de serviço.
9	Gerar solicitação de viagem	Permite criar uma nova solicitação de viagem.
10	Gestão de viagens pendentes	Permite visualizar e editar uma solicitação de viagem pendente para: editar despesas, designar motorista, encerrar, cancelar ou

N°	Caso de Uso	Descrição Resumida
		imprimir uma solicitação de serviço.
11	Emissão de relatórios	Exibir relatório por nome de usuário com total de serviços e viagens, por data de execução. Exibir relatório por controle de tráfego, por data de execução. Exibir relatório de solicitações canceladas, por data de solicitação.

Na tabela 8 estão relacionados os possíveis atores e a descrição do tipo de acesso que cada um deles terá no software.

Tabela 8: Descrição dos Atores (PAULA FILHO, 2009)

N°	Ator	Descrição
1	Administrador	Administradores do software, acesso completo, faz manutenção dos dados do software.
2	Motorista	Acesso restrito às solicitações de serviço e viagens que lhes foram designadas. Faz solicitação de novos serviços e viagens. Inserir informações sobrè o trajeto e podem encerrar as solicitações.
3	Usuário	Acesso restrito, usuário comum que faz solicitação de novos serviços e viagens.
4	Software de telefonia	Software com dados para consulta de dados de docentes e funcionário.

Também foram relacionadas na tabela 9 as principais características dos usuários que utilizarão o software.

Tabela 9: Características dos usuários representados pelos atores (PAULA FILHO, 2009)

N°	Ator	Frequência de uso	Nível de instrução	Proficiência na aplicação	Proficiência em informática
1	Administrador	Diário	3.° Grau	Completa	Aplicação – Software operacional – Planilha – Processador de texto - Navegação
2	Motorista	Diário em horário	2.° Grau	Operacional	Aplicação – Software operacional - Navegação

N°	Ator	Frequência de uso	Nível de instrução	Proficiência na aplicação	Proficiência em informática
		comercial			
3	Usuário	Diário em horário comercial	2.° Grau	Operacional	Aplicação – Software operacional - Navegação

b) Gerar diagrama de casos de uso

Os relacionamentos que indicam a comunicação entre os casos de uso e atores e casos de uso entre si estão representados pelo diagrama casos de uso da figura 7. É um diagrama de casos de uso desenvolvido na ferramenta STARUML, conforme seção 2.5 item d, com a finalidade de atribuir uma visão geral das funções do software.

d) Detalhar casos de uso

Nas tabelas de 10 a 17, encontra-se descrito o detalhamento dos casos de uso identificados para o software a ser desenvolvido que auxiliará na elaboração do protótipo a ser apresentado ao cliente para aprovação dos requisitos do software.

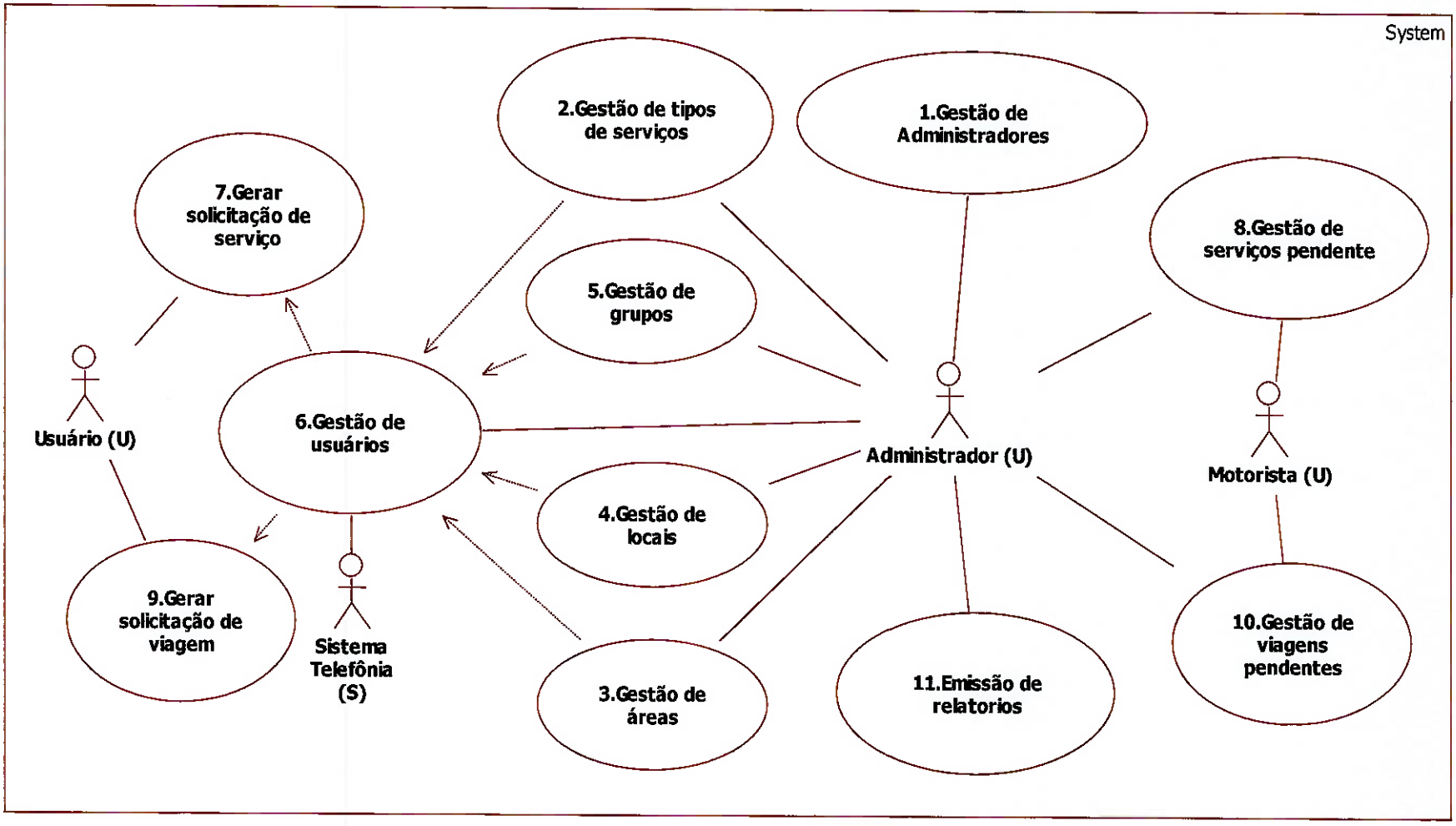

Figura 7: Diagrama de casos de uso

Tabela 10: Detalhamento caso de uso: gestão de administradores (PRESSMAN, 2011)

Número	1
Nome	Gestão de administradores
Descrição	Descreve o processo para criar um novo administrador do software e alterar dados, desativar e reativar um administrador existente.
Ator(es)	Administrador
Pré-condições	Estar conectado ao software como administrador e ter selecionado opção cadastros.
Evento iniciador	Ator seleciona opção cadastro de administradores
Fluxo de Evento	1. Software exibe campos nome, e-mail e permissão do novo administrador e opção salvar. Abaixo exibe os nomes, e-mails, permissões dos administradores cadastrados anteriormente e opções para alterar, reativa ou desativar. 2. Ator informa nome, e-mail e permissão do novo administrador e escolhe opção salvar. 3. Software verifica dados informados. 4. Software armazena dado e retorna mensagem: "novo administrador armazenado". 5. Software vai para o passo 1.
Pós-Condições	Dados do novo administrador armazenados e liberado para uso imediato.
Fluxo alternativo 1	1. Software retorna mensagem: "Caracteres inválidos nos campos digitados ou ele já existe, verifique" (passo 2). 2. O software vai para o passo 1.
Fluxo alternativo 2	1. Ator altera nome, e-mail e/ou permissão de um administrador cadastrado e escolhe opção alterar (passo 2). 2. Software armazena os dados e retorna mensagem: "dados do administrador alterados". 3. O software vai para o passo 1.
Fluxo alternativo 3	1. Ator seleciona opção desativar para um administrador cadastrado (passo 2). 2. Software armazena os dados e retorna mensagem: "administrador desativado". 3. O software vai para o passo 1.
Fluxo	1. Ator seleciona opção reativar para um administrado cadastrado e desativado

alternativo 4	anteriormente (passo 2). 2. Software armazena os dados e retorna mensagem: "administrador reativado". 3. O software vai para o passo 1.
Extensões	Não tem.

O caso de uso 2 Gestão de tipos de serviço é equivalente ao caso de uso 1 em sua funcionalidade, mudando apenas alguns dados a serem armazenados e nomenclatura, por isso não foi necessário detalhá-lo.

Tabela 11: Detalhamento caso de uso: gestão de áreas (PRESSMAN, 2011)

Número	3
Nome	Gestão de áreas
Descrição	Permite criar área e alterar e alterar seus dados
Ator(es)	Administrador
Pré-condições	Estar conectado ao software como administrador e ter selecionado opção cadastros.
Evento iniciador	Ator seleciona opção cadastro de áreas
Fluxo de Evento	1. Software exibe campos sigla e descrição da nova área opção salvar. Abaixo exibe sigla e descrição das áreas cadastradas anteriormente e opção para alterar. 2. Ator altera sigla e/ou descrição e escolhe opção salvar. 3. Software verifica dados informados. 4. Software armazena os dados e retorna mensagem: "dados da área alterados" 5. Software retorna para o passo 1.
Pós-Condições	Dados do novo administrador armazenados.
Fluxo alternativo	1. Software retorna mensagem: "Caracteres inválidos nos campos digitados ou ela já existe, verifique" (passo 4). 2. O software vai para o passo 2.
Extensões	Não tem.

Os casos de uso 4 Gestão de locais e 5 Gestões de grupos são equivalentes ao caso de uso 3 em sua funcionalidade, mudando apenas alguns dados a serem armazenados e nomenclatura, por isso não foi necessário detalhá-los.

Tabela 12: Detalhamento caso de uso: gestão de usuários (PRESSMAN, 2011)

Número	6
Nome	Gestão de usuários
Descrição	Permite criar usuário, ativá-lo, desativá-lo e alterar seus dados.
Ator(es)	Administrador
Pré-condições	Estar conectado ao software como administrador e ter selecionado opção cadastros.
Evento iniciador	Ator seleciona opção cadastro de usuários
Fluxo de Evento	1. Software exibe grupos disponíveis, campo e-mail e opção continuarem. 2. Ator informa grupo, e-mail e escolhe opção continuar. 3. Software verifica se o grupo informado é diferente de docente ou funcionário. 4. Software solicita preencher nome, e-mail, telefone, local, área e grupo. 5. Ator informa nome, e-mail, telefone, local, área, grupo e escolhe opção salvar. 6. Software verifica dados informados. 7. Software armazena os dados e retorna mensagem: "novo usuário armazenado". 8. Software retorna para o passo 1.
Pós-Condições	Dados do novo administrador armazenados.
Fluxo alternativo 1	1. Software encontra dados na base de telefonia e retorna mensagem: "o usuário já existe" (passo 4). 2. Software vai para o passo 1.
Fluxo alternativo 2	1. Software retorna mensagem: "Caracteres inválidos nos campos digitados ou ele já existe, verifique" (passo 4). 2. O software vai para o passo 1.
Extensões	Não tem.

Tabela 13: Detalhamento caso de uso: gerar solicitação de serviço (PRESSMAN, 2011)

Número	7
Nome	Gerar solicitação de serviço
Descrição	Permite criar uma nova solicitação de serviço.
Ator(es)	Administrador, motorista, usuário.
Pré-condições	As bases de dados área, local, grupo, tipo de serviço e usuário devem conter dados, pois o software verifica a existência de dados. Usuário autenticado como administrador, motorista ou usuário. Ter selecionado opção serviços que exibe opções: serviços pendentes, novo serviço, serviços encerrados e serviços encerrados por período.
Evento iniciador	Ator seleciona opção novo serviço.
Fluxo de Evento	1. Software exibe nome, e-mail, grupo, telefone, prédio e área do solicitante e aguarda entrada do nome do docente vinculado, nome do local, telefone, endereço, bairro e cidade do destino, tipo de serviço, descrição/finalidade, data para execução, hora para execução do serviço e opção enviar nova solicitação. 2. Ator informa nome do docente vinculado, nome do local, telefone, endereço, bairro, cidade, tipo de serviço, descrição/finalidade, data para execução, hora para execução e seleciona opção salvar nova solicitação. 3. Software verifica campos. 4. Software exibe mensagem "nova solicitação de serviço armazenada", envia e-mail com de confirmação para o solicitante e um e-mail para o administrador informando que existe nova solicitação pendente. 5. Software e retorna ao passo 1.
Pós-Condições	• Dados armazenados. • E-mail confirmando abertura de nova solicitação enviada para o e-mail do solicitante. • E-mail informando que existe nova solicitação pendente enviado para o administrador.
Fluxo alternativo 1	1. Software exibe mensagem "caracteres inválidos nos campos preenchidos, por favor, verificar" (passo 4). 2. Software retorno ao passo 2.
Extensões	Não há.

Tabela 14: Detalhamento caso de uso: gestão de serviços pendentes (PRESSMAN, 2011)

Número	8
Nome	Gestão de serviços pendentes
Descrição	Permite visualizar e editar uma solicitação de serviço pendente para: designar motorista, encerrar, cancelar ou imprimir uma solicitação de serviço.
Ator(es)	Administrador e motorista.
Pré-condições	Devem existir solicitações de serviços pendentes armazenadas na base serviços, pois o software verifica a existência de dados. Usuário autenticado como administrador ou motorista. Ter selecionado opção serviços que exibe opções: serviços pendentes, novo serviço, serviços encerrados e serviços encerrados por período.
Evento iniciador	Ator seleciona opção serviços pendente.
Fluxo de Evento	1. Software exibe lista com: código do serviço, data da solicitação, nome do solicitante, grupo, data prevista para execução, motorista designado. 2. Ator seleciona solicitação desejada. 3. Software exibe código, data, nome da solicitação, e-mail, telefone, local, grupo e área do solicitante, nome do docente vinculado, nome do local, telefone, endereço, bairro e cidade do destino, o tipo de serviço, descrição/finalidade, data para execução, hora para execução. Abaixo exibe opções de preenchimento: • Nomes dos motoristas e opção designar motorista; • Campo data de execução, hora da execução, número de controle de tráfego, observações do motorista e opção salvar e encerrar; • Data cancelamento, motivo cancelamento e opção cancelar; • Imprimir solicitação. 4. Ator informa nome do motorista e escolhe opção designar motorista. 5. Software verifica campos e exibe mensagem "motorista designado". 6. Ator informa data de execução, hora de execução, número do controle de tráfego, observações e escolhe opção salvar e encerrar. 7. Software verifica campos e exibe mensagem "solicitação de serviço encerrada". 8. Software e retorna ao passo 1.
Pós-Condições	Dados armazenados. Motorista designado. Solicitação de serviço encerrada.

Fluxo alternativo 1	1. Ator não designa motorista no passo 4 e executa o passo 6. 2. Software verifica campos e exibe mensagem "é necessário designar um motorista antes de salvar e encerrar". 3. Software retorna ao passo 3.
Fluxo alternativo 2	1. Software verifica campos e exibe mensagem "caracteres inválidos nos campos preenchidos, verifique" (passo 6). 2. Software retorno ao passo 3.
Fluxo alternativo 3	1. Ator informa motivo do cancelamento e escolhe opção cancelar (passo 4). 2. Software verifica campos e exibe mensagem: "solicitação cancelada". 3. Software retorna ao passo 3.
Fluxo alternativo 4	1. Ator informa motivo do cancelamento e escolhe opção cancelar (passo 4). 2. Software verifica campos e exibe mensagem: "caracteres inválidos nos campos preenchidos, verifique". 3. Software retorna ao passo 1 do fluxo alternativo 4.
Extensões	Não há.

Tabela 15: Detalhamento caso de uso: gerar solicitação de viagem (PRESSMAN, 2011)

Número	9
Nome	Gerar solicitação de viagem
Descrição	Permite criar uma nova solicitação de viagem.
Ator(es)	Administrador, motorista, usuário.
Pré-condições	As bases de dados área, local, grupo e usuário devem conter dados, pois o software verifica a existência de dados. Usuário autenticado como administrador, motorista ou usuário. Ter selecionado opção viagens que exibe opções: viagens pendentes, nova viagem, viagens encerradas e viagens encerradas por período.
Evento iniciador	Ator seleciona opção novo viagem.
Fluxo de Evento	1. Software exibe nome, e-mail, grupo, telefone, prédio, área do solicitante e aguarda entrada do nome do docente vinculado, nome do local, telefone, endereço, bairro, cidade e estado do destino, descrição/finalidade da viagem, data de saída, hora de saída, data de retorno, hora de retorno, despesas e opção enviar nova solicitação. 2. Ator informa nome do docente vinculado, nome do local, telefone, endereço, bairro, cidade, lista com estados, descrição/finalidade da viagem, data de

	saída, hora de saída, data de retorno, hora de retorno, despesas e seleciona opção salvar nova solicitação. 3. Software verifica campos. 4. Software exibe mensagem "nova solicitação de viagem armazenada", envia e-mail com de confirmação para o e-mail do solicitante e envia e-mail para o administrador informando que existe nova solicitação pendente. 5. Software e retorna ao passo 1.
Pós-Condições	• Dados armazenados. • E-mail confirmando abertura de nova solicitação enviada para o e-mail do solicitante. • E-mail informando que existe nova solicitação pendente enviado para o administrador.
Fluxo alternativo	1. Software exibe mensagem "caracteres inválidos nos campos preenchidos, por favor, verificar" (passo 4). 2. Software retorno ao passo 2.
Extensões	Não há.

Tabela 16: Detalhamento caso de uso: gestão de viagens pendentes (PRESSMAN, 2011)

Número	10
Nome	Gestão de viagens pendentes
Descrição	Permite visualizar e editar uma solicitação de viagem pendente para: informar e alterar despesas, informar e alterar quilometragem, designar motorista, encerrar, cancelar ou imprimir uma solicitação de viagem.
Ator(es)	Administrador e motorista.
Pré-condições	Devem existir solicitações de viagens pendentes armazenadas na base viagens, pois o software verifica a existência de dados. Usuário autenticado como administrador ou motorista. Ter selecionado opção viagens que exibe opções: viagens pendentes, nova viagem, viagens encerradas e viagens encerradas por período.
Evento iniciador	Ator seleciona opção viagens pendente.
Fluxo de Evento	1. Software exibe lita com: código da viagem, data da solicitação, nome do solicitante, grupo, data prevista para execução, motorista designado. 2. Ator seleciona solicitação desejada. 3. Software exibe código, data da solicitação, nome do local, telefone, endereço,

	bairro, cidade, estado do destino, descrição/finalidade da viagem, data de saída, hora de saída, data de retorno, hora de retorno e despesas. Abaixo exibe opções de preenchimento: • Nomes dos motoristas e opção designar motorista; • Campo valor estimado e valor utilizado para as seguintes despesas: adiantamento de recursos para posterior ressarcimento, adiantamento de recursos por conta do IQUSP, combustível, sem pernoite e com pernoite. • Campo data de execução, hora da execução, número de controle de tráfego, observações do motorista e opção salvar e encerrar; • Data cancelamento, motivo cancelamento e opção cancelar; • Imprimir solicitação. 4. Ator informa nome do motorista e escolhe opção designar motorista. 5. Software verifica campos e exibe mensagem "motorista designado". 6. Ator informa data de execução, hora de execução, número do controle de tráfego, observações e escolhe opção salvar e encerrar. 7. Software verifica campos e exibe mensagem "solicitação de viagem encerrada". 8. Software e retorna ao passo 1.
Pós-Condições	Dados armazenados. Motorista designado. Solicitação de viagem encerrada.
Fluxo alternativo 1	1. Ator não designa motorista no passo 4 e executa o passo 6. 2. Software verifica campos e exibe mensagem "é necessário designar um motorista antes de salvar e encerrar". 3. Software retorna ao passo 3.
Fluxo alternativo 2	1. Software verifica campos e exibe mensagem "caracteres inválidos nos campos preenchidos, verifique" (passo 6). 2. Software retorno ao passo 3.
Fluxo alternativo 3	1. Ator informa motivo do cancelamento e escolhe opção cancelar (passo 4). 2. Software verifica campos e exibe mensagem: "solicitação cancelada". 3. Software retorna ao passo 3.
Fluxo alternativo 4	1. Ator informa motivo do cancelamento e escolhe opção cancelar (passo 4). 2. Software verifica campos e exibe mensagem: "caracteres inválidos nos campos preenchidos, verifique". 3. Software retorna ao passo 1 do fluxo alternativo 4.
Extensões	Não há.

Tabela 17: Detalhamento caso de uso: emissão de relatórios (PRESSMAN, 2011)

Número	11
Nome	Emissão de relatórios
Descrição	Gerar e exibir relatórios diversos utilizando dados dos serviços e viagens armazenados.
Ator(es)	Administrador
Pré-condições	Devem existir solicitações de viagens e viagens armazenadas nas bases viagens e serviços, pois o software verifica a existência de dados. Usuário autenticado como administrador.
Evento iniciador	Ator seleciona opção relatórios.
Fluxo de Evento	1. Software exibe os seguintes campos para preenchimento: data inicial, data final, opções de relatórios por nome de usuário, por controle de tráfego, solicitações canceladas e opção gerar relatório. 2. Ator informa data inicial, data final, por nome de usuário e escolhe opção gerar relatório. 3. Software localiza as solicitações de serviços e viagens encerradas no período informado e exibe listagem com os seguintes dados: nome de usuário, total de serviços e total de viagens para cada usuário. Cada total permite acesso a uma lista com solicitações do usuário no período informado.
Pós-Condições	Serviços e viagens encerradas exibidas.
Fluxo alternativo 1	Software exibe mensagem "caracteres ou data inválidos, verifique" (passo 3). Software retorna para ao passo 2.
Fluxo alternativo 2	1. Ator informa data inicial, data final, por controle de tráfego e escolhe opção gerar relatório. 2. Software localiza as solicitações de serviços e viagens encerradas no período informado e exibe listagem com os seguintes dados: número de controle de tráfego, total de serviços e total de viagens para cada número de controle de tráfego. Cada total permite acesso a uma lista com solicitações relacionadas ao número do controle de tráfego no período informado.
Fluxo alternativo 3	1. Ator informa data inicial, data final, solicitações canceladas e escolhe opção gerar relatório. 2. Software localiza as solicitações de serviços e viagens encerradas no período informado com os seguintes dados: nome de usuário, total de serviços e total de viagens para cada usuário. Cada total permite acesso a uma lista com solicitações canceladas do usuário no período informado.
Extensões	Não há.

5.2.3 3ª Etapa: Especificar requisitos não funcionais

Tabela 18: Requisitos não funcionais, propriedades (PAULA FILHO, 2009).

N°	Nome	Tipo	Característica	Subcaterísticas	Motivação	Validação
1	Acesso com senhas	Qualidade	Funcionalidade	Segurança de acesso	Programar segurança para garantir que as funções sejam executadas somente pelos usuários com as devidas permissões.	Autenticação no servidor de e-mail e na base de usuários.
2	Restrição de acesso à rede IQUSP	Qualidade	Funcionalidade	Segurança de acesso	Evitar acesso indevido.	Restrição de acesso apenas pelos computadores da rede local do IQUSP.
3	Evitar erros de digitação	Qualidade Persistência	Confiabilidade	Tolerância a falhas Recuperabilidade	Evitar erros de digitação do usuário e sejam armazenados. Evitar que o usuário tenha que redigitar dados.	Realizar teste de unidade para cada campo preenchido. Manter conteúdo digitado pelo usuário no formulário.
4	Funcionar nos navegadores Internet Explorer e Mozilla	Qualidade	Portabilidade	Adaptabilidade Coexistência	Ser utilizado em qualquer navegador. Visualização simples.	Acessar software nos navegadores mais comuns utilizados pelos usuários.

5.2.4 4ª Etapa: Construir protótipo

Para a validação dos requisitos (casos de uso) foi desenvolvido um protótipo, descartável, simples em HTML e de baixa fidelidade, por ser mais rápida e mais barata a sua construção. O protótipo contém a estrutura do software, contemplando as telas e cenários dos casos de uso que se pretende testar, para o usuário poder verificar se atende suas necessidades e se assim for, permitir o seu desenvolvimento. Caso contrário, poderá apontar problemas e erros para que sejam corrigidos.

A seguir, apresenta-se a sequência de telas do fluxo de evento (o caminho principal ou caminho feliz) do caso de uso número 7, "Gerar solicitação de serviço"; demais casos de uso não serão apresentados, pois a finalidade deste item é apenas exemplificar como ficaria um protótipo em HTML.

Inicialmente o software apresenta a tela da figura 8. O ator seleciona a opção novo serviço. O software apresenta a tela da figura 9

Figura 8: Tela após ator executar o evento iniciador do caso de uso gerar solicitação de serviço.

Para melhor compreensão segue-se o fluxo de evento:

1. Software exibe nome, e-mail, grupo, telefone, prédio e área do solicitante e aguarda entrada do nome do docente vinculado, nome do local, telefone, endereço, bairro e cidade do destino, tipo de serviço, descrição/finalidade, data para execução, hora para execução do serviço e opção enviar nova solicitação (figura 9).

2. Ator informa nome do docente vinculado, nome do local, telefone, endereço, bairro, cidade, tipo de serviço, descrição/finalidade, data para execução, hora para execução e seleciona opção salvar nova solicitação (figura 10).

3. Software verifica campos.

4. Software exibe mensagem "nova solicitação de serviço armazenada", envia e-mail com de confirmação para o solicitante e um e-mail para o administrador informando que existe nova solicitação pendente (figura 10).

5. Software e retorna ao passo 1.

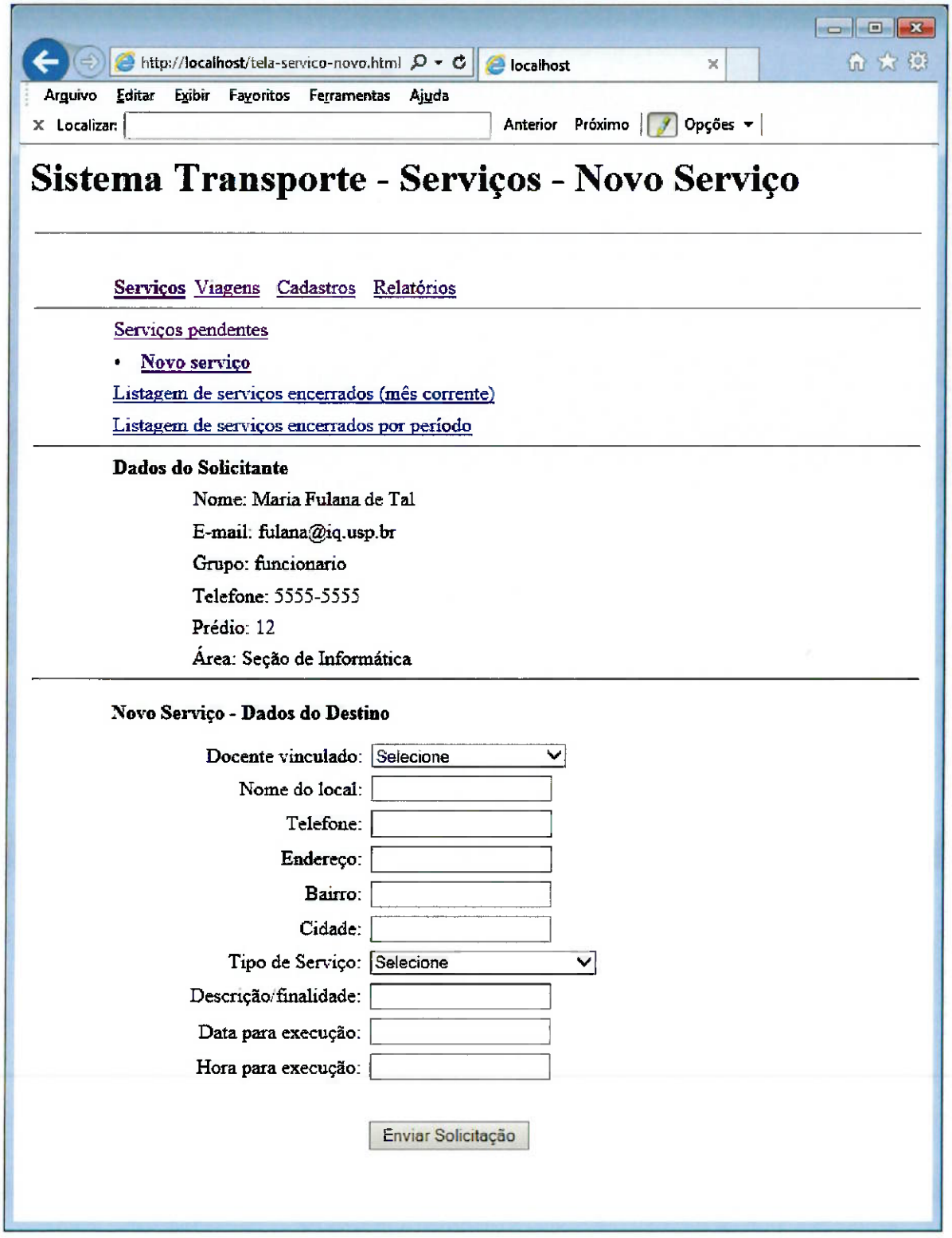

Figura 9: Tela após execução do passo 1 do caso de uso 7, gerar solicitação de serviço.

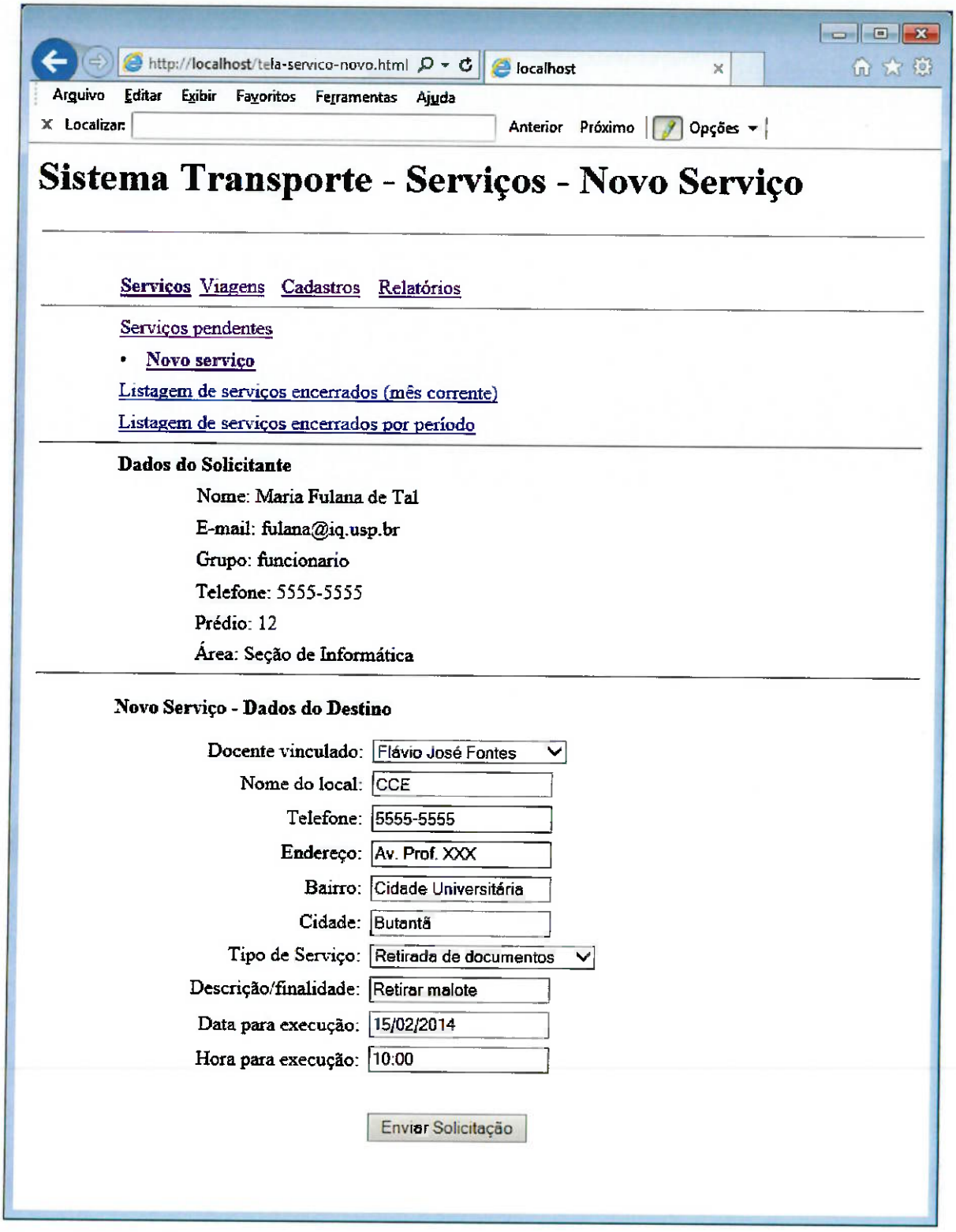

Figura 10: Tela referente ao passo 2 do caso de uso 7, gerar solicitação de serviço.

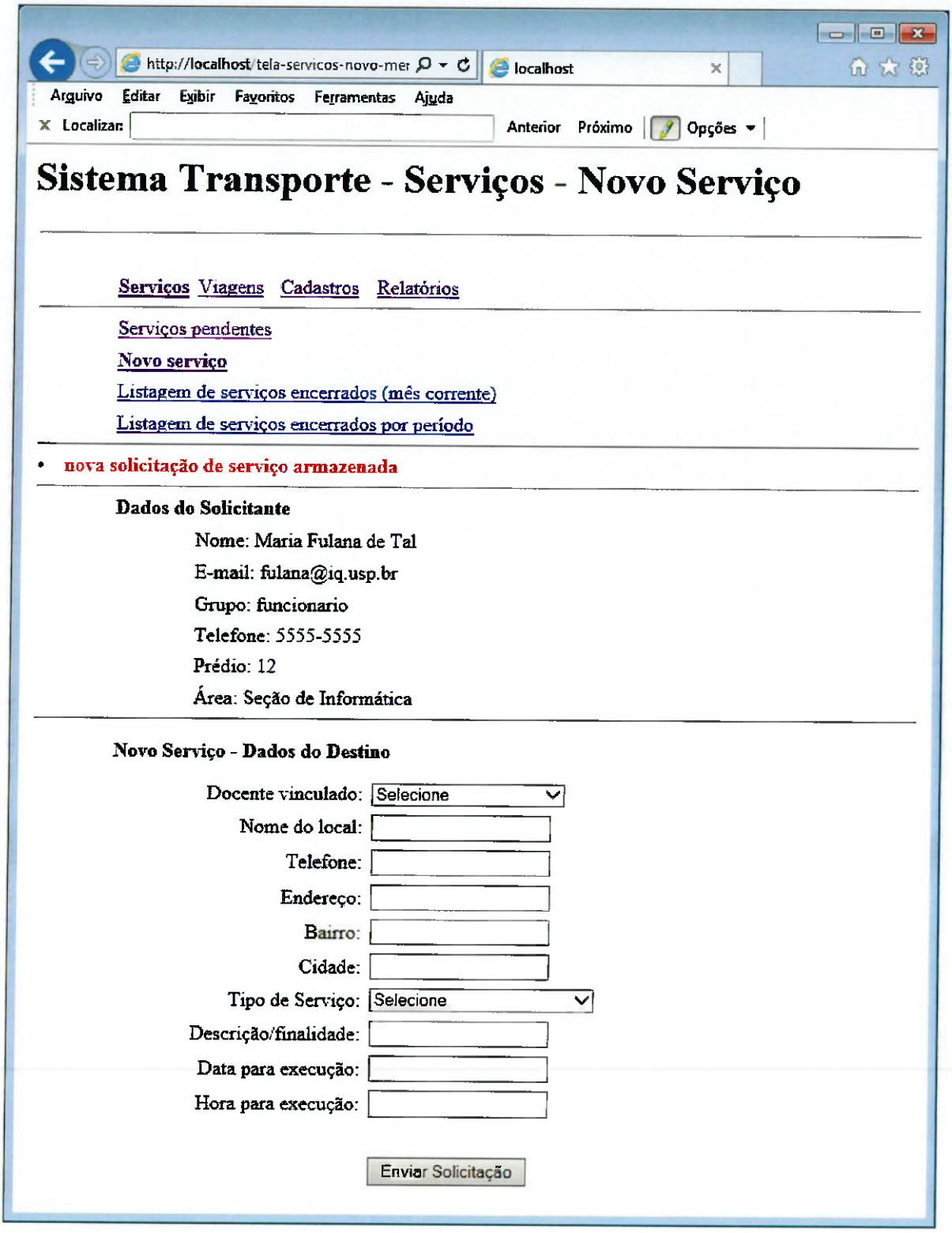

Figura 11: Tela referente aos passos 3, 4 e 5 do caso de uso 7, gerar solicitação de serviço.

5.2.5 5ª. Etapa: Validar requisitos

A validação de requisitos consistiu na apresentação de toda a documentação e protótipo produzidos para o cliente que os analisou a fim de verificar se o software atendia suas necessidades, se era o que realmente ele esperava.

Após análise, o cliente informou que, esta primeira versão da documentação dos requisitos, está dentro de suas expectativas, portanto autorizou a construção do software.

5.3 Processos de gerência de requisitos

Após a aprovação da primeira versão da documentação dos requisitos do software e das matrizes de rastreabilidade aprovados pelo cliente, foi combinado com o cliente que qualquer alteração deverá ser feita formalmente conforme proposta do novo processo de gerência de requisitos apresentada.

Para fazer a solicitação o cliente deverá preencher formulário conforme proposta de processo de gestão de requisitos. O formulário deverá ser encaminhado por e-mail.

O analista responsável pelo desenvolvimento do software em questão, deverá analisar quais serão as mudanças necessárias para realizar as alterações com base nas tabelas de rastreabilidade 19 e 20, calcular o tempo que será necessário à sua execução, identificar os impactos e riscos para o projeto e para a STI, responder ao solicitante.

A tabela 19 apresenta a rastreabilidade entre os casos de uso, no caso os requisitos funcionais, e os requisitos não funcionais que serão implementados.

Tabela 19: Rastreabilidade Casos de uso x Requisitos não funcionais (SAYÃO & LEITE, 2005)

		Requisitos não funcionais				
		Acesso com senhas	Restrição de acesso à rede IQUSP	Evitar erros de digitação	Facilidade de uso	Funcionar em qualquer navegador
Casos de uso		1	2	3	4	5
1	Gestão de administradores	X	X	X		X
2	Gestão de tipos de serviços		X	X		X
3	Gestão de áreas		X	X		X
4	Gestão de locais		X	X		X
5	Gestão de grupos		X	X		X
6	Gestão de usuários	X	X	X	X	X
7	Gerar solicitação de serviço		X			X
8	Gestão de serviços pendentes		X	X		X
9	Gerar solicitação de viagem		X			X
10	Gestão de viagens pendentes		X	X		X
11	Emissão de relatórios		X			X

A tabela 20 apresenta a rastreabilidade entre os casos de uso e os atores que terão acesso ao software. Cada ator terá acesso a determinados casos de uso **(SAYÃO & LEITE, 2005)**.

Tabela 20: Rastreabilidade Ator x Casos de uso (SAYÃO & LEITE, 2005)

		Atores			
		Administrador	Motorista	Usuário	Software de Telefonia
Casos de uso		1	2	3	4
1	Gestão de administradores	X			
2	Gestão de tipos de serviços	X			
3	Gestão de áreas	X			
4	Gestão de locais	X			
5	Gestão de grupos	X			
6	Gestão de usuários	X			X
7	Gerar solicitação de serviço	X	X	X	
8	Gestão de serviços pendentes	X	X		
9	Gerar solicitação de viagem	X	X	X	
10	Gestão de viagens pendentes	X	X		
11	Emissão de relatórios	X			

5.4 Processos de validação do software

A estratégia de teste para esta aplicação consistiu no desenvolvimento de um plano de testes envolvendo a elaboração de casos de teste para os casos de uso, que serão executados pelo usuário após a construção do software o que permitirá o registro e análise dos dados obtidos após sua conclusão. No entanto, a execução só poderá ser realizada quando a construção do software for concluída.

5.4.1 Planejamento de testes

Seu objetivo é permitir que o usuário executasse testes no software e em suas funcionalidades para encontrar defeitos e validar se os requisitos definidos foram atendidos. Lembre-se que testes podem ajudar a localizar erros, o que não garante que o software esteja livre deles se nenhum erro for localizado (SOMMERVILLE, 2011)

Para a validação do software serão executados testes alpha, beta e aceitação conforme descrito na seção 2.5.1.

Para o teste alpha os casos de teste foram elaborados com base nos seguintes casos de uso: gestão de administradores, gestão de tipos de serviços, gestão de áreas, gestão de locais, gestão de grupos, gestão de usuários, gerar solicitação de serviço, gerar solicitação de viagem, gestão de serviços pendentes e gestão de viagens pendentes (vide Apêndice A exemplo de casos de teste). Este teste será executado, após a conclusão do desenvolvimento do software, pelos usuários na área de desenvolvimento.

O teste beta o software será pelos administradores, motorista e alguns usuários selecionados com dados reais. Este teste será realizado no local de trabalho do usuário de modo que o desenvolvedor deverá preparar o ambiente para execução software.

O teste de aceitação consiste em disponibilizar o software aos usuários em exceções, informando que esta em fase experimental.

5.4.2 Execução e registro

A execução e registro de testes para este trabalho foram planejados para que sejam executados quando a construção do software for concluída, porque o software solicitado é relativamente pequeno e os requisitos foram considerados bem definidos por cliente e desenvolvedor. No entanto, não são apresentados na aplicação deste trabalho uma vez que o software ainda será desenvolvido (seção 2.5.2).

A execução dos testes será realizada da seguinte forma:

Teste alpha:

- Local para realização do teste: ambiente de desenvolvimento.
- Usuários selecionados: pelo menos um administrador.
- Duração do teste: máximo de 2 horas, um usuário por vez.
- Execução dos casos de teste.
- Os testes serão acompanhados pelo desenvolvedor anotará os defeitos e fará todas as correções posteriormente.

Teste beta:

- Local para realização do teste: ambiente de trabalho normal dos usuários.
- Usuários selecionados: administradores, motoristas e pelo menos 3 usuários comuns.
- Duração do teste: uma semana.
- Após as correções dos defeitos encontrados no teste alpha, o software será disponibilizado para os usuários selecionados e eles ficarão a vontade para entrar com todas as informações fictícias.
- Defeitos deverão ser reportados por e-mail ao desenvolvedor.

Teste de aceitação:

- Local para realização do teste: ambiente de trabalho normal dos usuários.
- Usuários selecionados: administradores, motoristas e usuários comuns.
- Duração do teste: duas semanas.
- Após as correções dos defeitos encontrados no teste beta, o software será disponibilizado para todos os usuários em ambiente de produção para entrada de informações legítimas.
- Todos deverão ser informados que o software está em fase experimental e que defeitos, assim que detectados, devem ser reportados ao desenvolvedor por e-mail.
- Os defeitos serão corrigidos e ao término de duas semanas, não havendo manifestação do solicitante deverá enviar um e-mail ao desenvolvedor informando seu aceite.

5.5. Avaliação dos Resultados

A aplicação do processo de engenharia de requisitos aproximou o desenvolvedor do usuário final, o que possibilitou a captação das informações necessárias para a compreensão das necessidades do cliente, e então elaborar a especificação dos requisitos de forma precisa. Porém na hora da validação dos requisitos, os clientes reclamaram da quantidade de documentos produzidos e só entendeu realmente o software durante a navegação no protótipo. Mas no contexto global, apresentou-se satisfeito por conseguir ser compreendido pelo desenvolvedor e sentir mais segurança para aprovar o desenvolvimento do novo software.

O cliente foi informado sobre como será o novo processo de gerência de requisitos, o que não gerou grandes impasses, o cliente apenas disse estar um pouco mais burocrático para solicitar mudanças no software. Porém para a área de desenvolvimento será importante, uma vez que auxiliará na análise das solicitações de mudanças, em sua manutenção ou para justificar a inviabilidade das solicitações.

A aplicação do processo de validação do software será realizada após a construção do software. No entanto, os casos de teste já foram planejados e autorizados pelo cliente e sua aplicação será como um treinamento para os funcionários da Seção.

5.6 Considerações sobre a aplicação

Analista de softwares e cliente chegaram a um acordo quanto ao conjunto de requisitos que deveriam ser atendidos nesta primeira versão do software e quanto ao planejamento de teste a ser realizado. Acredita-se que com estas implementações seja possível construir o software mais rapidamente, agregando certa qualidade e facilitando a rastreabilidade quando houver necessidade de mudanças.

6. CONSIDERAÇÕES FINAIS

O trabalho dos principais conceitos de engenharia de requisitos, gerência de requisitos e validação do software foram utilizados para desenvolver novos processos que permitiram melhorar o processo de desenvolvimento de software na STI. Com base na fundamentação teórica desenvolvida foi possível aplicar os novos processos a um novo software que será desenvolvido pela STI, implementar um meio de gerenciar possíveis mudanças de requisitos e planejar testes no software que permitirá ao usuário verificar se suas necessidades foram atendidas após a construção do software.

6.1 Conclusões

Ao estudar o processo de desenvolvimento da STI, com base nos conceitos de engenharia de requisitos apresentados na fundamentação teórica, diagnosticou-se que havia outros problemas além daqueles apresentados pelos analistas, principalmente porque muito pouco sobre o desenvolvimento de software da área foi documentado ao longo do tempo.

Com o diagnóstico obtido, três novos processos foram desenvolvidos: processo de elicitação de requisitos, processo de gestão de requisitos e validação de requisitos. Aplicados a um projeto de software solicitado à STI, apresentou resultados positivos com relação à satisfação do usuário em conseguir entender e se fazer entender com a área de desenvolvimento.

Após este trabalho, o processo de desenvolvimento de software da STI agregou padrões de documentação baseados nos conceitos de engenharia de software, o que melhorou a organização da demanda de solicitações de software.

Também facilitou o entendimento entre desenvolvedores e clientes, por focar principalmente em atividades que têm maior contato com o cliente para obter o melhor resultado na satisfação do cliente.

6.2 Contribuições do Trabalho

Os processos desenhados para elicitação de requisitos, gestão de requisitos e validação, aplicados na área de informática de uma instituição pública, constituíram a principal contribuição deste trabalho. Estes processos estão voltados para aplicações de pequeno porte e, desta forma, podem ser utilizados por um grande número de pequenas organizações públicas e privadas, ou em pequenos projetos.

6.3 Trabalhos Futuros

Pretende-se aplicar os padrões desenvolvidos neste trabalho aos outros softwares a serem desenvolvidos pela STI, formando uma base para consultas e rastreabilidade dos projetos, até que estes processos atinjam certa maturidade e possam ser assimilados por todos. Posteriormente pretende-se agregar ao processo outros itens da engenharia de requisitos como o planejamento do projeto e gerenciamento do projeto através de um trabalho mais adequado desses conceitos.

REFERÊNCIAS

ABNT – ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICA. **NBR ISO/IEC 9126**. (2003). Engenharia de software - Qualidade de produto - Parte 1: Modelo de Qualidade.

FERREIA, A. (2008). **Miniaurélio: o dicionário da língua portuguesa** (7a. ed.). Curitiba, PR, Brasil: Positivo.

JENNIFER, P., ROGERS, Y., & SHARP, H. (2005). **Design de interação: além da interação homem-computador**. Porto Alegre: Bookman.

KOTONYA, G., & SOMMERVILLE, I. (2002). **Requirements Engeneering: processes and techniques**. Chichester, England: John Wiley & Sons.

LEFFINGWELL, D., & WIDRIG, D. (2003). **Managing Software Requirements - A Use Case Approach** (2a. ed.). Pearson Education.

PAULA FILHO, W. (2009). **Engenharia de Software Fundamentos, Métodos e Padrões** (3. ed.). Rio de Janeiro: LTC.

PRESSMAN, R. (2011). En**genharia de Software Uma Abordagem Profissional** (7a. ed.). Porto Alegre, RS: AMGH.

SAYÃO, M., & LEITE, J. (2005). **Rastreabilidade de Requisitos. Revista de Informática Teórica e Aplicada - RITA**, XII.

SOMMERVILLE, I. (2011). **Engenharia de Software** (9. ed.). São Paulo: Pearson Education.

GLOSSÁRIO

[Software] – "1. Em um sistema computacional, o conjunto dos componentes informacionais, que não faz parte do equipamento físico e inclui os programas e os dados a eles associados. 2. Qualquer programa ou conjunto de programas de computador" **(FERREIA, 2008)**.

[Sistema] – "10. *Inform.* Conjunto formado por um ou mais computadores, seus periféricos e os programas mais utilizados" **(FERREIA, 2008)**.

APÊNDICE A – Caso de teste – Caso de uso gerar solicitação de serviço

Caso de teste referente ao caso de uso gerar solicitação de serviço.

1) Acesse o software com seu login e senha usuário.
2) Selecione Serviços
3) Selecione Novo Serviço
4) Verifique se seus dados estão corretos em Dados do Solicitante.

   Dados do Solicitante:

   Nome: Maria Fulana de Tal

   E-mail: fulana@iq.usp.br

   Grupo: funcionário

   Telefone: 5555-5555

   Prédio: 12

   Área: Seção de Informática

5) Preencha os campos em banco com os dados do destino conforme abaixo:
   Novo Serviço - Dados do Destino:

   Docente vinculado: Maria Antônia

   Nome do local: Centro de Computação Eletrônica

   Telefone: 5555-5555

   Endereço: Av. Prof. Luciano G. 745

   Bairro: Cidade Universitária

   Cidade: Butantã

   Tipo de Serviço: Retirada de documentos

   Descrição/finalidade: Retirar contratos

Data para execução: 28/01/2014

Hora para execução: 10:00.

6) Clique em enviar solicitação

7) Verifique se a mensagem "caracteres inválidos nos campos preenchidos, por favor, verificar" foi exibida.

8) Preencha os campos em banco novamente com os dados do destino conforme abaixo:

Novo Serviço - Dado do Destino

Docente vinculado: Selecione Maria Antônia%*
Nome do local: Centro de Computação Eletrônica
Telefone: 5555-5555
Endereço: Av. Prof. Luciano G. 745
Bairro: Cidade Universitária$%
Cidade: Butantã
Tipo de Serviço: Retirada de documentos
Descrição/finalidade: Retirar contratos
Data para execução: 28/01/2014
Hora para execução: 10:00

9) Clique em enviar solicitação

10)Verifique se a mensagem "caracteres inválidos nos campos preenchidos, por favor, verificar" foi exibida.

Apontamentos de erros, defeitos e problemas feitos pelo usuário.

Apontamentos de erros, defeitos e problemas feitos pelo desenvolvedor.

Outras observações

ANEXO A - NBR ISSO/IEC 9126-1 – Engenharia de software – Qualidade de produto – Pare 1: Modelo de qualidade – Modelo de qualidade para qualidade externa e interna.

Capítulo 6 - Modelo de qualidade para qualidade externa e interna

Esta seção define o modelo de qualidade externo e interno. Ele categoriza os atributos de qualidade de software em seis características (funcionalidade, confiabilidade, usabilidade, eficiência, manutenibilidade e portabilidade) as quais são, por sua vez, subdivididas em subcaracterísticas (figura 4). As subcaracterísticas podem ser medidas por meio de métricas externas e internas.

Figura 4 - Modelo de qualidade para qualidade externa e interna

Uma definição é atribuída para cada característica e para cada subcaracterística do software que influencia a característica de qualidade. A capacidade do software é determinada por um conjunto de atributos internos que pode ser medido, para cada característica e subcaracterística. Exemplos de métricas internas são dados na ISO/IEC 9126-3. As características e subcaracterísticas podem ser medidas externamente pelo grau da capacidade do sistema contendo o software. Exemplos de métricas externas são dados na ISO/IEC 9126-2.

NOTAS

1 Existe uma subcaracterística de conformidade para todas as características, já que os princípios são geralmente aplicáveis a todas as características de qualidade externa e interna.

2 Algumas características nesta parte da NBR ISO/IEC 9126 relacionam-se com dependabilidade. Características de dependabilidade são definidas para todos os tipos de sistemas na IEC 60050-191 e, quando um termo nesta parte da NBR ISO/IEC 9126 também estiver definido na IEC 60050-191, a definição dada é compatível no sentido geral.

6.1 Funcionalidade

Capacidade do produto em software em prover funções que atendam às necessidades explícitas e implícitas, quando o software estiver sendo utilizado em condições especificadas.

NOTAS

1 Esta característica está relacionada com o que o software faz para atender às necessidades, enquanto que outras características estão principalmente relacionadas à quando e como ele atende às necessidades.

2 Para as necessidades explícitas e implícitas nesta característica, a nota da definição de qualidade em B.21 é aplicável.

3 Para um sistema que seja operado por um usuário, a combinação de funcionalidade, confiabilidade, usabilidade e eficiência podem ser medidos externamente pela qualidade em uso (ver seção 7).

6.1.1 Adequação

Capacidade do produto de software em prover um conjunto apropriado de funções para tarefas e objetivos do usuário especificados.

NOTAS

1 Exemplo de conjunto apropriado de funções é a composição de funções orientadas por tarefas a partir de suas subfunções constituintes e também a capacidade de tabelas.

2 Adequações correspondem à adequação à tarefa da ISO 9241-10.

3 Adequações também afetas a operacionalidade.

6.1.2 Acurácia

Capacidade do produto de software em prover, com o grau de precisão necessário, resultados ou efeitos corretos ou conforme acordados.

6.1.3 Interoperabilidade

Capacidade do produto em software de interagir com um ou mais sistemas especificados.

NOTA - Interoperabilidade é usada no lugar de compatibilidade, para evitar possível ambiguidade com a subcaracterística capacidade para substituir (ver 6.6.4).

6.1.4 Segurança de acesso

Capacidade do produto de software em proteger informações e dados, de forma que pessoas ou sistemas não autorizados não possam lê-los nem modificá-los e que não seja negado o acesso às pessoas ou sistemas autorizados.

[NBR ISO/IEC 12207:1998][9]

NOTAS

1 Isto também se aplica os dados em transmissão.

2 Seguranças são definidas como uma característica de qualidade em uso, já que ela não está relacionada somente com o software, mas com o sistema como um todo.

---

[9] Esta definição não corresponde literalmente à tradução da NBR ISO/IEC 12207, nem ao texto original em Inglês. A Norma Internacional utilizou uma definição baseada na ISO/IEC 12207. A NBR ISO/IEC 12207:1998 define "segurança" como: "proteção de informações e dados de modo que pessoas ou sistemas não autorizados não possam lê-los ou modificá-los e que pessoas ou sistemas autorizados não tenham acesso negado a ele".

6.1.5 Conformidade relacionada à funcionalidade

Capacidade do produto de software estar de acordo com normas, convenções ou regulamentações previstas em leis e prescrições similares relacionadas à funcionalidade.

6.2 Confiabilidade

Capacidade do produto em software de manter um nível de desempenho especificado, quando usado em condições especificadas.

NOTAS

1 No software não ocorre desgaste ou envelhecimento. As limitações em confiabilidade são decorrentes de defeitos na especificação de requisitos, projeto e implementação. As falhas decorrentes desses defeitos dependem de como o produto de software é usado e das opções de programa selecionadas e não do tempo decorrido.

2 A definição de confiabilidade na ISO/IEC 2382-14:1997 é "A habilidade de uma unidade funcional executar uma função requisitada...". Neste documento, funcionalidade é somente uma das características de qualidade de software. Portanto, a definição de confiabilidade foi expandida para "manter um nível de desempenho especificado..." no lugar de "... executar uma função requisitada".

6.2.1 Maturidade

Capacidade do produto em software de evitar falhas decorrentes de defeitos no software.

6.2.2 Tolerância a falhas[10]

Capacidade do produto de software em manter um nível de desempenho especificado em casos de defeitos no software ou de violação de sua interface especificada.

NOTA - O nível de desempenho especificado pode incluir a capacidade de prevenção de falhas.

6.2.3 Recuperabilidade

Capacidade do produto de software de restabelecer seu nível de desempenho especificado e recuperar os dados diretamente afetados no caso de uma falha.

NOTAS

1 Após uma falha, o produto de software poderá ficar inativo por certo período de tempo. A sua recuperabilidade é influenciada por este período de tempo.

2 Disponibilidade é a capacidade de um produto de software de estar pronto para executar uma função requisitada num dado momento, sob condições especificadas de uso. Externamente, a disponibilidade pode ser avaliada pela proporção do tempo total durante o qual o produto de software está disponível. A disponibilidade é, portanto, a combinação de maturidade (a qual controla a frequência de falhas), tolerância a falhas e recuperabilidade (a qual controla o período de tempo inativo após cada falha). Por esta razão ela não foi incluída como uma subcaracterística distinta.

---

[10] A tradução direta do título deveria ser "Tolerância a defeitos", conforme definições em B.8 e B.9. No entanto, utilizou-se o título de "Tolerância a falhas", tendo em vista que o texto referente à subcaracterística, como também a NOTA, ampliam seu contexto para tolerância a falhas e não apenas a defeitos.

6.2.4 Conformidade relacionada à confiabilidade

Capacidade do produto de software estar de acordo com normas, convenções ou regulamentações relacionadas à confiabilidade.

6.3 Usabilidade

Capacidade do produto de software de ser compreendido, aprendido, operado e atraente ao usuário, quando usado em condições específicas.

NOTAS

1 Algum aspectos como funcionalidade, confiabilidade e eficiência também afetarão a usabilidade, mas para os propósitos da NBR ISO/IEC 9126 não são classificados como usabilidade.

2 Como usuários pode-se incluir operadores, usuários finais e usuários indiretos que sejam dependentes ou estejam sob influência do uso do software. Convém que a usabilidade considere todos os diferentes ambientes de usuários que o software afetará. Como exemplos de ambientes a considerar pode-se incluir o ambiente onde usuários estão sendo preparados para uso do produto e o ambiente onde já se Permite avaliação de resultados do uso do produto.

6.3.1 Inteligibilidade

Capacidade do produto de software de possibilitar ao usuário compreender se o software é apropriado e como ele pode ser usado para tarefas e condições de uso específicas.

NOTA - A inteligibilidade dependerá da documentação e das impressões iniciais oferecidas pelo software.

6.3.2 Apreensibilidade

Capacidade do produto de software de possibilitar ao usuário aprender sua aplicação.

NOTA - Os atributos internos correspondem à adequação ao aprendizado, como definido na ISO 9241-10.

6.3.3 Operacionalidade

Capacidade do produto de software de possibilitar ao usuário operá-lo e controlá-lo.

NOTAS

1 Aspecto de adequação, modificabilidade, adaptabilidade e capacidade para ser instalado podem afetar a operacionalidade.

2 Operacionalidades corresponde à controlabilidade, tolerância a erros e conformidade com as expectativas do usuário, como definido na ISO 9241-10.

3 Para um sistema que é operado por um usuário, a combinação de funcionalidade, confiabilidade, usabilidade e eficiência podem ser medidas externamente pela qualidade em uso.

6.3.4 Atratividade

Capacidade do produto de software de ser atraente ao usuário.

NOTA - Isto se refere a atributos de software que possuem a intenção de tornar o software mais atraente para o usuário, como o uso de cores e a natureza do projeto gráfico.

6.3.5 Conformidade relacionada à usabilidade

É a capacidade do produto de software estar de acordo com normas, convenções, guias de estilo ou regulamentações relacionadas à usabilidade.

6.4 Eficiência

É a capacidade do produto de software de apresentar desempenho apropriado, relativo à quantidade de recursos usados, sob condições especificadas.

NOTAS

1 Recurso podem incluir outros produtos de software, configurações de hardware e software do sistema e materiais (por exemplo, papel para impressão, disquetes).

2 Para um sistema que é operado por um usuário, a combinação de funcionalidade, confiabilidade, usabilidade e eficiência podem ser medidas externamente pela qualidade em uso.

6.4.1 Comportamento em relação ao tempo

Capacidade do produto de software de fornecer tempos de resposta e de processamento, além de taxas de transferência, apropriados, quando o software executa suas funções sob condições estabelecidas.

6.4.2 Utilização de recursos

Capacidade do produto de software de usar tipos e quantidades apropriados de recursos, quando o software executa suas funções sob condições estabelecidas.

NOTA - Recursos humanos está incluída como parte da produtividade (ver 7.1.2).

6.4.3 Conformidade relacionada à eficiência

Capacidade do produto de software de estar de acordo com normas e convenções relacionadas à eficiência.

6.5 Manutenibilidade

Capacidade do produto de software de ser modificado. As modificações podem incluir correções, melhorias ou adaptações do software devido a mudanças no ambiente e nos seus requisitos ou especificações funcionais.

6.5.1 Analisabilidade

Capacidade do produto de software de permitir o diagnóstico de deficiências ou causas de falhas no software, ou a identificação de partes a serem modificadas.

6.5.2 Modificabilidade

Capacidade do produto de software de permitir que uma modificação especificada seja implementada.

NOTAS

1 Implementação inclui modificações no código, projeto e documentação.

2 Se o software for modificável pelo usuário final, a modificabilidade pode afetar a operacionalidade.

6.5.3 Estabilidade

Capacidade do produto de software de evitar efeitos inesperados decorrentes de modificações no software.

6.5.4 Testabilidade

Capacidade do produto de software de permitir que o software, quando modificado, seja validado.

6.5.5 Conformidade relacionada à manutenibilidade

Capacidade do produto de software de estar de acordo com normas ou convenções relacionadas à manutenibilidade.

6.6 Portabilidade

Capacidade do produto de software de ser transferido de um ambiente para outro.
NOTA - O ambiente pode ser organizacional, de hardware ou de software.

6.6.1 Adaptabilidade

Capacidade do produto de software de ser adaptado para diferentes ambientes especificados, sem necessidade de aplicação de outras ações ou meios além daqueles fornecidos para essa finalidade pelo software considerado.

NOTAS

1 Adaptabilidade inclui a possibilidade de ajustes da capacidade interna (por exemplo, campos de tela, tabelas, volume de transações, formato de relatórios, etc.).
2 Se o software for adaptável pelo usuário final, adaptabilidade corresponde à adequação à individualização, como definido na NBR ISO 9241-10, e pode afetar a operacionalidade.

6.6.2 Capacidade para ser instalado

Capacidade do produto de software para ser instalado em um ambiente especificado.

NOTA - Se o software for instalável pelo usuário final, a capacidade para ser instalada afeta a adequação e a operacionalidade.

.

6.6.3 Coexistência

Capacidade do produto de software de coexistir com outros produtos de software independentes, em um ambiente comum, compartilhando recursos comuns.

6.6.4 Capacidade para substituir

Capacidade do produto de software de ser usado em substituição a outro produto de software especificado, com o mesmo propósito e no mesmo ambiente.

NOTAS

1 Por exemplo, numa nova versão de um produto de software, a capacidade para substituir é importante para o usuário quando da atualização da versão.

2 Capacidades para substituir são utilizadas no lugar de compatibilidade para evitar possível ambiguidade com interoperabilidade (ver 6.1.3).

3 Capacidades para substituir podem incluir atributos de capacidade para ser instalado e adaptabilidade. O conceito foi introduzido como uma subcaracterística própria devido à sua importância.

6.6.5 Conformidade relacionada à portabilidade

Capacidade do produto de software de estar de acordo com normas ou convenções relacionadas à portabilidade.

ANEXO B – Lista de questões sugeridas para verificação de requisitos

Abaixo transcrita está uma lista de questões (p. 130) (PRESSMAN, 2011) que podem auxiliar na verificação dos requisitos, elas podem ser modificadas, questões podem ser adicionadas ou excluídas se isso for necessário para o projeto.

- "Os requisitos estão declarados de forma clara"? Eles podem ser mal interpretados?
- A fonte (por exemplo, uma pessoa, uma regulamentação, um documento) do requisito foi identificada? A declaração final do requisito foi examinada pela fonte original ou com ela?
- O requisito será limitado em termos quantitativos?
- Que outros requisitos se relacionam a este requisito? Eles estão claramente indicados por meio de uma matriz de referência cruzada ou algum outro mecanismo?
- O requisito viola quaisquer restrições do domínio do sistema?
- O requisito pode ser testado? Em caso positivo, podemos especificar testes (algumas vezes denominados critérios de validação) para testar o requisito?
- O requisito pode ser associado a qualquer modelo de sistema que tenha sido criado?
- O requisito pode ser associado aos objetivos globais do sistema/produto?
- A especificação é estruturada de forma que leve ao fácil entendimento, fácil referência e fácil tradução em artefatos técnicos?
- Criou-se um índice para a especificação?
- Os requisitos associados ao desempenho, ao comportamento e às características operacionais foram declarados de maneira clara? Quais requisitos parecem estar implícitos?".

ANEXO C - Lista de questões para verificação de casos de uso

As questões (p. 138) (PRESSMAN, 2011) abaixo transcritas podem ser respondidas para cada caso de uso, elas podem ser modificadas, questões podem ser adicionadas ou excluídas se isso for necessário para o projeto.

- Quem é(são) o(s) ator(es) primário(s) e o(s) secundário(s)?
- Quais as metas do ator?
- Que precondições devem existir antes de uma história começar?
- Que tarefas ou funções principais são realizadas pelo ator?
- Que exceções deveriam ser consideradas à medida que uma história é descrita?
- Quais são as variações possíveis na interação do ator?
- Que informações de sistema o ator adquire, produz ou modifica?
- O ator terá de informar o sistema sobre mudanças no ambiente externo?
- Que informações o ator deseja do sistema?
- O ator gostaria de ser informado sobre mudanças inesperadas?"

ANEXO D – Tabelas de casos de uso

A tabela 21 apresenta os casos de uso para o sistema de informatização de mercearia, com resumidas descrições (PAULA FILHO, 2009).

Tabela 21: Casos de Uso (PAULA FILHO, 2009)

Caso de Uso	**Descrição**
Gestão de Usuários	Controle de usuários que terão acesso ao Merci. Provê recuperação, criação, alteração e exclusão.
Gestão Manual de Estoque	Controle manual de entrada e saída de mercadorias, com consulta e atualização do estoque respectivo.
Gestão de Mercadorias	Processamento de recuperação, criação, exclusão e alteração de Mercadorias. Durante a criação e alteração, podem-se incluir Fornecedores existentes da Mercadoria.
Gestão de Fornecedores	Processamento de recuperação, criação, exclusão e alteração de Fornecedores. Durante a criação e alteração, pode-se incluir ou excluir Mercadorias existentes como sendo fornecidas.
Gestão de Pedidos de Compra	Processamento de recuperação, criação, alteração, impressão, baixa e exclusão de Pedidos de Compra. Durante a criação, devem-se especificar o Fornecedor existente a quem o Pedido de Compra é dirigido e os Itens de Compra que o comporão, referentes a Mercadorias existentes.
Emissão de Relatórios	Emissão de relatórios das bases de dados do Merci: relatórios de Mercadorias, Fornecedores, Mercadorias com estoque baixo e relação de Pedidos de Compra.
Abertura do Caixa	Passagem para o MODO DE VENDA, liberando o Caixa da mercearia para a Operação de Venda.
Fechamento do Caixa	Fechamento do Caixa da Mercearia, com totalização das vendas dos dias e mudança para o MODO DE GESTÃO.
Operação de Venda	Operação de Venda ao cliente da mercearia. Durante a operação, é possível incluir, alterar e excluir Itens de Venda de Mercadorias especificadas. Ao término da operação, o Tíquete de Venda é emitido, e o saldo no Caixa e os níveis de estoque das Mercadorias dos Itens de Venda são atualizados.
Emissão de Nota Fiscal	Emissão opcional de Nota Fiscal para o cliente da mercearia (extensão da Operação de Venda).

A tabela 22 apresenta os atores do sistema de informatização de mercearia, com resumidas descrições (PAULA FILHO, 2009).

Tabela 22: Descrição dos atores (PAULA FILHO, 2009).

Número	Ator	Definição
1	Caixeiro	Funcionário operador comercial de caixa.
2	Gerente	Funcionário responsável pela abertura e fechamento do caixa, além do cadastramento de usuários.
3	Gestor de Compras	Funcionário responsável pela gestão dos cadastros de mercadorias e fornecedores e pela emissão e acompanhamento de pedidos de compra.
4	Gestor de Estoque	Funcionário responsável pela manutenção da consistência entre o estoque físico da mercearia e o estoque cadastrado no **Merci**.
5	Sistema Financeiro	Sistema de gestão financeira, que recebe os detalhes financeiros das transações diárias, para utilização posterior pela administração financeira da mercearia.

A tabela 23 apresenta as características dos atores do sistema de informatização de mercearia **(PAULA FILHO, 2009)**.

Tabela 23: Características dos usuários representados pelos atores (PAULA FILHO, 2009).

Número	Ator	Frequência de uso	Nível de instrução	Proficiência na aplicação	Proficiência em informática
1	Caixeiro	Diário em horário comercial	1.° Grau	Operacional	Aplicação
2	Gerente	Diário	2.° Grau	Completa	Aplicação – Sistema operacional

Número	Ator	Frequência de uso	Nível de instrução	Proficiência na aplicação	Proficiência em informática
3	Gestor de Compras	Diário	3.° Grau	Completa	Aplicação – Sistema operacional – Planilha – Processador de texto
4	Gestor de Estoque	Diário	1.° Grau	Operacional	Aplicação

ANEXO E – Formulários utilizados pela Seção de Veículos

	INSTITUTO DE QUÍMICA / USP ***SEÇÃO DE VEÍCULOS***
	SOLICITAÇÃO DE SERVIÇOS

Nome do Solicitante	
Função	
Ramal	
Bloco	
Local	
Endereço	
Bairro	
Cidade	
Telefone para contato	
Descrição	
Dia p/ execução	
Horário p/ execução	
Observação	
Data da Solicitação	

PARA USO DA SEÇÃO DE VEÍCULOS

Recebido por	
Data	
Atendido por	
Data	
Horário	
Controle de Tráfego nº	
Observação	

Figura 12: Formulário de serviços (exemplo)

INSTITUTO DE QUÍMICA/USP
SEÇÃO DE VEÍCULOS

SOLICITAÇÃO DE VEÍCULO OFICIAL PARA VIAGEM

NOME DO INTERESSADO:-

RAMAL:-	BLOCO:-	DATA DA SOLICITAÇÃO:- ____/____/____

LOCAL:-	Nº. DE ACOMPANHANTES:-

CIDADE:-	ESTADO:-

FINALIDADE:-

DATA DE SAÍDA:-____/____/____ HORÁRIO:- ____:____	DATA DE RETORNO:-____/____/____ HORÁRIO:- ____:____

AS DESPESAS COM A VIAGEM (COMBUSTÍVEL, REFEIÇÃO DO MOTORISTA, PEDÁGIO, ALOJAMENTO, ETC) CORRERÃO POR CONTA DE:-

( ) INTERESSADO ( ) INSTITUTO DE QUÍMICA ( ) OUTROS

ESPECIFIQUE:-

ASSINATURA DO INTERESSADO:-

DESCRIÇÃO DAS DESPESAS	VALOR ESTIMADO	VALOR UTILIZADO
Adiantamento de recursos para posterior ressarcimento pelo interessado		
Adiantamento de recursos por conta do Instituto de Química		
Combustível		
Pedágio		
Diárias:- ( ) com pernoite – ( ) sem pernoite		
TOTAL		

Km inicial:- Km final:- Km percorrido:-

AUTORIZAÇÃO DA ÁREA FINANCEIRA DO INSTITUTO DE QUÍMICA

ASSINATURA: DATA:- ____/____/____

Figura 13: Formulário de viagem (exemplo)