UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

ESCOLA DE INFORMÁTICA APLICADA

Avaliação do Uso de Linguagem Ubíqua no Detalhamento de Requisitos Utilizando Gherkin

Rafaela da Fonseca Sampaio

Orientador

Gleison dos Santos Souza

RIO DE JANEIRO, RJ – BRASIL

DEZEMBRO DE 2014

Avaliação do Uso de Linguagem Ubíqua no Detalhamento de Requisitos Utilizando Gherkin

Rafaela da Fonseca Sampaio

Projeto de Graduação apresentado à Escola de Informática Aplicada da Universidade Federal do Estado do Rio de Janeiro (UNIRIO) para obtenção do título de Bacharel em Sistemas de Informação.

Aprovada por:

_____

Prof. Gleison dos Santos Souza, D.Sc. (UNIRIO)

_____

Prof. Alexandre Luis Corrêa, D.Sc. (UNIRIO)

_____

Rafael Targino dos Santos , M.Sc. (Laboratório de Engenharia de Software da PUC-Rio)

RIO DE JANEIRO, RJ – BRASIL.
DEZEMBRO DE 2014

Agradecimentos

Agradeço à minha família por toda a dedicação e apoio: aos meus pais, Cosme e Maria Helena por me incentivarem, ajudarem e acreditarem em mim em todos os momentos. Vocês são a razão de tudo que eu sou e de tudo que eu já conquistei.

Ao Professor Gleison Santos, pelo apoio, auxílio, paciência e dedicação ao longo de todo este trabalho. Tenho certeza de que não poderia ter acertado mais na escolha de um orientador.

Aos meus amigos, em especial à Marena e ao Targino, agradeço por toda a motivação. Ao Pedro Fernandes por todo o apoio, compreensão e positividade durante este período.

Agradeço também a todos aqueles que contribuíram para o meu crescimento pessoal e profissional, a todos os professores, colegas de trabalho e amigos que me serviram de inspiração e me ajudaram a conquistar este sonho.

À todos vocês, meu muito obrigada!

Rafaela da Fonseca Sampaio

RESUMO

Esta monografia tem como objetivo avaliar de forma comparativa o detalhamento de requisitos através do uso de casos de uso e de documentação em linguagem ubíqua com o uso de Gherkin. Como base teórica para o trabalho, foram apresentados conceitos de desenvolvimento de software tradicional e ágil, TDD e BDD, linguagem ubíqua, Cucumber e Gherkin. Foi realizada uma pesquisa que abordou a comparação entre as duas formas de detalhamento e com a avaliação dos resultados desta pesquisa constatamos ser viável a utilização de features para o detalhamento de requisitos

Palavras-chave: detalhamento de requisitos, requisitos, casos de uso, Gherkin, linguagem ubíqua.

ABSTRACT

This work aims to evaluate comparatively the detailed requirements by using use cases and documentation in ubiquitous language using Gherkin. As a theoretical basis for the work were presented traditional software development concepts and agile, TDD and BDD, ubiquitous language, Cucumber and Gherkin. A survey was conducted that addressed the comparison between the two forms of detailing and evaluating the results of this study found to be feasible to use features for detailing requirements.

Keywords: detailing requirements, requirements, use cases, Gherkin, ubiquitous language.

Índice

Índice de Tabelas

Índice de Figuras

1 Introdução

1.1 Contexto

De acordo com Leffingwell (1997), os requisitos estão associados aos principais problemas do desenvolvimento de software e 40 a 60% dos problemas encontrados nos projetos são consequências de falhas no levantamento de requisitos.

No detalhamento de requisitos, muitas vezes é necessário entender que o cliente precisa de algo que nem ele mesmo sabe ou entende que precisa, outras vezes um requisito identificado no início de um projeto torna-se descartável ao longo do desenvolvimento (Kotonya e Sommervile, 1998).

Standish (1994) aponta, no relatório intitulado "Chaos", as seguintes estatísticas sobre os projetos de desenvolvimento de software:

- 31% dos projetos serão cancelados antes de estarem completos;
- 53% dos projetos irão custar mais que 189% de suas estimativas;
- Apenas 16,2% dos projetos são finalizados a tempo e dentro do orçamento;
- Em grandes empresas apenas 9% dos projetos são finalizados a tempo e dentro do orçamento;

De acordo com Blackburn *et al.* (2001), estudos mostram que 50% dos problemas identificados durante testes apontam para falhas nos requisitos. Estas falhas custam caro, mas é possível preveni-las. Boehm e Basili (2001) afirmam que 40 a 50% do retrabalho realizado nos projetos de software são evitáveis, e, além disso, o conserto de um problema em fases avançadas do desenvolvimento pode custar 100 vezes mais do que encontrá-lo e resolvê-lo nas fases de requisitos ou de projeto.

1.2 Motivação

As Organizações sofrem com problemas da engenharia de requisitos há muito tempo, independente do método utilizado no desenvolvimento de software, ainda existem casos em que são entregues sistemas que muitas vezes não refletem a realidade e não atendem às expectativas dos usuários e clientes. Requisitos que não refletem reais necessidades dos usuários, incompletos, e/ou inconsistentes, mudanças em requisitos já acordados e a dificuldade para conseguir um entendimento comum entre usuários e desenvolvedores são as principais dificuldades relatadas, provocando retrabalho, atrasos no cronograma, custos ultrapassados e a insatisfação das partes interessadas no projeto. Isso se deve ao fato de ser complexo identificar, entender e detalhar os requisitos (Kotonya e Sommervile, 1998). A insatisfação com a abordagem tradicional de desenvolvimento de software, segundo Sommerville (2011), levou alguns desenvolvedores a buscar métodos "ágeis". Os métodos ágeis baseiam-se no desenvolvimento incremental de software e são mais adequados para casos em que os requisitos mudam frequentemente durante o desenvolvimento. Estes métodos visam diminuir burocracias no processo evitando trabalho que tem valor a longo prazo apenas e eliminando documentações que provavelmente nunca serão utilizadas.

Algumas técnicas que foram propostas originalmente em cenários ágeis, como o Desenvolvimento Orientado por Comportamento (*Behaviour-Drivem Development – BDD*), que foi baseado no Desenvolvimento Orientado a Testes (*Test-Driven Development – TDD*) (Wynne e Hellesoy, 2012), também podem ser aplicadas em contextos tradicionais.

Sommerville (2011) aponta que a documentação dos requisitos tem um conjunto diversificado de interessados, dentre eles estão clientes, usuários e especialistas de domínio, gerentes, analistas, programadores e testadores.

Tendo em vista a importância da documentação de requisitos e o fato de que muitos dos problemas encontrados nos requisitos podem ser amenizados através de uma boa comunicação entre os interessados e uma documentação adequada, este trabalho é motivado a avaliar um formato de documentação de requisitos alternativo à tradicional utilização de casos de uso.

1.3 Objetivos

Este trabalho tem como objetivo estudar a aplicação de práticas ágeis para apoiar a engenharia de requisitos, de forma que seja possível introduzir o uso de princípios ágeis dentro de métodos tradicionais de desenvolvimento de software a fim de facilitar o detalhamento dos requisitos e melhorar a comunicação entre a equipe de desenvolvimento e o cliente. Acreditamos que com algumas mudanças, aplicadas no detalhamento dos requisitos, seja possível minimizar os problemas causados por falhas comuns da engenharia de requisitos tradicional.

Desta forma, a proposta desta monografia é avaliar duas abordagens de detalhamento de requisitos de forma comparativa: Casos de Uso, que são documentos tradicionais da engenharia de requisitos, e Features, que são documentações utilizadas normalmente para documentar testes de aceitação automatizados e quando se desenvolve utilizando BDD.

Acreditamos que os requisitos podem ser documentados através de Features e que essa prática pode ajudar no detalhamento de requisitos em projetos de desenvolvimento de software, independentemente de utilizarem processos de software ágeis ou tradicionais.

1.4 Organização do texto

O presente trabalho está estruturado em capítulos e, além desta introdução, será desenvolvido da seguinte forma:

- Capítulo II: Este capítulo contém a Revisão Bibliográfica do trabalho, que apresenta os principais conceitos abordados para o desenvolvimento deste trabalho. Inicialmente são abordados os modelos Cascata e RUP, que são modelos tradicionais de desenvolvimento de software. Em seguida falamos de metodologias ágeis e do Scrum e por fim é abordado o desenvolvimento de requisitos com Gherkin, onde são apresentados os conceitos de linguagem ubíqua, Gherkin, testes de aceitação automatizados, Cucumber e desenvolvimento orientado por comportamento.

- Capítulo III: Este capítulo trata da avaliação do detalhamento de requisitos através de *features* e de casos de uso e descreve os detalhes de cada etapa do estudo realizado.

- Capítulo IV: Este capítulo apresenta as conclusões deste trabalho. Reúne as considerações finais, assinala as contribuições da pesquisa e sugere possibilidades de aprofundamento posterior.

2 **Revisão Bibliográfica**

Neste capítulo são apresentados os principais conceitos que embasam este trabalho. São abordados os modelos Cascata e RUP, metodologias ágeis e Scrum, além de conceitos de linguagem ubíqua, Gherkin, testes de aceitação automatizados, Cucumber e desenvolvimento orientado por comportamento

2.1 Processos Tradicionais de Desenvolvimento de Software

O método mais tradicional de desenvolvimento de software é o modelo Cascata, também chamado de modelo Clássico. O modelo cascata é definido, segundo Leffingwell (2010), como uma sequencia de estágios ordenados, como exemplificado na Figura 1. Os requisitos são acordados primeiro, o projeto é criado, a codificação é feita em seguida e ao final o software é testado para verificar se está conforme os requisitos e o projeto.

Leffingwell (2010) aponta que a criação do modelo Cascata foi consequência da necessidade de melhores previsões e controles sobre os projetos de software de larga escala. Tal necessidade surgiu depois do avanço da indústria de software nas décadas de 1950 e 1960. Winston Royce é apontado como o criador do modelo cascata, mas ironicamente ele o descreve como um modelo que não é recomendado para softwares de grande escala (Leffingwell, 2010).

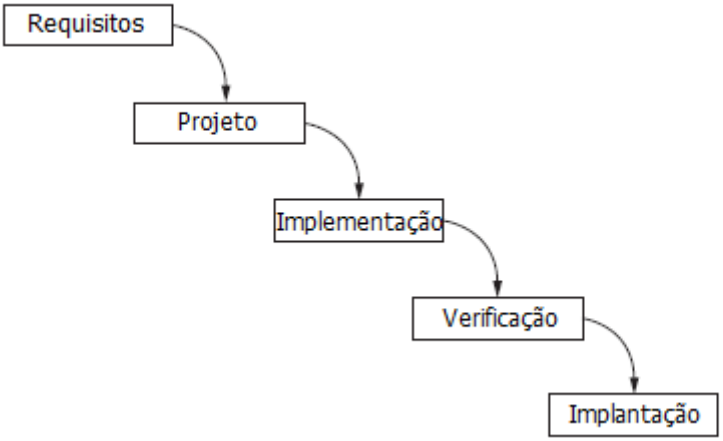


Figura 1 - Modelo "cascata" simplificado.
Fonte: Adaptado de (Leffingwell, 2010)

Falhas apresentadas pelo modelo Cascata, juntamente com as pressões do mercado e avanços nas ferramentas e tecnologias de desenvolvimento de software incentivaram a necessidade de modelos mais inovadores, baseados em descoberta, os quais nos levaram aos processos iterativos das décadas de 1980 e 1990 (Leffingwell, 2010).

Ainda segundo Leffingwell (2010), o modelo RUP (*Rational Unified Process)* foi baseado no modelo Espiral e no RAD (*Rapid Application Development*), com o objetivo de construir aplicações de larga escala nas quais a robustez, a escalabilidade e a extensibilidade eram obrigatórias. O RUP foi lançado em no final da década de 1990 e é um método iterativo e incremental amplamente adotado, atualmente comercializado e suportado pela IBM.

Leffingwell (2010) afirma ainda que o RUP tem provado ser um framework eficaz para a prática e o gerenciamento de desenvolvimento de software em larga escala. Este framework tem sido amplamente difundido na indústria e já foi utilizado em milhares de projetos de sucesso de vários tipos. O RUP foi o primeiro processo de software amplamente adotado que reconhecia a necessidade de sobrepor várias atividades que ocorrem durante as fases de iniciação, elaboração, construção e transição do o ciclo de vida do software (Leffingwell, 2010).

De acordo com Sommerville (2011), o RUP (ver Figura 2) é normalmente descrito a partir de três perspectivas diferentes. São elas:

- *Perspectiva dinâmica*: mostra as fases do modelo durante do tempo. O RUP é um modelo baseado em fases que distingue quatro fases no processo de software, porém é diferente do modelo cascata, onde as fases são atividades do

processo. No RUP as fases estão mais relacionadas ao negócio do que a preocupações técnicas. As interações no RUP podem funcionar de duas maneiras, cada fase pode ser propagada através do modelo e assim os resultados são desenvolvidos incrementalmente. Além disso um conjunto inteiro de fases pode ser propagado incrementalmente.

- *Perspectiva estática*: mostra as atividades do processo que são propagadas pelo modelo. Tem o foco direcionado às atividades de desenvolvimento de software. Essas atividades são chamadas de fluxos de trabalho. O RUP foi desenvolvido em conjunto com a UML (*Unified Process Language*) e cada fluxo de trabalho está ligado a modelos da UML.
- *Perspectiva prática*: sugere boas práticas para serem utilizadas ao longo do processo. São definidas 6 práticas principais para o desenvolvimento de software: desenvolver software iterativamente, gerenciar os requisitos, utilizar arquiteturas baseadas em componentes, utilizar modelos visuais para o software e verificar a qualidade do software.

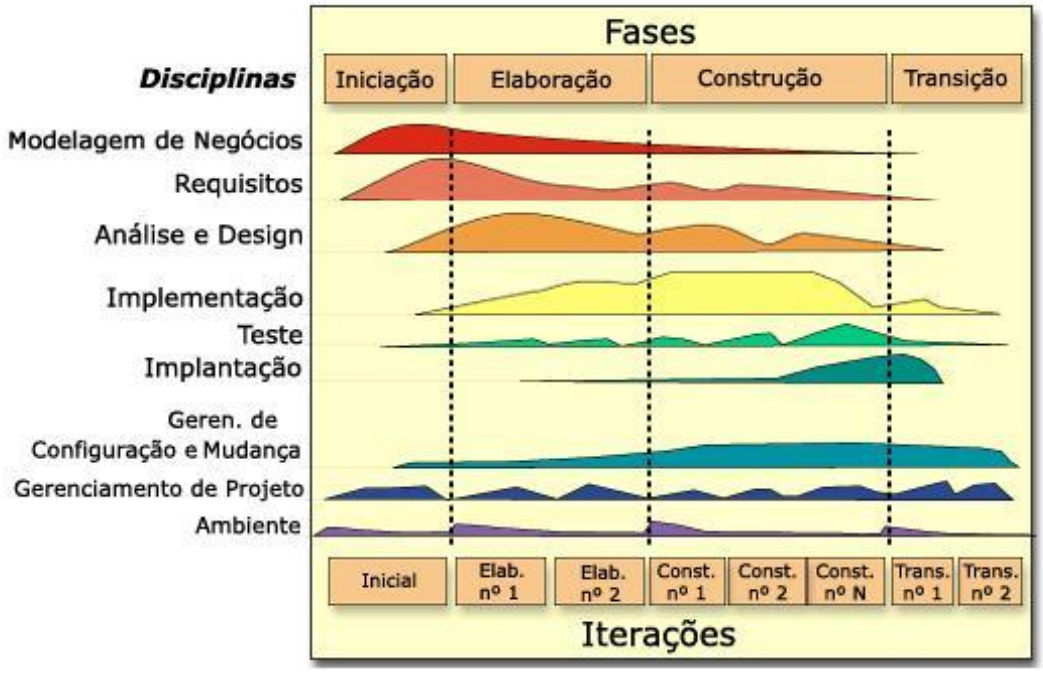


Figura 2 - Modelo RUP
Fonte: Wikipedia (2014)

Sommervile (2011) afirma que as inovações mais importantes do RUP são as separações de fases e fluxos de trabalho, e o reconhecimento de que a implantação do sistema no ambiente do cliente faz parte do processo. As fases são dinâmicas e possuem objetivos. Os workflows são estáticos e são atividades técnicas que não estão associadas

a um única fase, mas podem ser utilizados ao longo do desenvolvimento para que cada fase possa atingir seus objetivos.

Uma característica importante do RUP, derivada do seu uso extensivo da UML, está no fato de os requisitos serem modelados na forma de Casos de Uso. Segundo Bezerra (2007), casos de uso (do inglês *use case*) é a especificação de uma sequencia completa de interações entre um sistema e um ou mais agentes externos a esse sistema, representando um relato de uso de certa funcionalidade do sistema em questão, sem revelar a estrutura e o comportamento internos desse sistema.

2.2 Métodos Ágeis de Desenvolvimento de Software

Segundo Sommerville (2011), nas décadas de 80 e 90 acreditava-se que a melhor maneira de construir software era através de planejamentos cuidadosos, garantia de qualidade formalizada, o uso de ferramentas CASE para análise e design e processos controladores e rigorosos. Esta crença era derivada da engenharia de software utilizada para construir grandes sistemas com logos ciclos de vida.

Ainda de acordo com Sommerville (2011), este tipo de trabalho envolve grande esforço de planejamento, *design* e documentação. Este esforço é justificável quando envolve a coordenação de grandes equipes de desenvolvimento, quando o sistema é crítico e quando a quantidade de pessoas envolvida na manutenção do software será muito grande. Porém, quando este tipo de método é aplicado ao desenvolvimento de softwares pequenos e médios, todo o esforço de planejamento e documentação acaba dominando o processo de desenvolvimento do software, fazendo com que mais tempo seja gasto na especificação do sistema do que no desenvolvimento e nos testes.

A insatisfação com essa abordagem tradicional de desenvolvimento de software, segundo Sommerville (2011), levou alguns desenvolvedores a buscar métodos "ágeis". Os métodos ágeis baseiam-se no desenvolvimento incremental de software e são mais adequados para casos em que os requisitos mudam frequentemente durante o desenvolvimento. Estes métodos visam diminuir burocracias no processo evitando trabalho que tem valor a longo prazo apenas e eliminando documentações que provavelmente nunca serão utilizadas. Segundo Sabbagh (2013), uma definição comum para ágil pode ser: "que se movimenta com facilidade, ligeiro, leve". Este nome, ágil, foi escolhido para representar um movimento que surgiu em meados da década de 90 em resposta aos métodos tradicionais que eram predominantes na época. Os métodos

tradicionais são fortemente prescritivos e focados em planos detalhados que são elaborados no início do projeto que definem custo, escopo e tempo, em micro gerenciamento e na centralização do poder, além de processos complexos e extensa documentação. Segundo Sutherland (2010), em menos de uma década o Scrum tornou-se um dos frameworks de desenvolvimento de software mais utilizados e conhecidos no mundo. O Scrum é ágil porque, assim como outros métodos, metodologias e frameworks, ele deve seguir as premissas e princípios do Manifesto Ágil (Beck *et al.*, 2001).

O Scrum, de acordo com Sabbagh (2013), é um framework para gestão de processos, ou seja, é uma estrutura básica que serve de suporte à gestão de projetos e que deve ser expandida através do uso prático desta própria estrutura. O Scrum não define práticas específicas e detalhadas para serem seguidas, ele acredita que existem práticas específicas para cada contexto de projeto e por isso não existe um conjunto de práticas que seja adequado a todos os casos. Por ser um framework, o Scrum pode ser combinado com diferentes métodos e práticas que podem ser experimentados e adaptados pelo time de acordo com cada contexto.

Leffingwell (2010) lista as seguintes práticas como sendo as principais práticas do Scrum:

- O trabalho é realizado em "sprints", que são iterações limitadas por tempo, elas têm 30 dias ou menos de duração.
- O trabalho dentro de uma sprint é fixo. Uma vez que o escopo da Sprint foi definido, nenhuma funcionalidade adicional pode ser introduzida, a não ser pelo time de desenvolvimento.
- Todo o trabalho a ser feito é chamado de *product backlog*, o qual possui os novos requisitos a serem entregues, os defeitos a serem corrigidos e as atividades de infraestrutura e projeto.
- O Scrum Master é o mentor dos times responsáveis pela entrega de resultados bem-sucedidos em cada sprint.
- O Product Owner faz o papel de representante do cliente.
- Uma reunião diária, com todos de pé, é o principal método de comunicação.
- Um grande foco é dado às limitações de tempo (*timeboxing*). Sprints, reuniões em pé, reuniões de revisão de entregas, entre outros, são todos concluídos em tempos prescritos.

- A orientação típica do Scrum é de sprints fixas de 30 dias, com aproximadamente 3 sprints por release, suportando assim releases de mercado incrementais em prazos de 90 dias

O Scrum não prescreve nenhuma forma de descrição de requisitos. No entanto, uma forma de descrição de requisitos muito comum quando se fala em desenvolvimento ágil é o uso de histórias (Teles, 2005), porém também podem ser utilizados Casos de Uso para compor o backlog do produto e do sprint. Não existe uma formalização para a documentação de histórias e é comum que seu registro seja descartado depois de sua implementação, porém, a prática de BDD tem alcançado grande popularidade entre os times ágeis, e realizando os testes de aceitação de maneira automatizada as histórias acabam sendo documentadas de uma forma em que seu registro é mantido.

Segundo Wynne e Hellesoy (2012), a maioria dos projetos de software envolve times de várias pessoas trabalhando juntas de maneira colaborativa, então uma comunicação de alta qualidade é fundamental para o seu sucesso. Uma boa comunicação não significa apenas comunicar suas ideias eloquentemente para outros; também é preciso ter *feedback* para garantir que a compreensão foi correta. Essa é uma das razões para que times ágeis trabalhem em pequenos incrementos, utilizando o software construído de forma incremental como feedback que diz aos *stakeholders:* "É isso o que você disse?".

Porém, mesmo a utilização de metodologias ágeis para aproximar o cliente da equipe de desenvolvimento muitas vezes não é suficiente. Se os desenvolvedores passarem uma iteração de duas semanas implementando um mal-entendido, não somente desperdiçaram duas semanas de esforço, mas eles corromperam o código fonte com conceitos e funcionalidades que não refletem a ideia original, ou seja, mesmo com a proximidade que os princípios ágeis introduzem no processo de desenvolvimento de software, ainda assim desenvolvedor e cliente continuam falando línguas diferentes e enfrentando problemas de comunicação (Wynne e Hellesoy, 2012).

Um dos mecanismos que podem ser utilizados para instituir uma linguagem comum entre os times de desenvolvimento e seus clientes, fazendo assim com que a comunicação entre ambos melhore, é o uso do Gherkin (GitHub, 2013). A próxima seção apresenta o uso do Gherkin para o desenvolvimento ágil de requisitos e os demais conceitos por trás da sua definição.

2.3 Desenvolvimento Ágil de Requisitos utilizando o Gherkin

Segundo Eric Evans (2003) um projeto encara sérios problemas quando sua linguagem é dividida. Experts do domínio usam seu jargão enquanto os membros técnicos do time tem sua própria linguagem para discutir o domínio em termos de design. Por essa divisão linguística, os experts do domínio descrevem o que querem vagamente e os desenvolvedores, lutando para entender um domínio novo para eles, entendem vagamente.

De acordo com Prikladnick *et al.* (2014) é comum que os times de desenvolvimento tentem impor seus termos aos especialistas no domínio. São muitos os casos de clientes falando sobre banco de dados, índices, tabelas, servidores, etc. Isso acontece devido à falhas na comunicação. Com isso, os especialistas no domínio podem tentar sobrepor-se aos desenvolvedores, definindo como fazer tecnicamente, em vez de explicar o domínio. O ideal é que os desenvolvedores façam uma ponte inicial que abstraia os especialistas no domínio a parte técnica. Para isso é preciso que eles determinem uma terminologia, e consequentemente uma linguagem apropriada e baseada no domínio.

Segundo Evans (2003), uma linguagem ubíqua é uma linguagem baseada no domínio e que é utilizada para representar modelos e documentações, de forma que estes sejam compreendidos pelo cliente, analista, projetista, desenhista, testador, gerente, etc. Além disso esta linguagem deve ser utilizada na comunicação entre a equipe de desenvolvimento e os especialistas no domínio, ou seja, é a linguagem utilizada no dia-a-dia por todos os envolvidos no projeto. De acordo com Prikladnick *et al.* (2014), para que desenvolvedores e especialistas no domínio falem a mesma língua, é preciso que ambos utilizem os termos em sua rotina diária. Para os especialistas no domínio isso é natural e já acontece, já para os desenvolvedores isso pode ser um desafio. Para que a linguagem torne-se uma linguagem ubíqua é preciso utilizá-la e tudo o que é produzido: documentação, diagramas e principalmente no código.

A linguagem ubíqua está presente em todas as partes do desenvolvimento e é utilizada por todos os envolvidos com o objetivo de eliminar a necessidade de traduções. As traduções fazem com que termos e detalhes sejam perdidos, utilizando a

mesma linguagem isso não acontece, porém, para a utilização da mesma linguagem por todos, muitas vezes é necessário aprendizado (Prikladnick *et al.*, 2014).

2.3.1 Testes de Aceitação Automatizados e Desenvolvimento Orientado por Comportamento

Uma linguagem ubíqua no desenvolvimento de software não é utilizada apenas para descrever os requisitos em si. Pode ser utilizada também para a descrição de testes de aceitação automatizados quando se trabalha com Desenvolvimento Orientado a Testes (*Test-Driven Development – TDD*) (Wynne e Hellesoy, 2012).

De acordo com Aniche (2014), o TDD é uma técnica de desenvolvimento de software que segue a seguinte mecânica: escrever um teste que falha, fazê-lo passar da maneira mais simples possível e, por fim, refatorar o código. Esse ciclo é conhecido como Ciclo Vermelho-Verde-Refatora. Ao iniciar o desenvolvimento de uma funcionalidade, o desenvolvedor a divide em pequenas tarefas. Essas tarefas exigem a escrita de código. Ao praticar TDD, o desenvolvedor antes de começar a realizar a codificação de uma funcionalidade, explicita os objetivos da funcionalidade. Isto é realizado por meio de testes automatizados. O teste nada mais é do que um trecho de código que deixa claro o que outro determinado trecho de código deve fazer. Ao formalizar esse objetivo na forma de um teste automatizado, esse teste falha, claro; afinal, a funcionalidade ainda não foi implementada. O desenvolvedor então trabalha para fazer esse teste passar através da implementação da funcionalidade. Quando o teste passa, é iniciada próxima etapa no ciclo, a refatoração. Refatorar consiste na atividade de melhorar o código que já está escrito.

Segundo Wynne e Hellesoy (2012), os testes de aceitação automatizados são chamados assim porque eles expressam o que o software deve fazer para que o cliente o ache aceitável. Ao invés de passar os requisitos para o time de desenvolvimento sem muita oportunidade de *feedback*, o desenvolvedor e o cliente colaboram para escrever testes automatizados que expressam os resultados que o cliente deseja. Na primeira vez que o teste é escrito ele falha ao ser executado, porque não há codificação ainda. No entanto, ele já captura as necessidades do cliente e dá a todos um sinal claro do que deve ser codificado.

Ainda segundo Wynne e Hellesoy (2012), esses testes são diferentes de testes unitários, que são feitos para os desenvolvedores e os ajudam a checar e corrigir o projeto (*design*) de software. Algumas vezes é dito que testes unitários garantem que

você está construindo a coisa corretamente, enquanto testes de aceitação garantem que você está construindo a coisa correta.

North (2012) afirma que o BDD (Behavior-Driven Development) é o mesmo que TDD, porém voltado para uma audiência maior: testadores, analistas, gerentes de projeto. O BDD se aplicado em equipes formadas exclusivamente por programadores será exatamente igual ao TDD, porém se isto não é verdade, então o BDD se torna uma coisa diferente, maior: tona-se a comunicação entre todas essas partes interessadas para criar uma única visão coerente e para a qual as entregas são voltadas.

North (2006) explica que o BDD foi criado porque sempre eram encontrados mal entendidos quando se aplicava ou ensinava TDD e a maior parte destes problemas de entendimento era derivada do uso da palavra "teste". Dessa forma, uma nova visão sobre o TDD foi introduzida, focando no comportamento do sistema através de testes e não nos testes em si.

O Desenvolvimento Orientado por Comportamento (*Behaviour-Drivem Development – BDD*), de acordo com Wynne e Hellesoy (2012), baseia-se no Desenvolvimento Orientado a Testes (*Test-Driven Development – TDD*) por meio da formalização dos bons hábitos dos melhores praticantes de TDD. Os melhores praticantes de TDD, segundo os autores, trabalham de fora para dentro, começando com uma falha de testes de aceitação do cliente que descreve o comportamento do sistema de acordo com o ponto de vista do cliente. Praticantes de BDD tomam o cuidado de escrever testes de aceitação como exemplos que qualquer um do time possa ler.

2.3.2 Cucumber

Segundo Wynne e Hellesoy (2012), Cucumber é uma ferramenta de linha de comando utilizada em desenvolvimento orientado a comportamento, que quando executada, lê as especificações de documentos de texto chamados *features*, examina-os para cenários de teste, e roda os cenários contra o seu sistema. Cada cenário é uma lista de passos para o Cucumber trabalhar. Para que o Cucumber possa entender esses arquivos de *features*, eles devem seguir algumas regras básicas de sintaxe. O nome desse conjunto de regras é Gherkin (que será detalhado na subseção seguinte). Os testes de aceitação automatizados através do Cucumber possuem o benefício de documentos de especificação tradicionais, os quais podem ser escritos e lidos por *stakeholders* do negócio, mas eles têm a vantagem distinta de que podem ser interpretados

computacionalmente. Na prática, isso quer dizer que a sua documentação, ao invés de ser escrita uma vez e gradualmente ficar obsoleta, torna-se uma coisa viva que reflete o estado verdadeiro do projeto. Para muitos times, os testes Cucumber passam a ser uma "fonte de verdade" definitiva do que o sistema faz. Ter um único lugar para procurar essa documentação economiza tempo que usualmente é desperdiçado tentando manter documentos de requisitos, testes e código sincronizados. Isso também ajuda a construir confiança no time, porque diferentes partes do time não tem mais sua própria versão da verdade.

Juntamente com as *features*, também é necessário fornecer ao Cucumber um conjunto de definição dos passos (*step definitions*), os quais mapeiam a linguagem legível do negócio de cada passo para código a fim de carregar qualquer ação descrita pelo passo. Em um conjunto de testes maduro, as definições dos passos serão apenas uma ou duas linhas de código que direcionam a execução para uma biblioteca de código de suporte, especificamente para o domínio da sua aplicação, que sabem como realizar tarefas comuns no sistema. Normalmente, isso envolverá uma biblioteca de automação para interagir com o sistema em si.

Essa hierarquia pode ser vista na imagem a seguir:

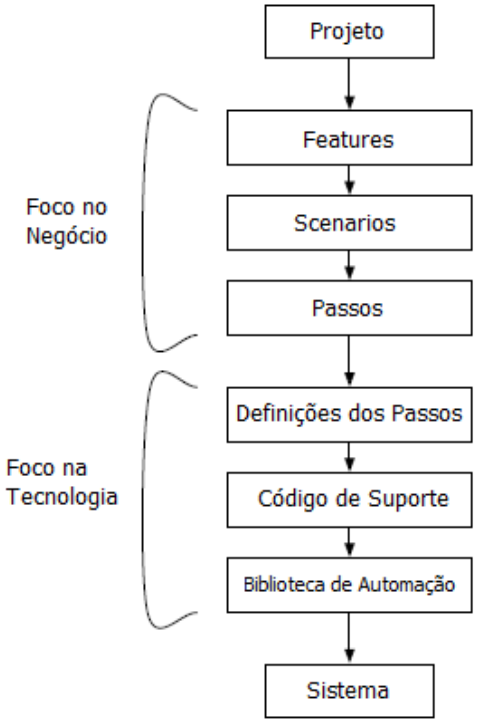


Figura 3 - Pilha de teste do Cucumber

Fonte: Adaptado de (Wynne e Hellesoy, 2012)

Se o código da definição do passo executa sem erros, então o Cucumber passa ao próximo passo do cenário. Se ele chegar ao fim do cenário sem que nenhum passo falhe, então o cenário é marcado como tendo passado. Caso algum dos passos falhe, o cenário é marcado como tendo falhado, e então passa ao cenário seguinte. Conforme os cenários são executados, o Cucumber imprime os resultados mostrando exatamente o que está funcionando e o que está falhando (Wynne e Hellesoy, 2012).

Ainda segundo Wynne e Hellesoy (2012), existem muitas vantagens que fazem do Cucumber uma excelente escolha: é possível escrever especificações em mais de quarenta línguas diferentes, usar etiquetas – tags – para organizar e agrupar os cenários e você integrar com uma série de bibliotecas de automação para executar quase qualquer tipo de aplicação.

2.3.3 Gherkin

De acordo com a Comunidade Open Source de Cucumber no GitHub (2013) o Gherkin é uma Linguagem de Domínio Específico que permite a descrição do comportamento do software sem o detalhamento de como este comportamento deve ser implementado. O Gherkin serve para dois propósitos: documentação e automação de testes.

O site Cukes.Info (2014) diz que o Gherkin é uma pequena linguagem de computação com uma sintaxe bem definida, mas que é tão simples que não é preciso saber programar para que se possa utilizá-la. Wynne e Hellesoy (2012) afirmam que o Gherkin é uma linguagem que pode ser utilizada para escrever especificações de software que servem tanto para os clientes como para testes utilizando Cucumber.

A seguir serão discutidas os principais elementos utilizadas no Gherkin de acordo com Wynne e Hellesoy (2012).

- *Feature*: cada arquivo Gherkin começa com a palavra chave Feature. O texto imediatamente após a palavra feature, na mesma linha, é o nome da funcionalidade e as linhas subsequentes são a descrição da mesma. É possível utilizar qualquer palavra na descrição desde que não se use no início de nenhuma linha as palavras Scenario, Background ou Scenario Outline. A descrição pode ter múltiplas linhas. É um ótimo lugar para colocar detalhes

sobre quem vai utilizar a funcionalidade, e porque, ou para colocar links para suportar a documentação como wireframes ou user research surveys. É comum nomear o arquivo da feature convertendo o nome da feature para caracteres minúsculos e trocando os espaços por underscores. Então, por exemplo, uma feature chamada User logs teria o arquivo user_logs.feature. Em Gherkin, uma feature deve ser seguida por um dos seguintes termos: Scenario, Background ou Scenario Outline.

- Scenario: Para expressar realmente o comportamento desejado, cada feature contém vários cenários. Cada cenário é um exemplo concreto de como o sistema deve comportar-se em determinada situação. Somando o comportamento definido por todos os cenários temos o comportamento esperado da feature. Ao escrever um cenário devemos ter em mente que cada cenário deve ter sentido e poder ser executado de forma independente de outros cenários. Todo cenário segue o padrão: obter o sistema a um estado particular; estimular o sistema; examinar o novo estado. No Gherkin são utilizadas as seguintes palavras chave para identificar esse comportamento dos cenários: *Given*, *Then*, *When*, *And*, e *But*.

- *Comentários*: além das descrições de cada feature, dos cenários e dos passos, também podem ser colocados comentários em qualquer parte do documento, desde que iniciem com "#" e sejam a única coisa presente na linha onde foram colocados. As descrições devem ser utilizadas para colocar informações diretamente relevantes para os clientes e os comentários para informações técnicas direcionadas ao time de desenvolvimento. É possível colocar os detalhes técnicos nas descrições, porém, para tal, deve ser certificado de que o cliente se sente confortável com essas informações.

- *Background*: é uma sessão da feature destinada a colocar informações necessárias a todos os cenários. É utilizada para evitar a repetição de informações nos cenários.

- *Scenario Outline*: é utilizado para quando existem cenários que têm a mesma estrutura e o mesmo propósito, porém possuem entradas e saídas diferentes. Dessa forma é criado um cenário genérico e colocado entre os símbolos "< >"

palavras que identifiquem as entradas e saídas e em seguida é colocada a seção "Examples" onde são colocadas em formato de tabela as entradas e saídas que devem ser testadas e as colunas são identificadas de acordo com as palavras entre "< >" apresentadas no *Scenario Outline*.

A Figura 4 apresenta um exemplo de Feature descrita utilizando-se o Gherkin.

```
Feature: Feedback when entering invalid credit card details

  In user testing we've seen a lot of people who made mistakes
  entering their credit card. We need to be as helpful as possible
  here to avoid losing users at this crucial stage of the
  transaction.

  Background:
    Given I have chosen some items to buy
    And I am about to enter my credit card details

  Scenario: Credit card number too short
    When I enter a card number that's only 15 digits long
    And all the other details are correct
    And I submit the form
    Then the form should be redisplayed
    And I should see a message advising me of the correct number of digits
```

Figura 4 - Exemplo de Feature
Fonte: Wynne e Hellesoy (2012)

2.4 Considerações Finais

Neste capítulo foi apresentada uma visão geral sobre o desenvolvimento de software tradicional e ágil, em seguida foram abordados os conceitos de TDD e BDD e linguagem ubíqua. Por fim foram introduzidos conceitos sobre a ferramenta Cucumber e a linguagem Gherkin. A fundamentação teórica apresentada neste capítulo serviu de base para todo o desenvolvimento deste trabalho.

3 Avaliação do Detalhamento de Requisitos Através de Features e de Casos de Uso

Este capítulo tem como objetivo elucidar os detalhes da pesquisa realizada, o material utilizado, o método de pesquisa, o público alvo e os resultados.

O objetivo do deste trabalho foi realizar uma avaliação comparativa entre o detalhamento de requisitos utilizando Casos de Uso e utilizando Features escritas de acordo com o padrão definido pela linguagem Gherkin. A expectativa é que esta abordagem possa ser utilizada em diferentes metodologias de desenvolvimento de software e que quando aplicada em times ágeis ela também fornece uma opção para o registro formal de histórias. Através da avaliação é possível verificar se o uso de features como documentação de requisitos é uma abordagem válida e sabendo que sim é possível utilizá-la como opção alternativa ao uso de casos de uso e com a intenção de melhorar a comunicação entre a equipe de desenvolvimento e os clientes.

3.1 Método de Pesquisa

Para que a avaliação proposta neste trabalho pudesse ser feita, foram seguidas as seguintes etapas:

1. Identificação da organização onde a avaliação seria feita e o escopo dos requisitos a serem utilizados;
2. Análise do perfil profissional dos participantes;
3. Preparação da documentação;
4. Aplicação da pesquisa;

5. Análise dos resultados.

A seguir cada uma destas etapas será detalhada.

3.1.1 Identificação da organização onde a avaliação seria feita e o escopo dos requisitos a serem utilizados

A documentação utilizada foi baseada em casos de uso de uma organização pública do ramo de energia. A organização utiliza atualmente um processo de desenvolvimento de software baseado no RUP, a documentação dos requisitos é realizada através de casos de uso e documentos complementares como os documentos de regras de negócio, glossário e especificação complementar, por exemplo.

O sistema escolhido foi o Sistema de Controle de Permissões, que é identificado na organização pela sigla ZIS1 (sigla esta que é proveniente de políticas internas de administração da organização). Este sistema foi escolhido porque entre os possíveis participantes existiam indivíduos que conhecem ou não o domínio do sistema, que participaram ou não do seu desenvolvimento e que trabalham ou não na organização, permitindo assim diferentes percepções sobre a documentação a ser analisada.

Através do envolvimento da equipe de desenvolvimento do sistema ZIS1 foi possível utilizar uma linguagem ubíqua, uma vez que todos os participantes da pesquisa que estavam familiarizados com o sistema conheciam já os termos utilizados, uma vez que o desenvolvimento estava em estágio avançado já quando a pesquisa foi executada, o que foi comprovado pela ausência de problemas no entendimento dos requisitos por parte destes participantes. Entretanto, como o estudo não alcançou os usuários do ZIS1, a ubiquidade da linguagem não pode ser garantida para toda a equipe envolvida, apenas para os indivíduos que participaram da pesquisa. No total foram utilizados quatro casos de uso escolhidos de acordo com a sua complexidade: dois de complexidade simples, um de média e um de alta complexidade.

3.1.2 Análise do perfil profissional dos participantes

Inicialmente foram identificados 28 possíveis participantes da pesquisa. Para analisar o perfil profissional de cada um dos participantes em potencial foi aplicado um questionário de perfil profissional (ver Anexo I) para coletar dados da função exercida atualmente, experiência profissional, escolaridade, conhecimento sobre casos de uso, BDD e Gherkin, e conhecimento sobre o domínio do sistema do qual os casos de uso foram adaptados.

Em seguida foram analisadas as informações de perfil profissional que seriam relevantes para definir os grupos de participantes do estudo. Os grupos foram definidos de forma a ficarem com homogeneidade de funções desempenhadas atualmente, conhecimento de casos de uso e conhecimento do domínio (ver Tabela 1). Para a divisão foram consideradas as funções: analista, programador/testador, gerente e arquiteto. Algumas pessoas não responderam qual a ocupação atual, apenas as ocupações passadas, mas estas não foram levadas em consideração durante as análises.

Tabela 1 - Divisão dos Participantes Por Grupos

Participante	Grupo	Analista	Programador	Gerente	Arquiteto	Caso Uso	Domínio do ZIS1
Possível Participante 1	G1	X	X	X	X	conheço/já usei	conheço
Possível Participante 2	G2			X		conheço muito/domino	não conheço
Possível Participante 3	G1					conheço/já usei	não conheço
Possível Participante 4	G1		X			conheço/já usei	conheço
Possível Participante 5	G2		X		X	conheço/já usei	conheço
Possível Participante 6	G1	X				conheço muito/domino	não conheço
Possível Participante 7	G1	X	X	X	X	conheço/já usei	não conheço
Possível Participante 8	G2	X	X	X		conheço/já usei	não conheço
Possível Participante 9	G1	X	X	X		conheço muito/domino	conheço
Possível Participante 10	G2					conheço/já usei	não conheço
Possível Participante 11	G2		X		X	conheço muito/domino	conheço parcialmente
Possível Participante 12	G2			X		conheço muito/domino	conheço parcialmente
Possível Participante 13	G2			X		conheço/já usei	conheço parcialmente
Possível Participante 14	G2		X			conheço/já usei	não conheço
Possível Participante 15	G1		X			conheço muito/domino	conheço parcialmente
Possível Participante 16	G1					conheço muito/domino	não conheço
Possível Participante 17	G1		X			já ouvi falar	conheço parcialmente
Possível Participante 18	G1			X		conheço/já usei	conheço parcialmente
Possível Participante 19	G1	X		X		conheço muito/domino	não conheço
Possível Participante 20	G2	X				conheço muito/domino	não conheço
Possível Participante 21	G1			X		conheço/já usei	não conheço
Possível Participante 22	G2	X				conheço muito/domino	não conheço

Participante	Grupo	Analista	Programador	Gerente	Arquiteto	Caso Uso	Domínio do ZIS1
Possível Participante 23	G1	X		X		conheço muito/domino	não conheço
Possível Participante 24	G2		X			conheço/já usei	não conheço
Possível Participante 25	G2	X		X	X	conheço muito/domino	conheço parcialmente
Possível Participante 26	G1		X	X	X	conheço/já usei	não conheço
Possível Participante 27	G2	X	X	X	X	conheço muito/domino	conheço
Possível Participante 28	G2	X	X			conheço/já usei	conheço

3.1.3 Preparação da documentação

Com o objetivo de avaliar o uso das *features* no detalhamento dos requisitos a pesquisa foi composta pela seguinte documentação:

- Descrições de Casos de Uso;
- *Features*;
- Documentação complementar para a leitura dos casos de uso;
- Glossário;
- Termo de Consentimento;
- Questionário de avaliação da documentação.

Os documentos de casos de uso, regras de negócio e especificação complementar foram adaptados para a pesquisa através das seguintes ações:

- Logotipos e textos que identificavam a organização que forneceu a documentação foram removidos;
- O caso de uso Administrar Cadastro de Grupo inicialmente era o caso de uso Administrar Cadastro de Órgão, como Grupo é um subitem de órgão o caso de uso teve os fluxos relativos a órgão removidos e deixados apenas os relativos a grupo, isto foi feito para enxugar o caso de uso que antes era muito extenso. O conceito de grupo não foi alterado, apenas foi adaptada a abordagem em que o objeto grupo é tratado no sistema, porém ele continuou sendo dependente de órgão;
- As ações executadas em relação a grupo inicialmente não continham consulta e detalhamento de grupo, pois isto era realizado nos fluxos de órgão, então foram inseridas estas ações no caso de uso;

Foi mantido o padrão utilizado pela organização nos casos de uso, este padrão possui algumas peculiaridades que foram mantidas e explicadas aos participantes da pesquisa que não estavam familiarizados com ele:

- Os fluxos de exceção não possuem este nome nos casos de uso, são chamados de fluxos alternativos assim como os fluxos alternativos do caso de uso.
- O que são chamadas “Mini-estruturas de Dados” são os campos que devem ser exibidos na tela e suas propriedades.

Em seguida foram escritas as features para os casos de uso, utilizando os padrões da linguagem Gherkin e com base nos seguintes critérios:

- Cada passo de cada fluxo do caso de uso é um passo em um scenario de uma feature;
- Informações técnicas como comportamentos, mensagens de log e mini-estruturas de dados e requisitos não funcionais dos casos de uso foram colocadas como comentários nas features;
- Regras de negócio e mensagens simples foram colocadas nas descrições das features ou de scenarios ou passos nas features.

Foram inseridos defeitos na documentação a ser aplicada aos participantes, a Tabela 2 mostra os defeitos introduzidos e a qual dupla de caso de uso e features correspondentes ao caso de uso cada defeito é aplicável.

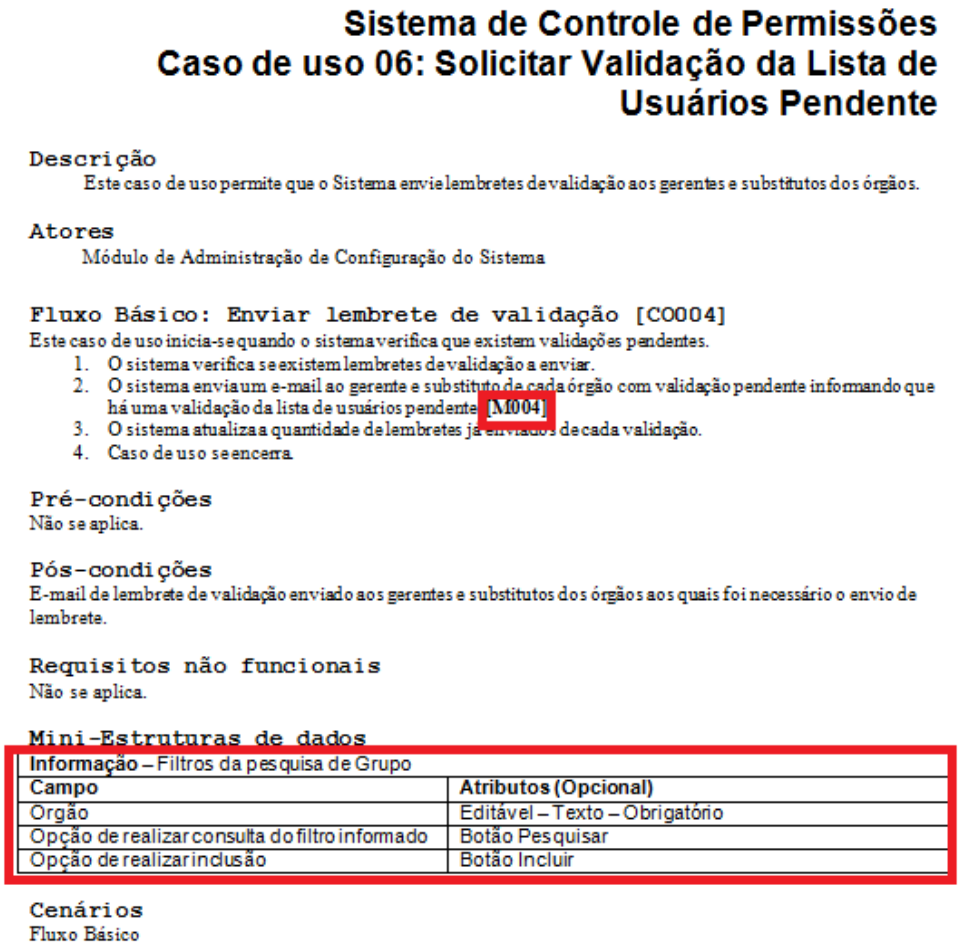
Tabela 2 - Defeitos Introduzidos na Documentação

Controle de Defeitos		
Aplicável à	**Defeito**	**Descrição do Defeito**
Caso de Uso 03 e Features Correspondentes	D1	Falta de referência para a mensagem M007 no primeiro passo do fluxo ou scenario que trata de "Campos Obrigatórios Não Preenchidos"
	D2	Requisito não funcional ambíguo: "A interface deve ser amigável."
Caso de Uso 04 e Features Correspondentes	D3	Regra de Negócio incorreta no segundo passo do fluxo ou scenario que trata de "Solicitar Validação da Lista de Usuários". A regra colocada é: "Ao excluir um órgão: se ele não possuir nenhuma validação seu registro é excluído do sistema. Caso contrário, o seu status é alterado para “desativado”.", porém a regra correta seria: "Não poderá haver mais de uma validação pendente por órgão. Uma nova validação suspende todas as validações pendentes.".
	D4	Atores que não executam nada no caso de uso: Administrador e Usuário Especial (Gerente e substituto)
Caso de Uso 06 e Features Correspondentes	D5	A mini-estrutura de dados "Informação – Filtros da pesquisa de Grupo" foi colocada. Ela não pertence a este requisito.
	D6	No segundo passo do fluxo ou scenario "Enviar Lembrete de Validação" a mensagem está para "Órgão cadastrado com sucesso." ao invés de "Gerente, existe uma lista de usuários pendente para validação. Acesse o link para realizar a validação.[Colocar um link redirecionando para a lista de usuários dos grupos do órgão]".
Caso de Uso 07 e Features	D7	O comportamento CO005 foi alterado para iniciar com "A tela deve ser carregada rapidamente"

Controle de Defeitos		
Aplicável à	**Defeito**	**Descrição do Defeito**
Correspondentes	D8	Retirados todos os passos do fluxo ou scenario: Remover Usuário
Defeitos Fixos do Maior Caso de Uso		
Caso de Uso 03	D9	Ao final do Fluxo A1 existe o passo: "O caso de uso retorna ao passo 6 do Fluxo Alternativo A2.", que é inconsistente com o caso de uso.
Features Correspondentes ao Caso de Uso 03	D10	A feature "Apagar Grupo" apresenta o scenario "Editar Grupo".

Os defeitos foram distribuídos de acordo com a Tabela 2, porém, no caso do maior dos casos de uso, que é o “Administrar Cadastro de Grupo” foram inseridos também dois defeitos fixos, um no caso de uso e um em uma de suas features. Isto foi feito porque como se trata de uma documentação mais extensa os defeitos poderiam passar sem serem percebidos mais facilmente e então optamos por colocar alguns defeitos a mais.

A Figura 5 e a Figura 6 mostram um exemplo de caso de uso e um exemplo de feature utilizados, respectivamente. Em ambas as figuras os erros introduzidos estão destacados em vermelho.



Sistema de Controle de Permissões
Caso de uso 06: Solicitar Validação da Lista de Usuários Pendente

Descrição
Este caso de uso permite que o Sistema envie lembretes de validação aos gerentes e substitutos dos órgãos.

Atores
Módulo de Administração de Configuração do Sistema

Fluxo Básico: Enviar lembrete de validação [CO004]
Este caso de uso inicia-se quando o sistema verifica que existem validações pendentes.
1. O sistema verifica se existem lembretes de validação a enviar.
2. O sistema envia um e-mail ao gerente e substituto de cada órgão com validação pendente informando que há uma validação da lista de usuários pendente [M004]
3. O sistema atualiza a quantidade de lembretes já enviados de cada validação.
4. Caso de uso se encerra.

Pré-condições
Não se aplica.

Pós-condições
E-mail de lembrete de validação enviado aos gerentes e substitutos dos órgãos aos quais foi necessário o envio de lembrete.

Requisitos não funcionais
Não se aplica.

Mini-Estruturas de dados

Informação – Filtros da pesquisa de Grupo	
Campo	**Atributos (Opcional)**
Orgão	Editável – Texto – Obrigatório
Opção de realizar consulta do filtro informado	Botão Pesquisar
Opção de realizar inclusão	Botão Incluir

Cenários
Fluxo Básico

Figura 5 - Exemplo de Caso de Uso Utilizado

A decisão de qual defeito seria aplicado a qual participante foi realizada de modo que os defeitos fossem aplicados aos dois grupos e que cada participante recebesse defeitos diferentes nos casos de uso e nas features. Os defeitos fixos foram aplicados ao caso de uso 03 e a uma de suas features, portanto todos os participantes do grupo G2 receberam estes defeitos.

Feature: Solicitar Validação da Lista de Usuários Pendente

In order to enviar lembretes de validação aos gerentes e substitutos dos órgãos.

As an Módulo de Administração de Configuração do Sistema

I want to Enviar lembretes

Background:

Given A rotina de verificação de validações pendentes foi iniciada

Scenario: Enviar lembrete de validação

 # As rotinas periódicas do sistema são executadas de 6 em 6 horas.

When o sistema verifica que existem validações pendentes;

 # Filtros de pesquisa de grupo

 # Órgão – Editável – Texto - Obrigatório

 # Opção de realizar consulta do filtro informado – Botão Pesquisar

 # Opção de realizar inclusão – Botão Incluir

And O sistema verifica se existem lembretes de validação a enviar.

Then O sistema envia um e-mail ao gerente e substituto de cada órgão com validação pendente informando que há uma validação da lista de usuários pendente.

 # Órgão cadastrado com sucesso.

And O sistema atualiza a quantidade de lembretes já enviados de cada validação.

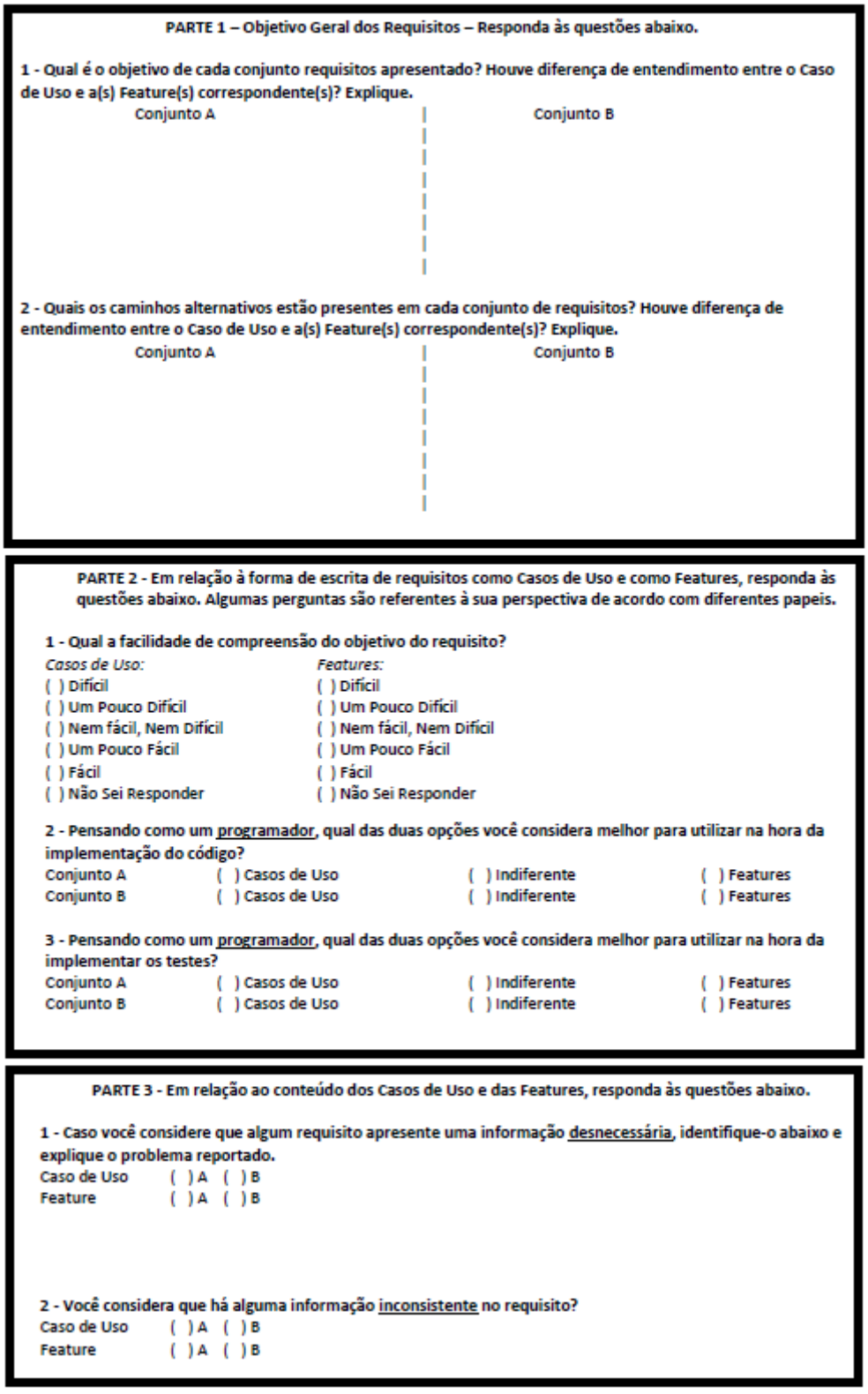
Figura 6 - Exemplo de Feature Utilizada

A Tabela 3 apresenta a distribuição de defeitos por participante e por grupo.

Tabela 3 - Distribuição de Defeitos

Participante	Grupo	Caso de Uso	Features
Participante 1	G1	D3 e D5	D4 e D6
Participante 2	G2	D2 e D8	D1 e D7
Participante 3	G2	D2 e D8	D1 e D7
Participante 4	G1	D3 e D5	D4 e D6
Participante 5	G2	D2 e D8	D1 e D7
Participante 6	G2	D1 e D7	D2 e D8
Participante 7	G2	D1 e D7	D2 e D8
Participante 8	G1	D4 e D6	D3 e D5
Participante 9	G1	D4 e D6	D3 e D5
Participante 10	G1	D3 e D5	D4 e D6
Participante 11	G2	D1 e D7	D2 e D8
Participante 12	G1	D4 e D6	D3 e D5
Participante 13	G1	D4 e D6	D3 e D5
Participante 14	G2	D2 e D8	D1 e D7
Participante 15	G2	D1 e D7	D2 e D8

Além dos casos de uso e das features foi elaborado um formulário de avaliação da qualidade do material (Figura 7).



PARTE 1 – Objetivo Geral dos Requisitos – Responda às questões abaixo.

1 - Qual é o objetivo de cada conjunto requisitos apresentado? Houve diferença de entendimento entre o Caso de Uso e a(s) Feature(s) correspondente(s)? Explique.

Conjunto A | Conjunto B

2 - Quais os caminhos alternativos estão presentes em cada conjunto de requisitos? Houve diferença de entendimento entre o Caso de Uso e a(s) Feature(s) correspondente(s)? Explique.

Conjunto A | Conjunto B

PARTE 2 - Em relação à forma de escrita de requisitos como Casos de Uso e como Features, responda às questões abaixo. Algumas perguntas são referentes à sua perspectiva de acordo com diferentes papeis.

1 - Qual a facilidade de compreensão do objetivo do requisito?

Casos de Uso:	*Features:*
( ) Difícil	( ) Difícil
( ) Um Pouco Difícil	( ) Um Pouco Difícil
( ) Nem fácil, Nem Difícil	( ) Nem fácil, Nem Difícil
( ) Um Pouco Fácil	( ) Um Pouco Fácil
( ) Fácil	( ) Fácil
( ) Não Sei Responder	( ) Não Sei Responder

2 - Pensando como um programador, qual das duas opções você considera melhor para utilizar na hora da implementação do código?

Conjunto A	( ) Casos de Uso	( ) Indiferente	( ) Features
Conjunto B	( ) Casos de Uso	( ) Indiferente	( ) Features

3 - Pensando como um programador, qual das duas opções você considera melhor para utilizar na hora da implementar os testes?

Conjunto A	( ) Casos de Uso	( ) Indiferente	( ) Features
Conjunto B	( ) Casos de Uso	( ) Indiferente	( ) Features

PARTE 3 - Em relação ao conteúdo dos Casos de Uso e das Features, responda às questões abaixo.

1 - Caso você considere que algum requisito apresente uma informação desnecessária, identifique-o abaixo e explique o problema reportado.

Caso de Uso	( ) A	( ) B
Feature	( ) A	( ) B

2 - Você considera que há alguma informação inconsistente no requisito?

Caso de Uso	( ) A	( ) B
Feature	( ) A	( ) B

Figura 7 - Partes do Questionário de Avaliação

O formulário foi dividido em três partes, cada uma com um objetivo de avaliação: a Parte 1 possuía o objetivo de verificar se o participante teve diferença de entendimento sobre o requisito documentado em caso de uso para o requisito documentado em feature, na Parte 2 as perguntas foram direcionadas a entender a facilidade de compreensão de cada documentação e a preferência de documentação de acordo com papéis e atividades comuns em processos de desenvolvimento de software, por fim, na Parte 3 o objetivo foi analisar em qual documentação os participantes teriam maior facilidade em encontrar discrepâncias.

3.1.4 Aplicação da Pesquisa

A pesquisa foi aplicada em 15 participantes dos 28 que responderam ao questionário de perfil profissional, isto aconteceu devido à falta de disponibilidade de algumas pessoas para a pesquisa e também porque as dificuldades de aplicar a pesquisa remotamente impediram que alguns indivíduos participassem por estarem geograficamente afastados. A Tabela 4 mostra os indivíduos que de fato participaram da pesquisa, suas características e divisão por grupos.

Tabela 4 - Características dos Participantes da Pesquisa

Participante	Grupo	Analista	Programador	Gerente	Arquiteto	Casos de Uso	Domínio do ZIS1
Participante 1	G1		X			conheço/usei	conheço
Participante 2	G2		X		X	conheço/usei	conheço
Participante 3	G2	X	X	X		conheço/usei	não conheço
Participante 4	G1	X	X	X		conheço muito/domino	conheço
Participante 5	G2					conheço/usei	não conheço
Participante 6	G2			X		conheço muito/domino	conheço parcialmente
Participante 7	G2			X		conheço/usei	conheço parcialmente
Participante 8	G1		X			conheço muito/domino	conheço parcialmente
Participante 9	G1			X		conheço/usei	conheço parcialmente
Participante 10	G1	X		X		conheço muito/domino	não conheço
Participante 11	G2	X				conheço muito/domino	não conheço
Participante 12	G1			X		conheço/usei	não conheço
Participante 13	G1	X		X		conheço muito/domino	não conheço
Participante 14	G2		X			conheço/usei	não conheço
Participante 15	G2	X	X	X	X	conheço muito/domino	conheço

Para a realização da pesquisa cada participante recebeu dois conjuntos de detalhamento de requisitos, identificados como “Conjunto A” e “Conjunto B”. Cada conjunto possuindo um caso de uso e uma ou mais features que representava(m) o mesmo requisito, a Tabela 5 apresenta a distribuição dos conjuntos. Além disso, o participante também recebeu o Termo de Consentimento (Anexo XIX), o Glossário (Anexo IX), o documento de Regras de Negócio (Anexo VIII), o documento de Especificação Complementar (Anexo VII) e o questionário de avaliação (Anexo II).

Tabela 5 - Conjuntos de Requisitos Detalhados

Grupo	Conjunto A	Conjunto B
G1	CSU 04 e as features correspondentes	CSU 06 e as features correspondentes
G2	CSU 03 e as features correspondentes	CSU 07 e as features correspondentes

Em seguida, cada participante foi orientado sobre a documentação. As orientações fornecidas foram as seguintes: “Você está participando de uma pesquisa sobre detalhamento de requisitos, para tal, você está recebendo dois conjuntos de detalhamento de requisitos, o Conjunto A e o Conjunto B. Cada conjunto possui a sua identificação escrita na primeira página. O Conjunto é composto pelo detalhamento de requisito em formato de Caso de Uso e em formato de Features. Os casos de uso foram adaptados de um sistema real, porém, não refletem o domínio real deste sistema. Os casos de uso possuem algumas peculiaridades que foram mantidas, como o fato de que fluxos de exceção são tratados como fluxos alternativos, isto não deve ser considerado um erro, foi mantido do template original. As features são representações de funcionalidades e em alguns casos um caso de uso pode estar relacionado a várias features. As features são compostas por palavras chave que estão destacadas em roxo e em negrito, estas palavras estão em inglês, caso possua alguma dúvida em relação à tradução informe. As partes em verde, e que iniciam com “#” são comentários, assim como é utilizado em linguagens de programação. A feature possui uma descrição inicial, uma parte em seguida que tem informações adicionais e é iniciada pela palavra Background e um ou mais Scenarios. Um Scenario é um conjunto de passos que leva o sistema de um estado a outro. A documentação de regras de negócio e especificação complementar deve ser utilizada em auxílio na leitura dos casos de uso, pois estes possuem referências que remetem a estes documentos. O documento de Glossário serve tanto para Features como para Casos de Uso, nele você pode tirar dúvidas sobre elementos do domínio. É necessário que você leia a documentação e responda ao

questionário. Não se preocupe com respostas certas ou erradas. É possível que a documentação contenha erros, caso encontre algum, reporte nas questões específicas do questionário. Caso tenha alguma dúvida, pergunte, porém, só poderemos responder caso a informação solicitada não influencia na pesquisa. Por favor marque no horário de início no questionário o horário em que começar a ler a documentação e no horário de fim o momento em que terminar de responder às questões. Ao final do questionário existe um espaço para que você faça quaisquer considerações, sugestões e/ou críticas sobre a documentação e sobre a pesquisa que achar relevantes. Obrigada pela participação."

Uma vez orientados sobre a pesquisa, os participantes possuíam tempo livre para executá-la e tinham a liberdade de fazê-lo na presença do aplicador da pesquisa ou não. Todos ficaram livres para tirar dúvidas sobre a pesquisa e pelo menos metade solicitou maiores orientações em relação às perguntas do questionário, principalmente da Parte 1.

Alguns participantes apresentaram dúvidas em relação ao domínio da aplicação, por conhecerem o domínio total ou parcialmente, apresentaram dificuldades de absorver as alterações realizadas no caso de uso 03.

A receptividade dos participantes em relação à pesquisa foi boa, porém, a maioria indicou insatisfação com o volume de documentação a ser analisado.

Na próxima seção será apresentada a avaliação dos resultados obtidos.

3.2 Avaliação dos Resultados

De acordo com as respostas obtidas no questionário as seguintes avaliações foram feitas, comparando o uso de casos de uso e de features, e serão descritas a seguir:

- Compreensão do Requisito;
- Facilidade de Entendimento;
- Preferência;
- Capacidade de Identificação de Defeitos.

3.2.1 Compreensão do Requisito

Esta avaliação buscou verificar se houve diferença de entendimento entre o requisito quando descrito em caso de uso e quando descrito em feature. Dos 15 participantes, apenas 1 afirmou ter detectado entendimento diferente entre uma feature e um caso de uso.

O participante relatou que o caso de uso 06 não citava validação de usuário pendente. Ao avaliar esta observação do participante, chegou-se à conclusão de que esta confusão foi causada porque neste caso de uso foi introduzida uma discrepância que fazia rastreabilidade de mensagem errada, fazendo com que a mensagem "Órgão cadastrado com sucesso." fosse referenciada no lugar onde deveria estar a referência para a mensagem "Gerente, existe uma lista de usuários pendente para validação. Acesse o link para realizar a validação.[Colocar um link redirecionando para a lista de usuários dos grupos do órgão]", ou seja, o participante ao ler o caso de uso não encontrou mensagem que informasse ao gerente a validação pendente. Ao identificar esta discrepância da documentação o participante a reportou como uma discrepância de inconsistência e também reportou como diferença de entendimento do requisito.

3.2.2 Facilidade de Entendimento

Quando indagados sobre a facilidade de entendimento da documentação, 27% dos participantes afirmou achar os casos de uso um pouco difíceis, ao passo que apenas 7% achou as features um pouco difíceis. Dessa maneira, mesmo sendo o primeiro contato de todos os participantes com a documentação em formato de features, poucos a apontaram dificuldades na compreensão. A Figura 8 e a Figura 9 mostram a distribuição percentual da opinião dos participantes sobre a facilidade de entendimento de cada documentação.

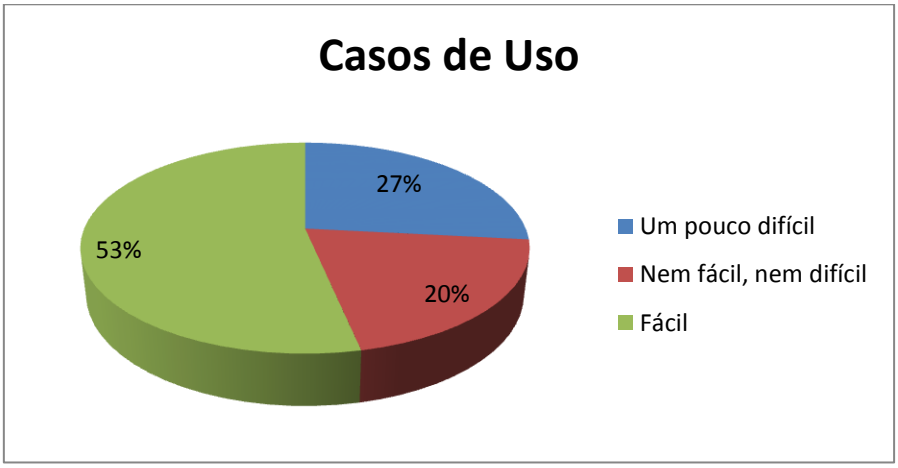


Figura 8 - Facilidade de Entendimento dos Casos de Uso

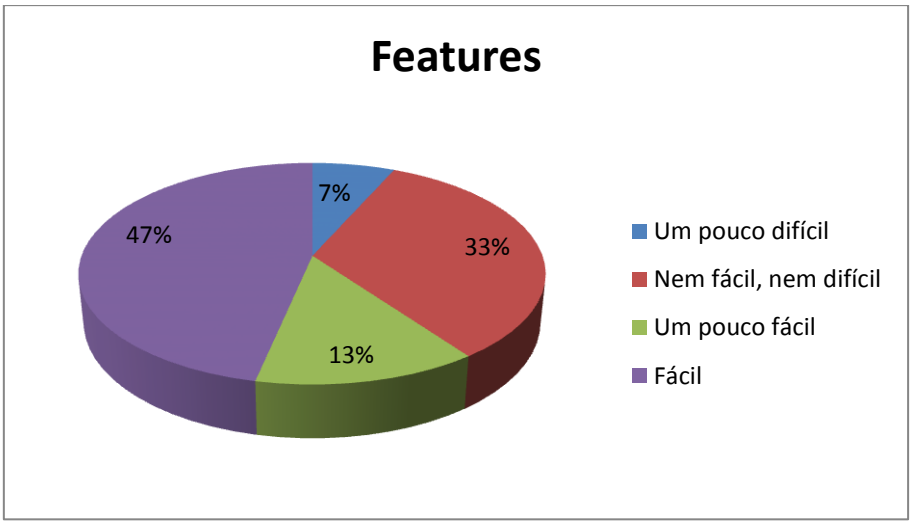


Figura 9 - Facilidade de Entendimento das Features

3.2.3 Preferência

Os participantes foram indagados sobre qual documentação é melhor de acordo com diferentes cenários. As perguntas foram feitas sob o ponto de vista dos papéis de programador, analista e membro da equipe de testes e foram abordadas atividades que são comuns em diferentes processos de desenvolvimento de software.

Quando perguntados sobre a preferência de documentação para a implementação do código, 47% disse preferir as features, 17% disse que é indiferente e 36% preferiu casos de uso. A maior popularidade das features de acordo com essa pergunta era esperada porque elas são muito mais próximas de linguagem de programação do que os casos de uso. A Figura 10 mostra o percentual de preferência para a implementação do código.

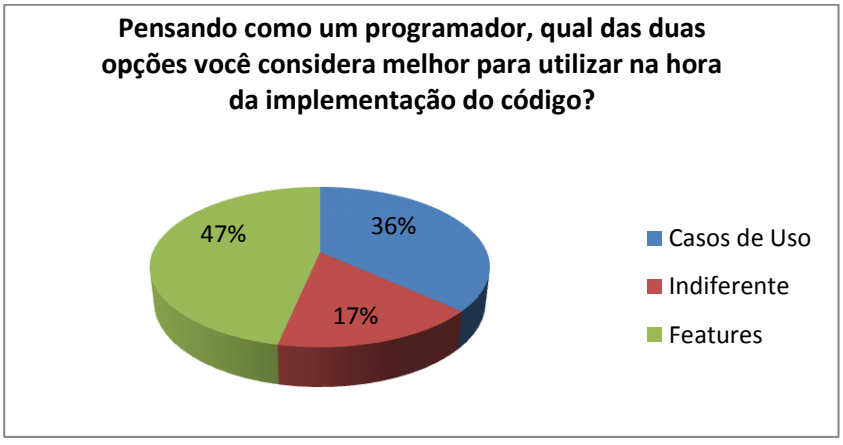


Figura 10 - Preferência de Programador para Implementar o Código

Quando perguntados sobre o ponto de vista de um programador para implementar os testes, a preferência ainda foi pelas features, com 43%, 17% foi apontou indiferença e 40% preferiu casos de uso, como mostrado na Figura 11.

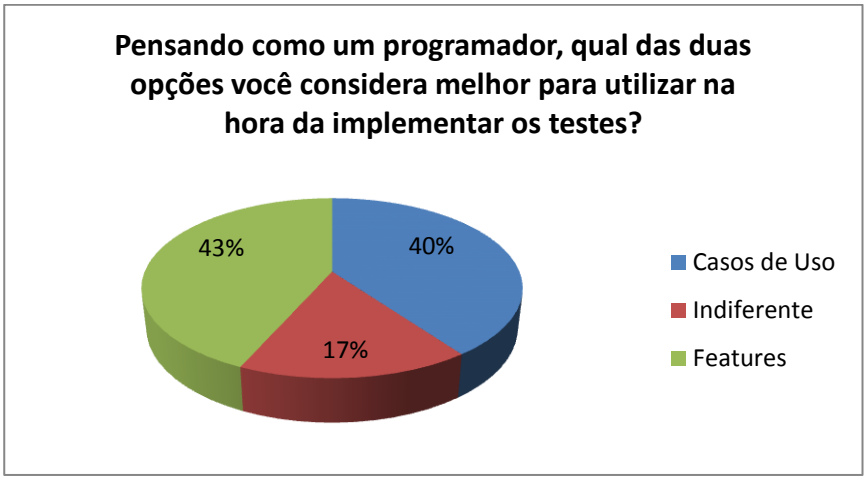


Figura 11 - Preferência de Programador para Implementar os Testes

Ainda do ponto de vista de um programador, a Figura 12 mostra que os participantes foram questionados sobre a preferência para entender o objetivo do requisito. Nesta questão, o percentual de indiferentes foi de 27%, enquanto casos de uso e features ficaram quase empatados, com 36 e 37% respectivamente. Encaramos isto como um resultado altamente positivo porque mostra que ambas as documentações conseguem expor o objetivo do requisito, sendo assim adequadas para o detalhamento de requisitos.

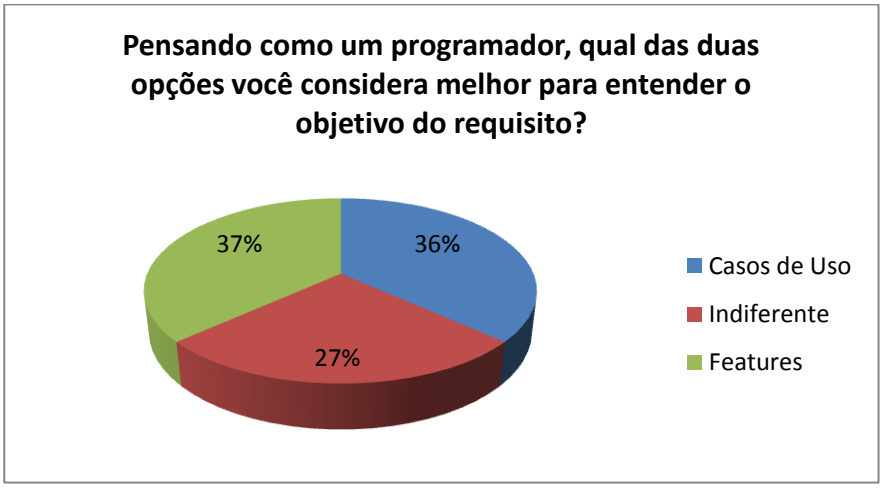


Figura 12 - Preferência de Programador para Entender o Objetivo do Requisito

Do ponto de vista de um membro da equipe de testes, para realizar a tarefa de definir os casos de teste da aplicação, em 43% dos casos foi preferido o uso de casos de uso (ver Figura 13). Esta informação é interessante porque, apesar de features terem sua

origem voltada para automação de testes de aceitação, elas se mostram menos adequadas, na opinião de muitos dos participantes, para outros tipos de testes.

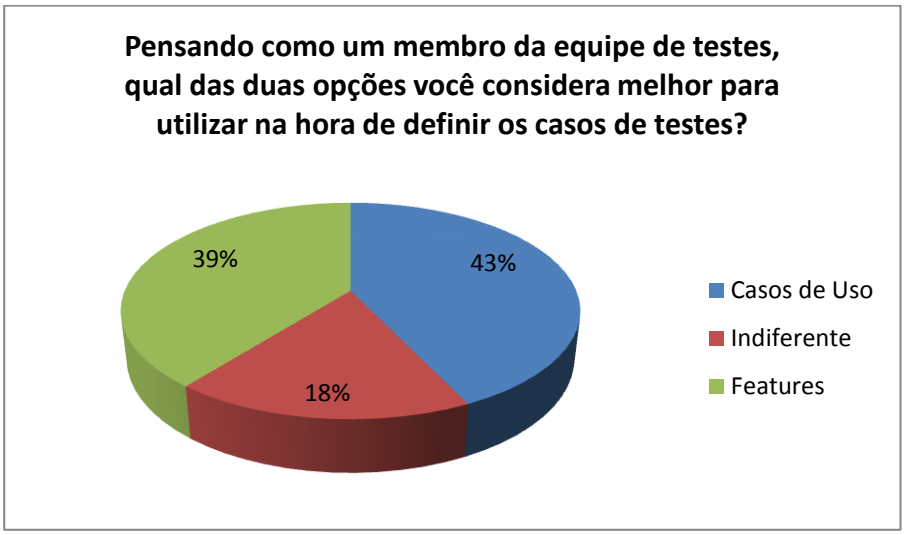


Figura 13 - Preferência de Membro da Equipe de Testes para Definir Casos de Teste

Apesar da preferência dos casos de uso para a elaboração dos casos de teste, na execução dos casos de teste as features foram apontadas como melhor opção em 46% dos casos, conforme podemos ver na Figura 14.

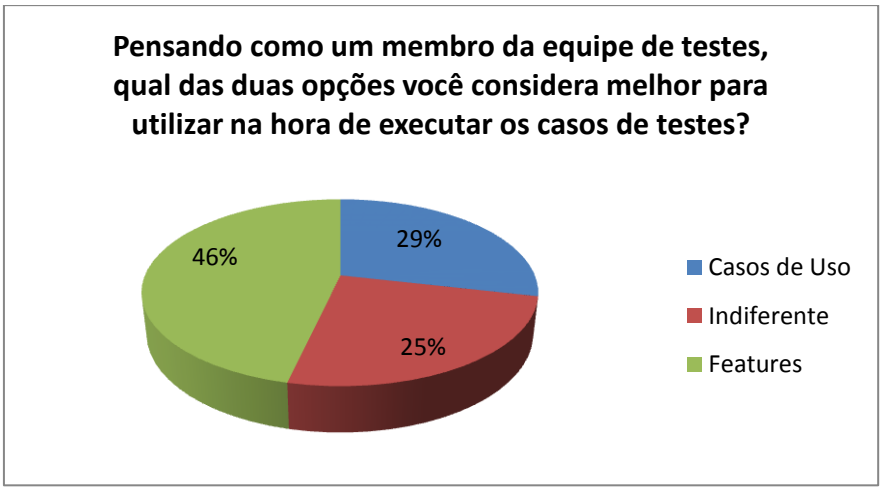


Figura 14 - Preferência de Membro da Equipe de Testes para Executar Casos de Teste

Pensando como membro da equipe de testes a maioria dos participantes achou as features melhores para entender o objetivo do requisito, a Figura 15 mostra os percentuais. Observou-se que 37% foi indiferente, reforçando a ideia de que ambos expressam o objetivo do requisito de forma adequada.

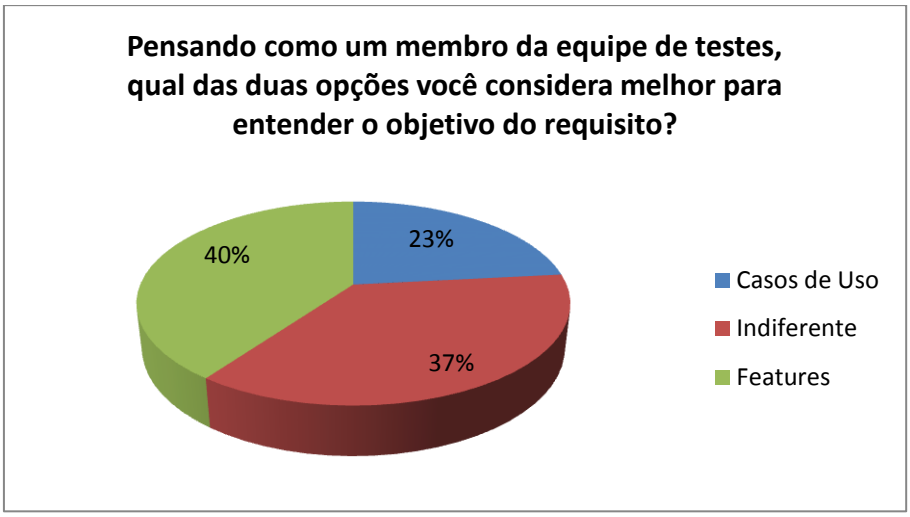


Figura 15 - Preferência de Membro da Equipe de Testes para Entender o Objetivo do Requisito

A mesma pergunta quando feita a partir do ponto de vista de um analista apresentou o maior percentual de preferência indicando indiferença, que foi de 53% conforme é mostrado na Figura 16.

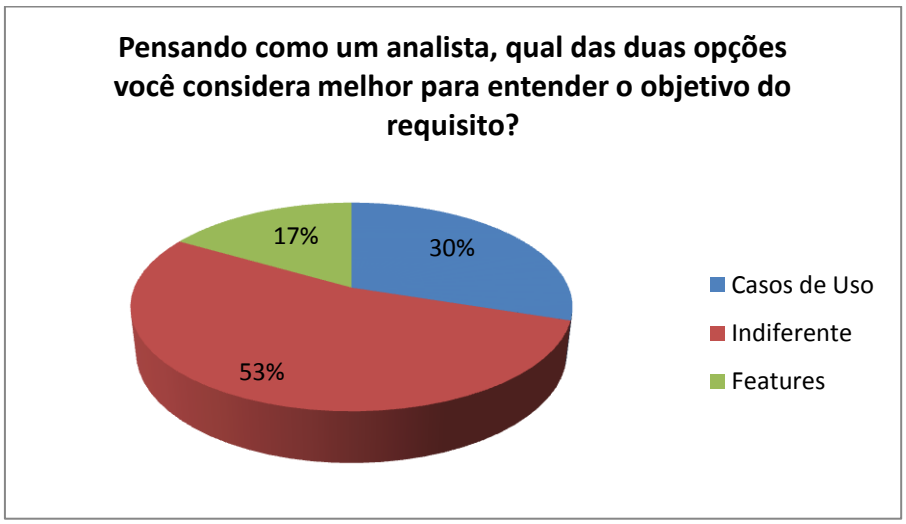


Figura 16 - Preferência de Analista para Entender o Objetivo do Requisito

A Figura 17 e a Figura 18 mostram que já para descrever o objetivo do requisito e documentá-lo, assumindo o papel de analista, a grande maioria acredita ser melhor a utilização dos casos de uso.

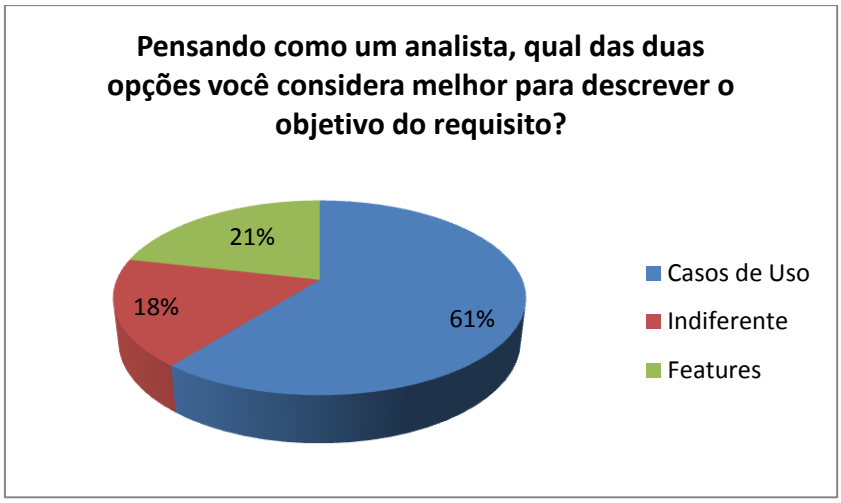


Figura 17 - Preferência de Analista para Descrever o Objetivo do Requisito

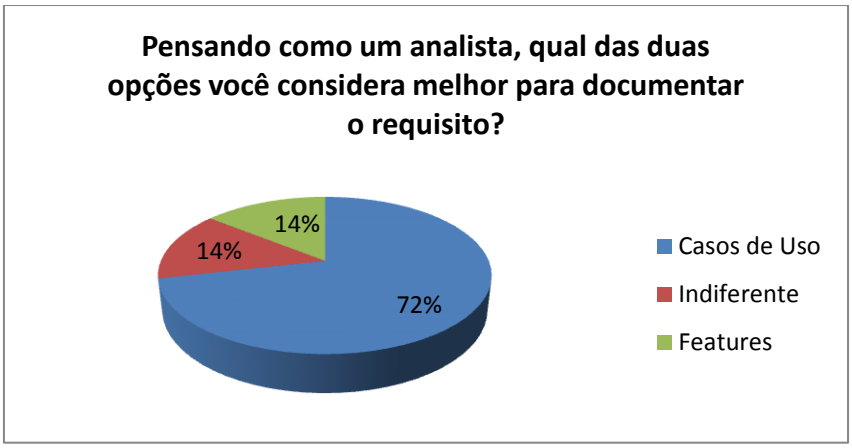


Figura 18 - Preferência de Analista para Documentar o Requisito

Quando indagados sobre a atividade do analista de passar os requisitos à equipe, os participantes expressaram opiniões mais divididas novamente, 43% ainda preferiu os casos de uso, mas 21% indicou indiferença entre as duas abordagens e ainda 36% preferiu as features, como mostrado na Figura 19. Isto mais uma vez reafirma a ideia de que as duas formas de documentação são válidas para o entendimento dos requisitos.

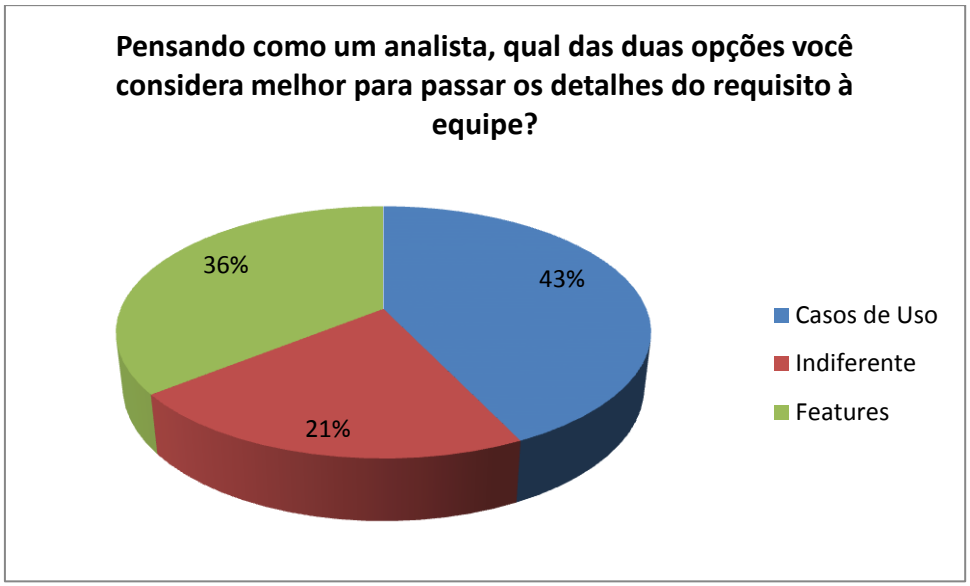


Figura 19 - Preferência de Analista para Passar os Detalhes do Requisito à Equipe

Um fator que fez falta nas features foi a rastreabilidade, pois quando indagados sobre a melhor maneira de descrever as regras de negócio, a preferência foi claramente dos casos de uso, uma vez que estes deixam as regras de negócio em um documento consolidado. A Figura 20 mostra que em nenhum dos casos foi indicada indiferença.

Figura 20 - Preferência de Analista para Descrever as Regras de Negócio

Para avaliar a opinião dos participantes sobre a representação de pré-condições sem influenciá-los utilizando este termo, que é comum nos casos de uso, foi apresentada a questão "Pensando como um analista, qual das duas opções você considera melhor para descrever o que deve ser garantido para que a funcionalidade representada pelo requisito possa ser executada?". Na maior parte dos casos foi preferido o uso de casos de uso, porém o percentual de preferência de features e de indiferença também foi alto,

mostrando que as features também são adequadas para a representação desta informação. Os percentuais são mostrados na Figura 21.

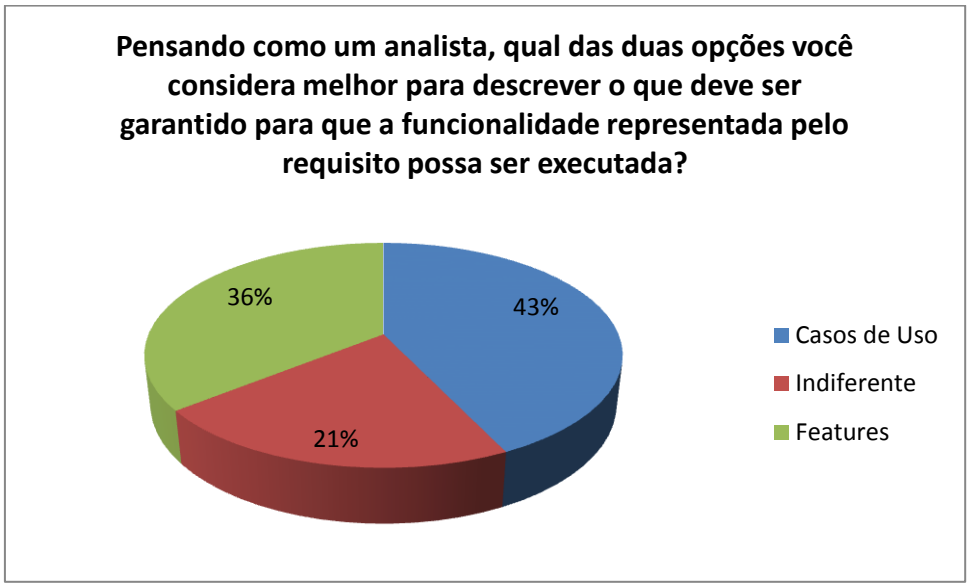


Figura 21 - Preferência de Analista para Descrever o que Deve Ser Garantido para que a Funcionalidade Seja Executada

Por fim, olhando a preferência de forma geral, mostrada na Figura 22, podemos observar que ela ficou bem dividida.

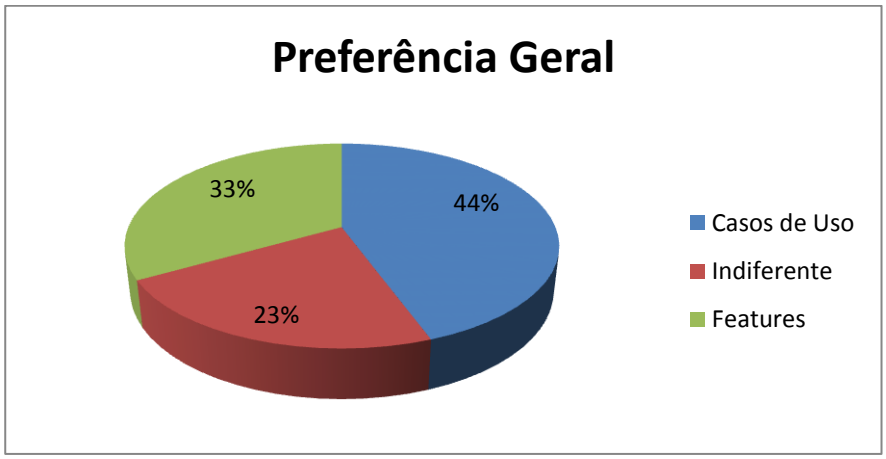


Figura 22 - Preferência Geral

Os casos de uso foram mais apontados como preferidos, o que era esperado visto que todos os participantes possuíam contato anterior com casos de uso e nenhum com features, mas o percentual de indiferença e de preferência das features mostra que essa abordagem teve boa aceitação entre os entrevistados.

3.2.4 Capacidade de Identificação de Defeitos

De acordo com a distribuição de defeitos realizada, cada participante recebeu no mínimo um defeito por caso de uso e um defeito nas features do caso de uso correspondente, sendo estes defeitos diferentes uns dos outros. A Figura 23 mostra a quantidade de defeitos identificados e o total de defeitos. A dificuldade de identificação dos defeitos foi quase a mesma nos dois casos, então pode-se dizer que há indícios que as features possuem aproximadamente a mesma adequação para inspeção que os casos de uso, ao menos para a equipe que participou do estudo.

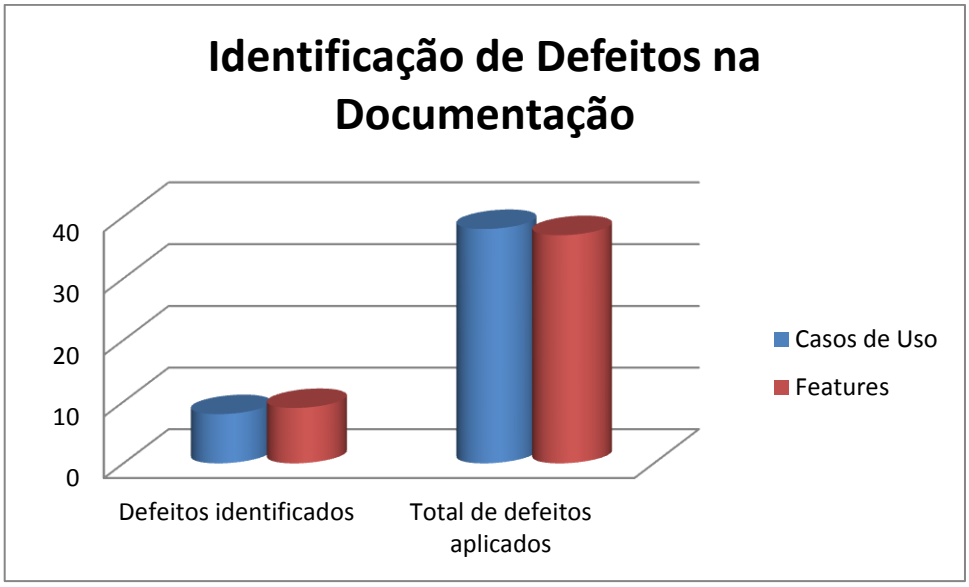


Figura 23 - Identificação de Defeitos na Documentação

Como a quantidade de defeitos identificados foi muito baixa, não foi possível realizar uma avaliação sobre quais defeitos foram identificados mais facilmente.

3.3 Considerações Finais

A pesquisa mostrou indícios, em sua avaliação, que as features apresentadas cumpriram o papel de documentar os requisitos de forma satisfatória. Alguns pontos negativos do uso de features levantados pelos participantes foram: a falta de rastreabilidade de regras de negócio e mensagens, a repetição de scenarios que são comuns a mais de uma feature e a dificuldade de elaborar casos de teste a partir das features. Por outro lado, muitos pontos positivos também foram indicados, como: a estruturação mais próxima de linguagem de programação, a consolidação das informações em um único documento para o desenvolvedor, entre outras. Como exemplo de observações temos um participante que escreveu: “Como programador,

achei interessante a descrição das features. O modo de apresentação me parece mais amigável, principalmente para entender o objetivo do requisito. [...] Como testador, confesso que fiquei em dúvida sobre qual alternativa seria melhor para utilizar na definição de casos de teste. Neste caso, os casos de uso parecem organizar e agrupar melhor as informações necessárias.”.

Além do exemplo supracitado, foi colocado por outro participante que “A feature sem os comentários não serve para descrever o requisito.[...]”, indicando que é necessária a presença de informações técnicas nas features para que estas possam ser utilizadas no detalhamento de requisitos.

A pesquisa mostrou-se promissora e passível de aplicação em outros conjuntos de participantes, organizações e para outros sistemas, porém, como feedback da pesquisa muitos apontaram a quantidade de documentação a ser analisada como um problema, o que nos leva a refletir sobre uma abordagem de pesquisa utilizando maior quantidade de participantes para que cada um receba menores quantidades de informações para processar durante a pesquisa.

4 Conclusão

4.1 Considerações finais

Este trabalho teve como objetivo avaliar de forma comparativa o detalhamento de requisitos através do uso de casos de uso e de documentação em linguagem ubíqua com Gherkin. Como base teórica para o trabalho foram apresentados conceitos de desenvolvimento de software tradicional e ágil, TDD e BDD, linguagem ubíqua, Cucumber e Gherkin. O trabalho realizou uma pesquisa que avaliou a compreensão do requisito, a facilidade de entendimento da documentação, a preferência pela forma de documentar e a capacidade de identificação de defeitos. Através da pesquisa constatamos ser viável a utilização de features para o detalhamento de requisitos, porém, a pesquisa possui limitações que devem ser analisadas a fim de realizar trabalhos futuros para dar continuidade ao estudo.

4.2 Limitações do Trabalho

O trabalho realizado têm limitações que devem ser levadas em consideração para avaliá-lo e para dar continuidade a este estudo, são estas:

- O número de participantes foi pequeno, apenas 15 participaram até o final da pesquisa;
- Nenhum dos participantes estava familiarizado com o estilo de documentação apresentado pelas features, nem com a escrita em Gherkin;
- Não foi possível avaliar a identificação de defeitos de forma relevante para afirmar qual das documentações seria mais adequada a este propósito;
- Não foi considerado se os participantes trabalham ou já trabalharam com metodologias ágeis;
- A pesquisa não foi aplicada aos usuários do sistema ZIS1;

- Tendo em mente que a pesquisa não envolveu os usuários do sistema, não podemos garantir que a linguagem utilizada é ubíqua para toda a equipe envolvida no desenvolvimento do ZIS1, mas apenas para aqueles que participaram da pesquisa, pois estes não apresentaram dificuldades quanto aos termos e o vocabulário utilizado na documentação e apresentaram entendimento dos requisitos documentados.

A fim de continuar o trabalho realizado em futuras pesquisas é preciso levar as limitações em consideração e traçar planos para superar as que forem julgadas mais relevantes.

4.3 Trabalhos Futuros

Durante o desenvolvimento do trabalho foram identificadas possibilidades de evoluir o estudo e expandir a aplicação da pesquisa para outros contextos e organizações. Além disso, uma vez que as features não foram inicialmente criadas com o intuito de documentar requisitos e sim para a realização de testes de aceitação automatizados, e tendo em vista que a aceitação da documentação em formato de features foi boa, é possível considerar o desenvolvimento de um framework ou metodologia para fornecer diretrizes para criação e manutenção destas documentações.

A continuação da pesquisa deve envolver a equipe completa de desenvolvimento para garantir a ubiquidade da linguagem e avaliar o entendimento das features principalmente pelos usuários, pois representam um papel fundamental na engenharia de requisitos e não foi possível alcança-los neste estudo.

É preciso também avaliar com maior cautela o uso de features na geração de casos de teste e o impacto da redução de rastreabilidade, visto que estas foram as principais críticas apresentadas pelos participantes da pesquisa.

Referências Bibliográficas

Aniche, M. (2014) "Test-Driven Development", http://tdd.caelum.com.br/. Acessado em 09 de Dezembro de 2014.

Beck, K. *et al.* (2001) "Manifesto for Agile Software Development", http://www.agilemanifesto.org/. Acessado em 20 de Outubro de 2014.

Bezerra, E. (2007), Princípios de Análise e Projeto de Sistemas com UML, Campus, 2ª edição.

Blackburn, M., Busser, R., Nauman, A. (2001) "Removing Requirement Defects and Automating Test", Software Productivity Consortium NFP, https://www.cmcrossroads.com/sites/default/files/article/file/2013/3988360.pdf. Acessado em 10 de Novembro de 2014.

Boehm, B. W., Basili, V.R. (2001), "Software Defect Reduction Top 10 List.", IEEE Computer 34 (1): 135-137.

Cukes.Info (2014) "Background and Credits", http://cukes.info. Acessado em 09 de Dezembro de 2014.

Cukes.Info (2014) "Gherkin", http://cukes.info/gherkin.html. Acessado em 09 de Dezembro de 2014.

Evans, E. (2003), "Domain-Driven Design: Tackling Complexity in the Heart of Software". http://www-public.int-evry.fr/~gibson/Teaching/CSC7322/ReadingMaterial/Evans03.pdf. Acessado em 15 de Novembro de 2014.

GitHub (2013) "Gherkin", https://github.com/cucumber/cucumber/wiki/Gherkin. Acessado em 09 de Dezembro de 2014.

Kotonya, G., Sommerville, I. (1998), Requirements Engineering: Process and Techniques; John Wiley & Sons.

Leffingwell, D. (2010), Agile Software Requirements: Lean Requirements Practices for Teams, Addison Wesley, 1st edition.

Leffinwell, D. (1997), Calculating the Return on Investment from More Effective Requirements Management; American Programmer 10(4); p.13-16.

North, D. (2006) "Introducing BDD", http://dannorth.net/introducing-bdd/. Acessado em 02 de Novembro de 2014.

North, D. (2012) "BDD is like TDD if...", http://dannorth.net/2012/05/31/bdd-is-like-tdd-if/. Acessado em 02 de Novembro de 2014.

Prikladnicki, R., Willi, R., Milani, F. (2014), Métodos Ágeis para Desenvolvimento de Software, Bookman, 1ª edição.

Programming", Dissertação de Mestrado em Informática da Universidade Federal do

Rio de Janeiro – UFRJ.

Sabbagh, R. (2013), Scrum: Gestão Ágil para Projetos de Sucesso, Casa do Código, 1ª edição.

Sommerville, Ian (2011), Software Engineering, Pearson Education, 9th edition.

Sutherland, Jeff (2010), Jeff Sutherland's Scrum Handbook, Scrum Training Institute Press.

Teles, V. M. (2005) "Um Estudo de Caso da Adoção das Práticas e Valores do Extreme

The Standish Group (1995), "Chaos Report", https://net.educause.edu/ir/library/pdf/NCP08083B.pdf. Acessado em 05 de Dezembro de 2014.

Wikipedia (2014), http://pt.wikipedia.org/wiki/IBM_Rational_Unified_Process, Acessado em 28 de Novembro de 2014.

Wynne, M., Hellesoy, A. (2012), The Cucumber Book, LLC, 1st edition.

Ye, W. (2013), Instant Cucumber BDD How-to, Packt Publishing, 1st edition.

ANEXO I – Questionário de Perfil Profissional

Esse questionário é parte do Trabalho de Conclusão de Curso do Bacharelado em Sistemas de Informação da UNIRIO, com foco em processos de software. Seu objetivo é avaliar a forma de descrição de requisitos por meio de Casos de Uso e de Features.

Em um primeiro momento é necessário coletar informações sobre o perfil dos profissionais que participarão da avaliação. Solicitamos sua colaboração, respondendo a algumas questões. Os dados de identificação não serão mencionados no relatório da pesquisa, o que preservará o anonimato e sigilo dos respondentes.

Se houver necessidade de maiores esclarecimentos, por favor envie um e-mail para os responsáveis:
Rafaela Sampaio – rafaela.sampaio@uniriotec.br
Gleison Santos - gleison.santos@uniriotec.br

Obrigado pela sua participação.

PARTE 1 - Caracterização de Perfil

Nome: _____

1 - Qual o seu nível de escolaridade?

( ) Ensino Médio ( ) Superior ( ) Pós Graduado
( ) Mestrado ( ) Doutorado

2 - Que papéis você já desempenhou em equipes de desenvolvimento de software:

Analista de Sistemas	( ) Atuo no momento	( ) Já atuei em projetos passados
Programador	( ) Atuo no momento	( ) Já atuei em projetos passados
Membro da Equipe de Testes	( ) Atuo no momento	( ) Já atuei em projetos passados
Gerente /Líder de Projetos	( ) Atuo no momento	( ) Já atuei em projetos passados
Arquiteto/Líder Técnico	( ) Atuo no momento	( ) Já atuei em projetos passados
Outro: _____	( ) Atuo no momento	( ) Já atuei em projetos passados

3 - Há quanto tempo você trabalha em desenvolvimento de software?

( ) nunca trabalhei ( ) menos de 1 ano ( ) de 1 a 3 anos
( ) de 3 a 5 anos ( ) mais de 5 anos

4 - Em quantos projetos de desenvolvimento de software aproximadamente você participou?

( ) nenhum projeto ( ) de 1 a 3 projetos ( ) de 3 a 5 projetos
( ) mais de 5 projetos

5 – Qual o seu conhecimento sobre as seguintes tecnologias?

UML / Casos de Uso

( ) Não conheço ( ) Já ouvi falar ( ) Conheço e Já usei ( ) Tenho muito conhecimento / Domino

BDD (Behavior Driven Development)

( ) Não conheço ( ) Já ouvi falar ( ) Conheço e Já usei ( ) Tenho muito conhecimento / Domino

Gherkin

( ) Não conheço ( ) Já ouvi falar ( ) Conheço e Já usei ( ) Tenho muito conhecimento / Domino

Outros: _____

( ) Não conheço ( ) Já ouvi falar ( ) Conheço e Já usei ( ) Tenho muito conhecimento / Domino

6 – Qual o seu conhecimento sobre o domínio do Sistema de Controle de Permissões (ZIS1)?

( ) Não conheço ( ) Conheço Parcialmente ( ) Conheço

7 – Você é membro da equipe responsável pelo desenvolvimento do Sistema de Controle de Permissões (ZIS1)?

( ) Sim ( ) Não

ANEXO II – Questionário de Avaliação da Documentação

Esse questionário é parte do Trabalho de Conclusão de Curso do Bacharelado em Sistemas de Informação da UNIRIO, com foco em processos de software. Seu objetivo é avaliar a forma de descrição de requisitos por meio de Casos de Uso e de Features.

Solicitamos sua colaboração, respondendo a algumas questões. Não há respostas certas ou erradas em relação a quaisquer dos itens. Os dados de identificação não serão mencionados no relatório da pesquisa, o que preservará o anonimato e sigilo dos respondentes.

Se houver necessidade de maiores esclarecimentos, por favor envie um e-mail para os responsáveis:
Rafaela Sampaio – rafaela.sampaio@uniriotec.br
Gleison Santos - gleison.santos@uniriotec.br

Obrigado pela sua participação.

Você está recebendo 2 conjuntos de requisitos. Cada conjunto é composto por 1 Caso de Uso e 1 ou mais Features. Leia-os e responda conforme solicitado. Por favor, marque o tempo gasto.

Nome: _____

Hora de Início: _____ Hora Fim: _____ Total: _____

PARTE 1 – Objetivo Geral dos Requisitos – Responda às questões abaixo.

1 - Qual é o objetivo de cada conjunto requisitos apresentado? Houve diferença de entendimento entre o Caso de Uso e a(s) Feature(s) correspondente(s)? Explique.

Conjunto A	Conjunto B

2 - Quais os caminhos alternativos estão presentes em cada conjunto de requisitos? Houve diferença de entendimento entre o Caso de Uso e a(s) Feature(s) correspondente(s)? Explique.

Conjunto A	Conjunto B

PARTE 2 - Em relação à forma de escrita de requisitos como Casos de Uso e como Features, responda às questões abaixo. Algumas perguntas são referentes à sua perspectiva de acordo com diferentes papeis.

1 - Qual a facilidade de compreensão do objetivo do requisito?

Casos de Uso:	*Features:*
( ) Difícil	( ) Difícil
( ) Um Pouco Difícil	( ) Um Pouco Difícil
( ) Nem fácil, Nem Difícil	( ) Nem fácil, Nem Difícil
( ) Um Pouco Fácil	( ) Um Pouco Fácil
( ) Fácil	( ) Fácil
( ) Não Sei Responder	( ) Não Sei Responder

2 - Pensando como um programador, qual das duas opções você considera melhor para utilizar na hora da implementação do código?

Conjunto A	( ) Casos de Uso	( ) Indiferente	( ) Features
Conjunto B	( ) Casos de Uso	( ) Indiferente	( ) Features

3 - Pensando como um programador, qual das duas opções você considera melhor para utilizar na hora da implementar os testes?

Conjunto A	( ) Casos de Uso	( ) Indiferente	( ) Features
Conjunto B	( ) Casos de Uso	( ) Indiferente	( ) Features

4 - Pensando como um programador, qual das duas opções você considera melhor para entender o objetivo do requisito?

Conjunto A	( ) Casos de Uso	( ) Indiferente	( ) Features
Conjunto B	( ) Casos de Uso	( ) Indiferente	( ) Features

5 - Pensando como um membro da equipe de testes, qual das duas opções você considera melhor para utilizar na hora de definir os casos de testes?

Conjunto A	( ) Casos de Uso	( ) Indiferente	( ) Features
Conjunto B	( ) Casos de Uso	( ) Indiferente	( ) Features

6 - Pensando como um membro da equipe de testes, qual das duas opções você considera melhor para utilizar na hora de executar os casos de testes?

Conjunto A	( ) Casos de Uso	( ) Indiferente	( ) Features
Conjunto B	( ) Casos de Uso	( ) Indiferente	( ) Features

7 - Pensando como um membro da equipe de testes, qual das duas opções você considera melhor para entender o objetivo do requisito?

Conjunto A	( ) Casos de Uso	( ) Indiferente	( ) Features
Conjunto B	( ) Casos de Uso	( ) Indiferente	( ) Features

8 - Pensando como um analista, qual das duas opções você considera melhor para entender o objetivo do requisito?

Conjunto A	( ) Casos de Uso	( ) Indiferente	( ) Features
Conjunto B	( ) Casos de Uso	( ) Indiferente	( ) Features

9 - Pensando como um analista, qual das duas opções você considera melhor para descrever o objetivo do requisito?

Conjunto A	( ) Casos de Uso	( ) Indiferente	( ) Features
Conjunto B	( ) Casos de Uso	( ) Indiferente	( ) Features

10 - Pensando como um analista, qual das duas opções você considera melhor para documentar o requisito?

Conjunto A	( ) Casos de Uso	( ) Indiferente	( ) Features
Conjunto B	( ) Casos de Uso	( ) Indiferente	( ) Features

11 - Pensando como um analista, qual das duas opções você considera melhor para passar os detalhes do requisito à equipe?

Conjunto A	( ) Casos de Uso	( ) Indiferente	( ) Features
Conjunto B	( ) Casos de Uso	( ) Indiferente	( ) Features

12 - Pensando como um analista, qual das duas opções você considera melhor para descrever as regras de negócio?

Conjunto A	( ) Casos de Uso	( ) Indiferente	( ) Features
Conjunto B	( ) Casos de Uso	( ) Indiferente	( ) Features

13 - Pensando como um analista, qual das duas opções você considera melhor para descrever o que deve ser garantido para que a funcionalidade representada pelo requisito possa ser executada?

Conjunto A	( ) Casos de Uso	( ) Indiferente	( ) Features
Conjunto B	( ) Casos de Uso	( ) Indiferente	( ) Features

PARTE 3 - Em relação ao conteúdo dos Casos de Uso e das Features, responda às questões abaixo.

1 - Caso você considere que algum requisito apresente uma informação desnecessária, identifique-o abaixo e explique o problema reportado.

Caso de Uso ( ) A ( ) B
Feature ( ) A ( ) B

2 - Você considera que há alguma informação inconsistente no requisito?

Caso de Uso ( ) A ( ) B
Feature ( ) A ( ) B

3 - Você considera que há alguma informação ambígua no requisito?

Caso de Uso ( ) A ( ) B
Feature ( ) A ( ) B

4 - Você considera que há alguma informação omissa no requisito?

Caso de Uso ( ) A ( ) B
Feature ( ) A ( ) B

5 - Você considera que há alguma informação estranha no requisito?

Caso de Uso ( ) A ( ) B
Feature ( ) A ( ) B

Considerações Gerais:

Neste espaço informe quaisquer considerações sobre o material, a pesquisa e o questionário que julgar serem relevantes.

ANEXO III – Caso de Uso 03 - Administrar Cadastro de Grupo

Sistema de Controle de Permissões
Caso de uso 03: Administrar Cadastro de Grupos de Permissões

Descrição

Este caso de uso permite que um Administrador do Sistema gerencie os grupos de permissões de cada órgão.

Atores

Administrador

Fluxo Básico: Consultar Grupo [CO002]

Este caso de uso inicia-se quando o usuário indica a intenção de pesquisar grupo.

1. O sistema apresenta os **Filtros de pesquisa de Grupo** e a opção:
    - Incluir Grupo [A1]
2. O usuário preenche e submete a pesquisa;
3. O sistema processa a pesquisa;
4. O sistema apresenta todos os itens do **Resultado da consulta de Grupo** conforme o filtro selecionado e as opções:
    - Detalhar Grupo
    - Incluir Grupo [A1]
    - Editar Grupo [A2]
    - Apagar Grupo [A3]
5. O usuário seleciona a opção Detalhar Grupo.
6. O sistema apresenta a **Lista de atributos de detalhes da consulta de grupo**;
7. O caso de uso se encerra.

Fluxo Alternativo A1: Incluir Grupo [CO002]

Este fluxo inicia-se quando o usuário indica a intenção de incluir grupo no passo 6 do Fluxo Alternativo A2;

1. O sistema solicita ao usuário o preenchimento dos dados da **Lista de atributos de detalhes da inclusão de Grupo**.
2. O usuário preenche os dados e solicita a inclusão;
3. O sistema valida e realiza a inclusão dos dados informados;**[RN 7.1.1] [A4][A5]**
4. O sistema exibe uma mensagem informando que a inclusão foi realizada com sucesso. **[M009]**
5. O caso de uso retorna ao passo 6 do Fluxo Alternativo A2.

Fluxo Alternativo A2: Editar Grupo [CO001]

Este fluxo inicia-se quando o usuário indica a intenção de Editar Grupo no passo 6do Fluxo Alternativo A2;

1. O sistema apresenta a **Lista de atributos de detalhes da alteração de Grupo**;
2. O usuário altera os dados desejados e solicita a alteração;
3. O sistema valida os dados informados; **[RN 7.1.1][A4] [A5]**
4. O sistema registra as alterações;
5. O sistema exibe uma mensagem de confirmação informando que a alteração foi realizada com sucesso; **[M012]**
6. O caso de uso se encerra.

Fluxo Alternativo A3: Apagar Grupo

Este fluxo inicia-se quando o usuário seleciona a opção Apagar um Grupo no passo 6 do Fluxo Alternativo A2;

1. O sistema solicita a confirmação para a exclusão;

2. O usuário confirma a exclusão;**[M014]**
3. O sistema valida e remove o Grupo confirmado;
4. O sistema exibe uma mensagem informando que a exclusão do Grupo foi realizada com sucesso; **[M011]**
5. O caso de uso se encerra.

Fluxo Alternativo A4: Campos obrigatórios não preenchidos

Este fluxo inicia-se quando o sistema detecta que o usuário não informou algum campo obrigatório na inclusão ou edição de grupo;

1. O sistema exibe uma mensagem de erro;
2. O caso de uso retorna ao passo anterior do fluxo chamador.

Fluxo Alternativo A5: String LDAP inválida

Este fluxo inicia-se quando o sistema detecta a quebra da regra **[RN 7.1.1];**

1. O sistema exibe uma mensagem de erro; **[M047]**
2. O caso de uso retorna ao passo anterior do fluxo chamador.

Pré-condições

Não se aplica.

Pós-condições

Não é relevante.

Requisitos não funcionais

RNF01 – A interface deve ser amigável.

Mini-Estruturas de dados

Informação – Filtros da pesquisa de Grupo	
Campo	**Atributos (Opcional)**
Órgão	Editável – Texto – Obrigatório
Opção de realizar consulta do filtro informado	Botão Pesquisar
Opção de realizar inclusão	Botão Incluir

Informação – Resultado da pesquisa de Grupo	
Campo	**Atributos (Opcional)**
Órgão	Protegido – Label – Obrigatório
Grupos do Órgão	Protegido – Label – Obrigatório
Nome do Grupo	Protegido – Label – Obrigatório
Texto de Busca no LDAP	Protegido – Label – Obrigatório
Operações Editar Grupo Apagar Grupo	 Link Editar Grupo para edição de grupo Link Apagar Grupo para exclusão de grupo

Informação – Lista de atributos de detalhes da consulta de Grupo	
Campo	**Atributos (Opcional)**
Órgão	Protegido – Label – Obrigatório
Sigla	Protegido – Label – Obrigatório
Nome	Protegido – Label – Obrigatório
Matrícula Gerente	Protegido – Label – Opcional
Grupos do Órgão	Protegido – Label – Obrigatório
Nome do Grupo	Protegido – Label – Obrigatório
Texto de Busca no LDAP	Protegido – Label – Obrigatório
Operações Editar Grupo Apagar Grupo	 Botão Editar Grupo Botão Apagar Grupo

Informação - Lista de atributos de detalhes da inclusão de Grupo	
Campo	**Atributos (Opcional)**
Nome	Editável – Texto – Obrigatório
Texto no LDAP	Editável – Texto – Obrigatório
Opção de inclusão	Botão Incluir Grupo
Opção de retorno	Botão Voltar

Informação - Lista de atributos de detalhes da edição de Grupo	
Campos	**Atributos (Opcional)**
Nome	Editável – Texto – Obrigatório
Texto no LDAP	Editável – Texto – Obrigatório
Opção de edição	Botão Editar Grupo
Opção de retorno	Botão Voltar

Cenários

Fluxo Básico
Fluxo Básico + Fluxo Alternativo A1
Fluxo Básico + Fluxo Alternativo A2
Fluxo Básico + Fluxo Alternativo A3
Fluxo Básico + Fluxo Alternativo A1 + Fluxo Alternativo A4 + Fluxo Alternativo A1
Fluxo Básico + Fluxo Alternativo A1 + Fluxo Alternativo A5 + Fluxo Alternativo A1
Fluxo Básico + Fluxo Alternativo A2 + Fluxo Alternativo A4 + Fluxo Alternativo A2
Fluxo Básico + Fluxo Alternativo A2 + Fluxo Alternativo A5 + Fluxo Alternativo A2

ANEXO IV – Caso de Uso 04 - Importar Dados de Gerentes e Substitutos

Sistema de Controle de Permissões
Caso de uso 04: Importar dados de Gerentes e Substitutos

Descrição

Este caso de uso permite que o sistema importe os dados de gerentes e substitutos de órgão.

Atores

Módulo de Administração de Configuração do Sistema
Administrador
Usuário Especial (Gerente e substituto)

Fluxo Básico: Importar dados Diariamente [CO004] [CO006]

Este caso de uso inicia-se quando o sistema detecta a necessidade de Importar dados de gerentes e substitutos.

1. O sistema valida os dados do arquivo de importação; **[A3][A4]**
2. Sistema persiste os dados e registra as alterações realizadas e o resumo da operação em um arquivo de log de importação; **[RN 2.1.3][RN 3.1][A1][M045]**
3. Caso de uso se encerra.

Fluxo Alternativo A1: Solicitar validação da lista de usuários

Este fluxo inicia-se quando o há uma alteração de gerência (gerente atual diferente do gerente anterior) de um órgão;

1. O sistema obtém a lista de usuários que constam em cada grupo do órgão no AD;
2. O sistema cria as validações necessárias e realiza a inclusão da lista de usuários que constam em cada grupo do órgão, obtida através do AD; **[RN 2.1.2][RN 3.2.2][RN 6.1.1][A2]**
3. O sistema envia um e-mail ao novo gerente do órgão informando que há uma validação da lista de usuários pendente; **[M044][RN 3.2.3]**
4. O caso de uso se encerra.

Fluxo Alternativo A2: Órgão sem gerente e sem substituto

Este caso de uso inicia-se quando o sistema detecta que não há nem gerente nem substituto em algum órgão.

1. O sistema registra o problema no arquivo de log de importação; **[M045]**
2. O sistema envia e-mail para a produção alertando o problema;**[RN 6.1.1][M046]**
3. O caso de uso retorna ao passo 2 do fluxo alternativo A1.

Fluxo Alternativo A3: Formato de Arquivo Inválido

Este caso de uso inicia-se quando o sistema detecta que o formato do arquivo de informação é inválido.

1. O sistema envia e-mail para a produção alertando o problema; **[M048]**
2. O caso de uso se encerra.

Fluxo Alternativo A4: Extensão de Arquivo Inválida

Este caso de uso inicia-se quando o sistema detecta que a extensão do arquivo de importação é inválida.

1. O sistema envia e-mail para a produção alertando o problema; **[M049]**
2. O caso de uso se encerra.

Pré-condições

Não se aplica.

Pós-condições

Registro de importação de gerentes criado em arquivo de log.
Solicitações de validação enviadas aos novos gerentes.
Validações pendentes criadas para cada órgão onde o gerente foi alterado.
Para órgãos que estavam com validação pendente e sofreram alteração de gerência a validação antiga recebe o status de "Suspensa por validação de alteração de gerência".

Requisitos não funcionais

RNF01 – O arquivo de importação dos dados de gerentes e substitutos deve ter a extensão txt.

Mini-Estruturas de dados

Não se aplica.

Cenários

Fluxo Básico
Fluxo Básico + Fluxo Alternativo A1
Fluxo Básico + Fluxo Alternativo A3
Fluxo Básico + Fluxo Alternativo A4
Fluxo Básico + Fluxo Alternativo A1 + Fluxo Alternativo A2 + Fluxo Alternativo A1

ANEXO V – Caso de Uso 06 - Solicitar Validação da Lista de Usuários Pendente

Sistema de Controle de Permissões
Caso de uso 06: Solicitar Validação da Lista de Usuários Pendente

Descrição

Este caso de uso permite que o Sistema envie lembretes de validação aos gerentes e substitutos dos órgãos.

Atores

Módulo de Administração de Configuração do Sistema

Fluxo Básico: Enviar lembrete de validação [CO004]

Este caso de uso inicia-se quando o sistema verifica que existem validações pendentes.

1. O sistema verifica se existem lembretes de validação a enviar.
2. O sistema envia um e-mail ao gerente e substituto de cada órgão com validação pendente informando que há uma validação da lista de usuários pendente. **[M004]**
3. O sistema atualiza a quantidade de lembretes já enviados de cada validação.
4. Caso de uso se encerra.

Pré-condições

Não se aplica.

Pós-condições

E-mail de lembrete de validação enviado aos gerentes e substitutos dos órgãos aos quais foi necessário o envio de lembrete.

Requisitos não funcionais

Não se aplica.

Mini-Estruturas de dados

Informação – Filtros da pesquisa de Grupo	
Campo	**Atributos (Opcional)**
Órgão	Editável – Texto – Obrigatório
Opção de realizar consulta do filtro informado	Botão Pesquisar
Opção de realizar inclusão	Botão Incluir

Cenários

Fluxo Básico

ANEXO VI – Caso de Uso 07 - Registrar Validação da Lista de Usuários

Sistema de Controle de Permissões
Caso de uso 07: Registrar Validação da Lista de Usuários

Descrição

Este caso de uso permite que um Gerente ou Substituto de órgão registre a validação da lista de permissões de usuários do órgão.

Atores

Usuário Especial (Gerente e substituto)

Fluxo Básico: Listar Validações Pendentes

Este caso de uso inicia-se quando o usuário indica a intenção de listar as validações pendentes dos órgãos que é gerente ou substituto.

8. O sistema apresenta todos os itens do **Resultado da listagem de validações pendentes [A1]** e a opção:
    - Registrar Validação **[A2]**
9. Caso de uso se encerra.

Fluxo Alternativo A1: Nenhuma Validação Encontrada

Este fluxo inicia quando o usuário não possui nenhuma validação pendente no passo 1 do Fluxo Básico;

1. O sistema apresenta uma mensagem informando que nenhum registro foi encontrado. **[M003]**
2. Caso de uso se encerra.

Fluxo Alternativo A2: Registrar Validação [CO005]

Este fluxo inicia quando o usuário solicita opção Registrar Validação no passo 1 do Fluxo Básico;

5. O sistema solicita ao usuário o preenchimento dos dados da **Lista de atributos de detalhes do registro de Validação** e apresenta as opções:
    - Adicionar Usuário **[A3]**
    - Remover Usuário **[A4]**
6. O usuário preenche os dados e solicita o registro.
7. O sistema valida e realiza o registro dos dados informados. **[RN 5.1]**
8. O sistema envia um e-mail ao grupo de produção informando os dados da validação. **[M042]**
9. O sistema exibe uma mensagem informando que o registro foi realizado com sucesso. **[M021]**
10. O caso de uso se encerra.

Fluxo Alternativo A3: Adicionar Usuário

Este fluxo inicia quando o usuário indica a intenção de adicionar um usuário a um grupo da validação no passo 1 do Fluxo Alternativo A2;

1. O sistema valida a matricula do usuário. **[RN 5.1.4][A5][A6]**
2. O sistema registra os dados do usuário e o associa ao grupo.
3. O caso de uso retorna ao passo 1 do Fluxo Alternativo A2.

Fluxo Alternativo A4: Remover Usuário

Este fluxo inicia-se quando o usuário indica a intenção de remover um usuário de um grupo da validação no passo 1 do Fluxo Alternativo A2;

Fluxo Alternativo A5: Matrícula não informada

Este fluxo inicia-se quando o sistema detecta que a matrícula do usuário não foi informada.

1. O sistema exibe mensagem de erro; **[M040]**
2. O caso de uso retorna ao passo 1 do Fluxo Alternativo A2.

Fluxo Alternativo A6: Matrícula Inexistente

Este fluxo inicia-se quando o sistema detecta que a matrícula fornecida não existe.

1. O sistema exibe mensagem de erro; **[M022]**
2. O caso de uso retorna ao passo 1 do Fluxo Alternativo A2.

Pré-condições

Não se aplica.

Pós-condições

Será possível o registro de uma validação da lista de usuários que terão acesso ao drive de rede do órgão.

Requisitos não funcionais

Não se aplica.

Mini-Estruturas de dados

Informação – Resultado da listagem de validações pendentes	
Campo	**Atributos (Opcional)**
Nome do Órgão	Protegido – Label – Obrigatório
Status	Protegido – Label – Obrigatório
Ações Registrar	 Ícone Registrar para realizar registro

Informação – Lista de atributos de detalhes do registro de Validação	
Campo	**Atributos (Opcional)**
Sigla do Órgão	Protegido – Label – Obrigatório
Nome do Órgão	Protegido – Label – Obrigatório
Gerente do Órgão	Protegido – Label – Obrigatório
Data de criação	Protegido – Label – Obrigatório
Data limite para registro	Protegido – Label – Obrigatório
Status	Protegido – Label – Obrigatório
Inclusão de Usuário	
Grupo	Protegido - Combobox
Matrícula	Editável – Texto – Opcional
Opção de adicionar usuário	Botão Adicionar
Lista de Usuários	
Usuário	Protegido – Label – Obrigatório
Matrícula	Protegido – Label – Obrigatório
Opção de remover usuário	Botão excluir
Opção de realizar Registro	Botão Registrar
Opção de retorno	Botão Voltar

Cenários

Fluxo Básico + Fluxo Alternativo A1
Fluxo Básico + Fluxo Alternativo A2
Fluxo Básico + Fluxo Alternativo A2 + Fluxo Alternativo A3 + Fluxo Alternativo A2
Fluxo Básico + Fluxo Alternativo A2 + Fluxo Alternativo A4 + Fluxo Alternativo A2
Fluxo Básico + Fluxo Alternativo A2 + Fluxo Alternativo A3 + Fluxo Alternativo A5 + Fluxo Alternativo A2
Fluxo Básico + Fluxo Alternativo A2 + Fluxo Alternativo A3 + Fluxo Alternativo A6 + Fluxo Alternativo A2

ANEXO VII – Documento de Especificação Complementar

Sistema de Controle de Permissões
Especificação complementar

1. Descrição:

A finalidade deste documento é descrever as estruturas de dados do Sistema de Controle de Drives de Rede e as mensagens apresentadas ao usuário quando ocorre algum evento ou interação.

2. Mensagens do sistema

Código	Descritivo
	Gerais
M001	Informe os campos obrigatórios.
M002	O usuário não tem permissão para acessar a funcionalidade.
M003	Nenhum registro foi encontrado para o critério de pesquisa.
	Administrar Cadastro de Órgão
M004	Órgão cadastrado com sucesso.
M005	O nome e a sigla do órgão não podem ser nulos.
M006	Já existe um órgão ativado com a sigla [sigla_do_orgao].
M007	Nome deve ser preenchido.
M008	A String deve ser preenchida.
M047	A String LDAP fornecida é inválida.
M009	O grupo foi adicionado ao órgão com sucesso.
M010	O órgão foi desativado com sucesso.
M011	O grupo foi apagado com sucesso.
M012	O grupo foi atualizado com sucesso.
M013	Confirma a exclusão do Órgão?
M014	Confirma a remoção do Grupo?
	Importação de Dados
M015	Importação realizada com sucesso. Um arquivo de log foi criado com o nome [nome_do_arquivo].
M016	Arquivo com conteúdo inválido.
M017	Nenhum arquivo selecionado.
M018	Arquivo no formato inválido.
M019	A validação do órgão falhou.
M020	Desculpe, ocorreu um erro ao realizar a requisição. Tente novamente e, se o erro persistir, contate o administrador.
	Registro de Validação.
M021	Validação registrada com sucesso.
M022	A Matrícula informada não pôde ser encontrada, verifique se a mesma está correta.
M023	O usuário foi incluído com sucesso no grupo.
M024	A validação já expirou.
M025	Usuário já esta dentro do grupo da validação.
M026	Ocorreu um erro. A transação não foi concluída.
M027	O usuário foi removido com sucesso do grupo.
M030	Confirma a remoção do Usuário?

M040	Informe o campo obrigatório: Matrícula.
	Validação de Matricula e Senha
M031	Erro na Autenticação do usuário.
M032	Identificação inválida. A Identificação é formada pelo prefixo do empregado (FC, FR ou FE) mais a matrícula sem o dígito.
	Configurações do Sistema
M033	O número mínimo de meses para a revalidação da lista de usuários deve ser de 2 meses.
M034	O número de lembretes deve ser no mínimo 2.
M035	Alterações realizadas com sucesso
M036	Informe o campo obrigatório: Caminho do arquivo de importação
M037	Informe o campo obrigatório: E-mail da Produção
M038	Informe o campo obrigatório: Número de meses para a validação periódica
M039	Informe o campo obrigatório: Quantidade de lembretes a enviar
	Email para produção alertando Validação expirada
M041	Assunto: [ZIS1] Alerta de validacao expirada Prezados, A validacao de Id <idDaValidacao> com data de expiração em <dataDeExpiracao> foi expirada.
	Email para produção alertando Registro de Validação
M042	Assunto: [ZIS1] Registro da Validacao <idDaValidacao> Prezados, Segue o resumo do registro da validacao no sistema: Identificador da Validacao: <idDaValidacao> Data de Criacao: dd/mm/yyyy Data de Expiracao: dd/mm/yyyy Data de Registro: dd/mm/yyyy Matricula e Nome do Gerente que registrou a validacao: <matricula> - <nome> --- Permissoes --- Usuarios com permissao no Grupo <nomeDoGrupo> - <matricula> | <nome> - <matricula> | <nome> - <matricula> | <nome> - <matricula> | <nome> ********** Usuarios com permissao no Grupo <nomeDoGrupo> - <matricula> | <nome> - <matricula> | <nome> - <matricula> | <nome> - <matricula> | <nome> **********
	Email para produção alertando erro na criação de Validação
M043	Assunto: [ZIS1] Erro ao Criar Validacao de Orgao

	Prezados, Ocorreu um erro ao tentar criar validacao para o orgao: <siglaDoOrgao>
	Email enviado ao gerente relativo a validação pendente.
M044	Gerente, existe uma lista de usuários pendente para validação. Acesse o link para realizar a validação.[Colocar um link redirecionando para a lista de usuários dos grupos do órgão]
	Log de Importação
M045	Orgao presente no arquivo e ignorado por nao existir: XXXX Orgao presente no arquivo e ignorado por nao existir: YYYY Nome: USUARIO GERENTE Gerente Principal: Orgao.A Gerente Substituto: Orgao.B Nome: USUARIO GERENTE 2 Gerente Principal: Gerente Substituto: Nome: USUARIO GERENTE 3 Gerente Principal: Orgao.C Gerente Substituto:
	E-mail para produção alertando Órgão sem gerentes
M046	Assunto: [ZIS1] Órgão sem gerentes Prezados, O Órgão <nome do orgão> não possui gerentes; Impossível criar validações para o mesmo.
	Email para produção alertando formato inválido
M048	Assunto: [ZIS1] Erro ao Realizar Importação Prezados, Ocorreu um erro ao tentar importar os dados do arquivo <link caminho do arquivo>. O formato do arquivo é inválido.
	Email para produção alertando extensão inválida
M049	Assunto: [ZIS1] Erro ao Realizar Importação Prezados, Ocorreu um erro ao tentar importar os dados do arquivo <link caminho do arquivo>. A extensão do arquivo é inválida.

3. Comportamentos

Identificação e Nome	Descrição
Comportamentos Gerais	▪ Todas as telas devem apresentar um Título no topo do Navegador, o mesmo título que aparece no Menu/Submenu. Por exemplo: Se no menu aparece “Órgão”, então o sistema deve apresentar o mesmo título em todas as telas desse assunto. O mesmo vale para o submenu. Se no submenu aparece “Validação”, então o sistema deve apresentar o mesmo título em todas as telas desse assunto; ▪ A mensagem “Nenhum registro foi encontrado para o critério de pesquisa” somente deve ser apresentada, após a realização de uma pesquisa que não retorna nenhum registro; ▪ No resultado das pesquisas, o sistema deve apresentar 10 linhas por página; ▪ Nos fluxos de inclusão e de alteração, a validação dos campos deve ser ao clicar no botão; ▪ Ao apresentar uma mensagem de erro do sistema ou mensagem informando que os campos obrigatórios devem ser preenchidos;
CO001	▪ Ao carregar a tela, os campos devem ser preenchidos com os dados cadastrados no sistema; ▪ Ao clicar em “Limpar”, todos os campos da tela devem ter seus valores apagados.
CO002	▪ Ao carregar a tela, todos os campos devem aparecer vazios; ▪ Ao clicar em “Limpar”, todos os campos da tela devem ter seus valores apagados;
CO003	▪ Ao carregar a tela, todos os campos devem estar preenchidos com os dados cadastrados, porém devem estar protegidos, e não deve permitir nenhuma operação, apenas a visualização;
CO004	▪ As rotinas periódicas do sistema são executadas de 6 em 6 horas.
CO005	▪ A tela deve ser carregada rapidamente. ▪ Todos os campos devem estar preenchidos com os dados cadastrados; ▪ Para cada grupo cadastrado no órgão da validação haverá uma lista de usuários na tela, identificando quais usuários possuem permissão no grupo de segurança do LDAP.
CO006	▪ O sistema só deve permitir a importação de arquivos no formato texto; ▪ Sistema deve validar se o layout do arquivo é válido. ▪ Layout do arquivo de importação: [MATRICULA], [DIGITO_VER], [NOME_COMPLETO], [NOME_EMAIL], [DOMINIO] [char] [DIRETORIA] [char], [SUPERINTEN], [DEPARTAMENTO], [DIVISAO],

	[GERENTE], [SUBSTITUTO] ▪ Ao final da importação, o sistema deve gravar em um arquivo de log uma lista com o resultado da importação.

ANEXO VIII – Documento de Regras de Negócio

Sistema de Controle de Permissões
Regras de Negócio

RN.1 Finalidade

O objetivo deste documento é reunir todas as regras de negócio relacionadas ao Sistema de Controle de Drives de Rede.

RN.2 Definições

Configurações do Sistema

RN.2.1 Validação de alteração de Configurações do Sistema.

RN.2.1.1 O número mínimo de meses para a revalidação da lista de usuários deve ser de 2 meses, podendo ser estendido.

RN.2.1.2 Após o envio da lista de usuários para o gerente e substituto, o período máximo para registro da validação deve ser 35 dias.

RN.2.1.3 O número de lembretes deverá ser no mínimo 2, podendo ser estendido. A freqüência deverá respeitar a periodicidade (diário, semanal, mensal).

RN.2.1.4 O que delimita o intervalo entre validações periódicas é a data de criação de uma validação periódica acrescida da quantidade de meses definida na configuração do sistema no momento da criação desta mesma validação.

Órgão

RN.2.2 Inclusão, alteração e exclusão de órgão

RN.2.2.1 Não poderá haver repetição de sigla do órgão.

RN.2.2.2 Ao excluir um órgão: se ele não possuir nenhuma validação seu registro é excluído do sistema. Caso contrário, o seu status é alterado para "desativado".

RN.2.2.3 A sigla do órgão será composta por <Sigla do Nome do Órgão>.<Sigla da Diretoria a qual o órgão pertence>. Caso seja apenas uma diretoria, então a sigla será a própria sigla da diretoria.

Gerentes e Substitutos

RN.2.3 Validação de Importação de Gerentes e Substitutos

RN.2.3.1 A cada nova importação serão atualizados os dados de gerentes/substitutos de órgãos que forem diferentes dos registrados no sistema. Ou seja, serão atualizados gerentes/substitutos novos ou em caso de órgãos que passaram a ficar sem gerente/substituto será realizada a remoção de gerente/substituto;

RN.2.4 Validação do envio da lista de usuários após a importação da lista de gerentes.

RN.2.4.1 Não poderá haver mais de uma validação pendente por órgão. Uma nova validação suspende todas as validações pendentes.

RN.2.4.2 Sempre que houver alteração na gerência do órgão, deve ser solicitada uma validação da lista de usuários dos grupos do órgão;

RN.2.4.3 O e-mail enviado para a gerência, solicitando a revalidação da lista, deve conter um link que redirecionara para o sistema, onde será feita a revalidação.

Validação periódica

RN.2.5 Validação periódica conforme configuração

RN.2.5.1 A periodicidade da solicitação de validação da lista de usuários do grupo de trabalho do órgão deve respeitar as configurações do sistema;

Registrar Validação

RN.2.6 Registro de Validação

RN.2.6.1 Somente o gerente / substituto pode realizar a validação dos seus órgãos;

RN.2.6.2 É permitido ao gerente / substituto adicionar qualquer usuário a lista de usuários que será validada;

RN.2.6.3 Não será possível modificar uma validação registrada no sistema;

RN.2.6.4 Na inclusão de um membro no grupo em uma validação, o sistema deverá utilizar a matrícula informada pelo usuário para encontrar o nome do colaborador no AD.

Validações

RN.2.7 Órgão sem gerente

RN.2.7.1 Se no ato de criação de validação para o órgão, seja esta por alteração de gerência ou periodicidade, o órgão não possuir nem gerente e nem substituto a produção deve ser alertada sobre este problema e a validação pendente não é criada.

Grupos

RN.2.8 Inclusão de grupos

RN.2.8.1 Na inclusão ou alteração de um grupo o sistema deverá verificar se a string LDAP fornecida é válida.

ANEXO IX – Glossário

Sistema de Controle de Permissões
Glossário

5 Introdução

5.1 Finalidade

Este documento é utilizado para registrar as terminologias específicas ao domínio do sistema. Estas terminologias são relevantes para a compreensão das descrições de casos de uso e outras documentações do projeto. O objetivo deste documento é estabelecer um ponto inicial para entendimento de padrões e definição de termos que serão necessários para evitar problemas de interpretação durante o ciclo de vida do projeto.

5.2 Referências

Não se aplica.

6 Termos do Glossário

6.1 Siglas

6.1 GIR.A

Gerência de Infra Estrutura de Rede.

6.1 AD

Active Directory. É utilizado para a autenticação de usuários no sistema e também para a consulta de matrícula e nome de usuários.

6.2 Termos do domínio

6.2 Drive de rede

Todo departamento de Furnas possui um drive de rede que armazena dados relativos a órgão.

6.2 Lista de Usuários

Refere-se a lista de usuários que deve ser validada em relação a permissão de acesso ao drive de rede do órgão.

6.2 Query

Significa Consulta. É o processo de extração de informações de um banco de dados e sua apresentação em forma adequada ao uso. Neste caso, é consulta é feita no AD.

6.2 LDAP

Lightweight Directory Access Protocol, ou LDAP, é um protocolo de aplicação aberto, livre de fornecedor e padrão de indústria para acessar e manter serviços de informação de diretório distribuído sobre uma rede de Protocolo da Internet (IP).

6.2 Substituto

Gerente substituto do órgão que pode responder pela gerência do órgão em caso de ausência do gerente principal.

6.2 Produção

Equipe responsável pela configuração dos ambientes de Furnas de acordo com as decisões tomadas no sistema.

6.2 Validação Periódica

Validação da lista de usuários que deve ter permissão de acesso aos drives de rede do órgão que é criada periodicamente de acordo com a as configurações do sistema.

6.2 Validação por Alteração Gerencial

Validação da lista de usuários que deve ter permissão de acesso aos drives de rede do órgão que é criada quando há uma alteração (mudança ou remoção) de gerente.

6.2 Órgão

Subdivisão do ambiente corporativo de Furnas.

6.2 Grupo

Agrupamento de usuários que possuem permissão de acesso a um determinado drive de rede do órgão.

ANEXO X – Features do Caso de Uso 03

Feature: Consultar Grupos

 # A interface deve ser amigável.

In order to visualizar os grupos de um determinado órgão

As an administrador do sistema

I want to consultar os grupos do órgão

Background:

Given Estou na pesquisa de grupos

And quero consultar os grupos de um órgão

Scenario: Consultar Grupos

 # Ao carregar a tela, todos os campos devem aparecer vazios;

 # Ao clicar em "Limpar", todos os campos da tela devem ter seus valores apagados;

When O sistema apresenta os filtros de pesquisa e a opção "Incluir Grupo"

 # Filtros de pesquisa de grupo

 # Órgão – Editável – Texto - Obrigatório

 # Opção de realizar consulta do filtro informado – Botão Pesquisar

 # Opção de realizar inclusão – Botão Incluir

And Eu informo a sigla do órgão

Then O sistema processa a pesquisa

And o sistema apresenta os dados e as opções de "Detalhar Grupo", "Incluir Grupo", "Editar Grupo" e "Apagar Grupo".

 # Resultado da pesquisa de Grupo

 # Órgão – Protegido – Label – Obrigatório

 # Grupos do Órgão – Protegido – Label – Obrigatório

 # Nome do Grupo – Protegido – Label – Obrigatório

 # Texto de Busca no LDAP – Protegido – Label – Obrigatório

 # Operações:

 # Editar Grupo Link Editar Grupo para edição de grupo

 # Apagar Grupo Link Apagar Grupo para exclusão de grupo

Then Eu seleciono a opção "Detalhar Grupo".

And o sistema disponibiliza o grupo no detalhamento do órgão com as opções de "Editar Grupo" e "Apagar Grupo".

 # Lista de atributos de detalhes da consulta de Grupo

 # Órgão Protegido – Label – Obrigatório

 # Sigla Protegido – Label – Obrigatório

 # Nome Protegido – Label – Obrigatório

 # Matrícula Gerente – Protegido – Label – Opcional

 # Grupos do Órgão – Protegido – Label – Obrigatório

 # Nome do Grupo – Protegido – Label – Obrigatório

 # Texto de Busca no LDAP – Protegido – Label – Obrigatório

 # Operações:

 # Editar Grupo – Botão Editar Grupo

 # Apagar Grupo – Botão Apagar Grupo

Feature: Incluir Grupo

 # A interface deve ser amigável.

In order to incluir um grupo que contém os usuários que possuem certo tipo de acesso a um determinado drive de rede de um órgão

As an administrador do sistema

I want to incluir um novo grupo de um órgão

Background:

Given Eu estou na consulta de grupo

And quero incluir um grupo.

Scenario: Incluir Grupo

Ao carregar a tela, todos os campos devem aparecer vazios;

Ao clicar em “Limpar”, todos os campos da tela devem ter seus valores apagados;

When O sistema disponibiliza a tela de inclusão

Lista de atributos de detalhes da consulta de Grupo

Nome – Editável – Texto – Obrigatório

Texto no LDAP – Editável – Texto – Obrigatório

Opção de inclusão – Botão Incluir Grupo

Opção de retorno – Botão Voltar

And Eu informo o nome do grupo e o texto para busca dos dados do grupo no sistema LDAP

Then O sistema verifica que o texto fornecido é válido para a busca no LDAP

And o sistema registra os dados do grupo

And o sistema disponibiliza o grupo no detalhamento do órgão com as opções de editar e apagar grupo.

And o sistema exibe a mensagem: “O grupo foi adicionado ao órgão com sucesso.”.

Scenario: String LDAP inválida

When Eu informo o nome do grupo e o texto para busca dos dados do grupo no sistema LDAP

Then O sistema verifica que o texto fornecido é inválido para a busca no LDAP

And o sistema exibe a mensagem: “A String LDAP fornecida é inválida.”.

Scenario: Campos obrigatórios não preenchidos

When Eu não informo um ou mais campos obrigatórios.

Then O sistema verifica que existem campos obrigatórios não preenchidos.

And o sistema exibe a mensagem: “Existem campos obrigatórios que não foram preenchidos: <nomes dos campos>.”.

Feature: Editar Grupo

 # A interface deve ser amigável.

In order to editar um grupo que contém os usuários que possuem certo tipo de acesso a um determinado drive de rede de um órgão

As an administrador do sistema

I want to editar um novo grupo de um órgão

Background:

Given Eu estou no detalhamento do órgão.

And quero editar um grupo.

Scenario: Editar Grupo

#Ao carregar a tela, os campos devem ser preenchidos com os dados cadastrados no sistema;

#Ao clicar em “Limpar”, todos os campos da tela devem ter seus valores apagados.

When Eu acesso a opção de "Editar Grupo"

Then O sistema exibe os dados do grupo para edição.

Lista de atributos de detalhes da edição de Grupo

Nome – Editável – Texto – Obrigatório

Texto no LDAP – Editável – Texto – Obrigatório

Opção de inclusão – Botão Editar Grupo

Opção de retorno – Botão Voltar

And Eu altero os dados de nome e texto LDAP.

And O sistema verifica que o texto LDAP fornecido é válido.

And O sistema registra os dados.

And o sistema exibe a mensagem: “O grupo foi atualizado com sucesso.”.

Scenario: String LDAP inválida

When Eu informo o nome do grupo e o texto para busca dos dados do grupo no sistema LDAP

Then O sistema verifica que o texto fornecido é inválido para a busca no LDAP

And o sistema exibe a mensagem: “A String LDAP fornecida é inválida.”.

Scenario: Campos obrigatórios não preenchidos

When Eu não informo um ou mais campos obrigatórios.

Then O sistema verifica que existem campos obrigatórios não preenchidos.

And o sistema exibe a mensagem: “Existem campos obrigatórios que não foram preenchidos: <nomes dos campos>.”.

Feature: Apagar Grupo

 # A interface deve ser amigável.

In order to apagar um grupo que contém os usuários que possuem certo tipo de acesso a um determinado drive de rede de um órgão

As an administrador do sistema

I want to apagar um novo grupo de um órgão

Background:

Given Eu estou no detalhamento do órgão.

And quero apagar um grupo.

Scenario: Editar Grupo

When Eu acesso a opção de "Apagar Grupo"

Then O sistema solicita a confirmação da exclusão com a mensagem: “Confirma a remoção do Grupo?”.

And Eu confirmo a exclusão.

And O sistema apaga os dados.

And o sistema exibe a mensagem: “O grupo foi apagado com sucesso.”.

Scenario: Exclusão não confirmada

When Eu não confirmo a exclusão do grupo.

Then O sistema não realiza a exclusão dos dados.

And o sistema retorna para o resultado da consulta de grupos.

ANEXO XI – Features do Caso de Uso 04

Feature: Importar Dados de Gerentes e Substitutos

In order to importar os dados de gerentes e substitutos dos órgãos

As an Módulo de Administração de Configuração do Sistema / Administrador / Usuário Especial (Gerente e substituto)

I want to localizar o arquivo de importação e atualizar os dados do sistema

Background:

Given A rotina de verificação de importação foi iniciada

And O sistema detecta que é necessário realizar a importação dos dados

Scenario: Importar Dados Diariamente

A cada nova importação serão atualizados os dados de gerentes/substitutos de órgãos que forem diferentes dos registrados no sistema. Ou seja, serão atualizados gerentes/substitutos novos ou em caso de órgãos que passaram a ficar sem gerente/substituto será realizada a remoção de gerente/substituto;

A sigla do órgão quando feita a importação, será composta por <Sigla do Nome do Órgão>.<Sigla da Diretoria a qual o órgão pertence>. Caso seja apenas uma diretoria, então a sigla será a própria sigla da diretoria.

 # As rotinas periódicas do sistema são executadas de 6 em 6 horas.

When O sistema detecta a necessidade de Importar dados de gerentes e substitutos

 # O sistema só deve permitir a importação de arquivos no formato texto;

 # Sistema deve validar se o layout do arquivo é válido.

 # Layout do arquivo de importação:

 # [MATRICULA],

 # [DIGITO_VER],

 # [NOME_COMPLETO],

 # [NOME_EMAIL],

 # [DOMINIO] [char]

 # [DIRETORIA] [char],

 # [SUPERINTEN],

 # [DEPARTAMENTO],

 # [DIVISAO],

 # [GERENTE],

 # [SUBSTITUTO]

 # Ao final da importação, o sistema deve gravar em um arquivo de log uma lista com o resultado da importação.

Then O sistema verifica que os dados do arquivo de importação são válidos.

And O sistema persiste os dados e registra as alterações realizadas e o resumo da operação em um arquivo de log de importação.

 # Padrão do log de importação:

 # Orgao presente no arquivo e ignorado por nao existir: XXXX

 # Orgao presente no arquivo e ignorado por nao existir: YYYY

#

 # Nome: USUARIO GERENTE

 # Gerente Principal: Orgao.A

 # Gerente Substituto: Orgao.B

#

 # Nome: USUARIO GERENTE 2

 # Gerente Principal:

 # Gerente Substituto:

#

 # Nome: USUARIO GERENTE 3

 # Gerente Principal: Orgao.C

 # Gerente Substituto:

Scenario: Solicitar Validação da Lista de Usuários

Ao excluir um órgão: se ele não possuir nenhuma validação seu registro é excluído do sistema. Caso contrário, o seu status é alterado para “desativado”.

Sempre que houver alteração na gerência do órgão, deve ser solicitada uma validação da lista de usuários dos grupos do órgão;

O e-mail enviado para a gerência, solicitando a revalidação da lista, deve conter um link que redirecionara para o sistema, onde será feita a revalidação.

When O sistema detecta que há uma alteração de gerência (gerente atual diferente do gerente anterior) de um órgão

And O sistema obtém a lista de usuários que constam em cada grupo do órgão no AD;

Then O sistema cria as validações necessárias e realiza a inclusão da lista de usuários que constam em cada grupo do órgão, obtida através do AD;

And O sistema envia um e-mail ao novo gerente do órgão informando que há uma validação da lista de usuários pendente;

Scenario: Órgão sem gerente e sem substituto

When O sistema detecta que há uma alteração de gerência (gerente atual diferente do gerente anterior) de um órgão

And O sistema obtém a lista de usuários que constam em cada grupo do órgão no AD;

Then O sistema cria as validações necessárias e realiza a inclusão da lista de usuários que constam em cada grupo do órgão, obtida através do AD;

But O sistema detecta que não há nem gerente nem substituto em algum órgão

Then O sistema envia e-mail para a produção alertando o problema;

 # E-mail para produção alertando Órgão sem gerentes

 # Assunto: [ZIS1] Órgão sem gerentes

#

 # Prezados,

 # O Órgão <nome_do_orgão> não possui gerentes; Impossível criar validações para o mesmo.

Scenario: Formato de arquivo inválido

When O sistema detecta a necessidade de Importar dados de gerentes e substitutos

 # O sistema só deve permitir a importação de arquivos no formato texto;

 # Sistema deve validar se o layout do arquivo é válido.

 # Layout do arquivo de importação:

 # [MATRICULA],

 # [DIGITO_VER],

 # [NOME_COMPLETO],

 # [NOME_EMAIL],

 # [DOMINIO] [char]

 # [DIRETORIA] [char],

 # [SUPERINTEN],

 # [DEPARTAMENTO],

 # [DIVISAO],

 # [GERENTE],

 # [SUBSTITUTO]

 # Ao final da importação, o sistema deve gravar em um arquivo de log uma lista com o resultado da importação.

But O sistema verifica que os dados do arquivo de importação são inválidos.

Then O sistema envia e-mail para a produção alertando o problema.

 # Email para produção alertando erro na importação de arquivo

 # Assunto: [ZIS1] Erro ao Realizar Importação

#

 # Prezados,

 # Ocorreu um erro ao tentar importar os dados do arquivo <link caminho do arquivo>.

 # O formato do arquivo é inválido.

Scenario: Extensão de arquivo inválida

When O sistema detecta a necessidade de Importar dados de gerentes e substitutos

 # O sistema só deve permitir a importação de arquivos no formato texto;

 # Sistema deve validar se o layout do arquivo é válido.

 # Layout do arquivo de importação:

 # [MATRICULA],

 # [DIGITO_VER],

 # [NOME_COMPLETO],

 # [NOME_EMAIL],

 # [DOMINIO] [char]

 # [DIRETORIA] [char],

 # [SUPERINTEN],

 # [DEPARTAMENTO],

 # [DIVISAO],

 # [GERENTE],

 # [SUBSTITUTO]

 # Ao final da importação, o sistema deve gravar em um arquivo de log uma lista com o resultado da importação.

But O sistema verifica que a extensão do arquivo de importação é inválida.

Then O sistema envia e-mail para a produção alertando o problema.

 # Email para produção alertando erro na importação de arquivo

 # Assunto: [ZIS1] Erro ao Realizar Importação

#

 # Prezados,

 # Ocorreu um erro ao tentar importar os dados do arquivo <link caminho do arquivo>.

 # A extensão do arquivo é inválida.

ANEXO XII – Features do Caso de Uso 06

Feature: Solicitar Validação da Lista de Usuários Pendente

In order to enviar lembretes de validação aos gerentes e substitutos dos órgãos.

As an Módulo de Administração de Configuração do Sistema

I want to Enviar lembretes

Background:

Given A rotina de verificação de validações pendentes foi iniciada

Scenario: Enviar lembrete de validação

 # As rotinas periódicas do sistema são executadas de 6 em 6 horas.

When o sistema verifica que existem validações pendentes;

 # Filtros de pesquisa de grupo

 # Órgão – Editável – Texto - Obrigatório

 # Opção de realizar consulta do filtro informado – Botão Pesquisar

 # Opção de realizar inclusão – Botão Incluir

And O sistema verifica se existem lembretes de validação a enviar.

Then O sistema envia um e-mail ao gerente e substituto de cada órgão com validação pendente informando que há uma validação da lista de usuários pendente.

 # Órgão cadastrado com sucesso.

And O sistema atualiza a quantidade de lembretes já enviados de cada validação.

ANEXO XIII – Features do Caso de Uso 07

Feature: Listar Validações Pendentes

In order to verificar quais são as validações pendentes para o meu usuário

As an Usuário Especial (Gerente e substituto)

I want to Listar as validações pendentes

Scenario: Listar Validações Pendentes

When Eu indico a intenção de listar as validações pendentes

And O sistema apresenta todos os itens do Resultado da listagem de validações pendentes e a opção “Registrar Validação”

 # Resultado da listagem de validações pendentes

 # Nome do Órgão – Protegido – Label – Obrigatório

 # Status – Protegido – Label – Obrigatório

 # Registrar – Ícone Registrar para realizar registro

Scenario: Nenhuma Validação Encontrada

When Eu indico a intenção de listar as validações pendentes

But Não há nenhuma validação pendente para o meu usuário

Then O sistema apresenta a mensagem: “Nenhum registro foi encontrado para o critério de pesquisa.”.

Feature: Registrar Validações Pendentes

Somente o gerente / substituto pode realizar a validação dos seus órgãos;

Não será possível modificar uma validação registrada no sistema;

In order to registrar a validação da lista de permissões de usuários do órgão.

As an Usuário Especial (Gerente e substituto)

I want to Registrar uma validação

Scenario: Registrar Validação

 # A tela deve ser carregada rapidamente.

 # Ao carregar a tela, todos os campos devem estar preenchidos com os dados cadastrados;

 # Para cada grupo cadastrado no órgão da validação haverá uma lista de usuários na tela, identificando quais usuários possuem permissão no grupo de segurança do LDAP.

When Eu indico a intenção de registrar uma validação

And O sistema solicita ao usuário o preenchimento dos dados da Lista de atributos de detalhes do registro de validação e apresenta as opções: “Adicionar Usuário” e “Remover Usuário”

 # Lista de atributos de detalhes do registro de validação

 # Sigla do Órgão – Protegido – Label – Obrigatório

 # Nome do Órgão – Protegido – Label – Obrigatório

 # Gerente do Órgão – Protegido – Label – Obrigatório

 # Data de criação – Protegido – Label – Obrigatório

 # Data limite para registro – Protegido – Label – Obrigatório

 # Status – Protegido – Label – Obrigatório

 # Lista de Usuários:

 # Usuário – Protegido – Label – Obrigatório

 # Matrícula – Protegido – Label – Obrigatório

 # Opção de remover usuário – Botão excluir

 # Opção de realizar Registro – Botão Registrar

 # Opção de retorno – Botão Voltar

And Eu preencho os dados e solicito o registro

Then O sistema valida e realiza o registro dos dados informados.

And O sistema envia um e-mail ao grupo de produção informando os dados da validação.

 # Email para produção alertando Registro de Validação

 # Assunto: [ZIS1] Registro da Validacao <idDaValidacao>

#

 # Prezados,

 # Segue o resumo do registro da validacao no sistema:

#

 # Identificador da Validacao: <idDaValidacao>

 # Data de Criacao: dd/mm/yyyy

 # Data de Expiracao: dd/mm/yyyy

 # Data de Registro: dd/mm/yyyy

 # Matricula e Nome do Gerente que registrou a validacao: <matricula> - <nome>

#

 # ---

 # Permissoes

 # --

#

 # Usuarios com permissao no Grupo <nomeDoGrupo>

 # - <matricula> | <nome>

 # - <matricula> | <nome>

 # - <matricula> | <nome>

 # - <matricula> | <nome>

#

*# *****************

#

 # Usuarios com permissao no Grupo <nomeDoGrupo>

 # - <matricula> | <nome>

 # - <matricula> | <nome>

 # - <matricula> | <nome>

 # - <matricula> | <nome>

#

*# *****************

And O sistema exibe a mensagem “Validação registrada com sucesso.”.

Feature: Adicionar Usuário

É permitido ao gerente / substituto adicionar qualquer usuário a lista de usuários que será validada;

Na inclusão de um membro no grupo em uma validação, o sistema deverá utilizar a matrícula informada pelo usuário para encontrar o nome do colaborador no AD.

In order to validar a lista de usuários de um determinado grupo

As an Usuário Especial (Gerente e substituto)

I want to Adicionar um usuário ao grupo

Background:

Given Eu estou no registro de validação

And Quero adicionar usuários a um grupo de permissões

Scenario: Adicionar Usuário

 # Inclusão de Usuário:

 # Grupo – Protegido - Combobox

 # Matrícula – Editável – Texto – Opcional

 # Opção de adicionar usuário – Botão Adicionar

When Eu indico que quero adicionar um usuário a um grupo

And Eu informo a matrícula de um usuário

And O sistema verifica que a matrícula do usuário é válida

Then O sistema encontra o nome do colaborador no AD

And O sistema registra os dados do usuário e o associa ao grupo.

Scenario: Matrícula Não Informada

When Eu indico que quero adicionar um usuário a um grupo

But Eu não informo a matrícula de um usuário

Then O sistema exibe a mensagem “Informe o campo obrigatório: Matrícula.”.

Scenario: Matrícula Inexistente

When Eu indico que quero adicionar um usuário a um grupo

And Eu informo a matrícula de um usuário

But O sistema verifica que a matrícula informada não existe

Then O sistema exibe a mensagem “A Matrícula informada não pôde ser encontrada, verifique se a mesma está correta.”.

Feature: Remover Usuário

In order to validar a lista de usuários de um determinado grupo

As an Usuário Especial (Gerente e substituto)

I want to Remover um usuário do grupo

Background:

Given Eu estou no registro de validação

And Quero remover usuários de um grupo de permissões

Scenario: Remover Usuário

When Eu indico que quero remover um usuário de um grupo

ANEXO XIX – Termo de Consentimento

TERMO DE CONSENTIMENTO

Prezado,

Convido você para participar de um estudo que é parte do Trabalho de Conclusão de Curso do Bacharelado em Sistemas de Informação da UNIRIO, com foco em processos de software. Seu objetivo é avaliar a forma de descrição de requisitos por meio de Casos de Uso e de Features.

O estudo ocorrerá da seguinte maneira: em um primeiro momento você fornecerá informações sobre o seu perfil profissional. Os dados de identificação não serão mencionados no relatório da pesquisa, o que preservará o anonimato e sigilo dos respondentes. Na segunda etapa do estudo você receberá um material de detalhamento de requisitos e instruções sobre a composição do material. Será solicitado que você preencha um questionário ao final com suas impressões sobre o material de detalhamento dos requisitos.

A sua participação é voluntária. Você pode desistir de participar a qualquer momento, sem sofrer penalidades.

Para garantir sua privacidade, a sua identidade não será revelada. Os resultados do estudo serão divulgados exclusivamente pelo pesquisador e por seus orientadores na literatura especializada ou em congressos e eventos científicos.

Qualquer dúvida a respeito dessa pesquisa entre em contato pelos e-mails:

Rafaela Sampaio – rafaela.sampaio@uniriotec.br
Gleison Santos - gleison.santos@uniriotec.br

UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO

DECLARAÇÃO DE CONSENTIMENTO

Li as informações contidas neste documento antes de assinar esta Declaração de Consentimento. Declaro que toda a linguagem utilizada na descrição do estudo foi explicada e que recebi respostas para todas as minhas dúvidas. Confirmo que recebi uma cópia deste Termo de Consentimento. Compreendo que posso me retirar do estudo a qualquer momento, sem sofrer qualquer penalidade.

Dou meu consentimento de livre e espontânea vontade para participar deste estudo.

_____ , _____/_____/_____
Local e Data

_____
Assinatura do Participante

_____
Assinatura do Pesquisador