Guia de Especificação de Requisitos

Ministério de Minas e Energias

Versão 1.0

Histórico de Revisão

|  |  |  |  |
| --- | --- | --- | --- |
| Data | Versão | Descrição | Autor |
| 26/07/2013 | 1.0 | Criação do Artefato | Anderson Junqueira |
| 09/09/2013 | 1.1 | Ajuste da figura 1 para contemplar a rastreabilidade dos requisitos não funcionais e o roteiro de testes. | Fernanda Braga |

Sumário

[1. Introdução 7](#_Toc363029302)

[2. Requisitos 7](#_Toc363029303)

[2.1. Categorias dos Requisitos 7](#_Toc363029304)

[2.2. Tipos de Requisitos 7](#_Toc363029305)

[2.2.1. Necessidades (NEC) 7](#_Toc363029306)

[2.2.2. Característica (CAR) 8](#_Toc363029307)

[2.2.3. Requisitos Funcionais (REQ) 8](#_Toc363029308)

[2.2.4. Requisitos Não Funcionais (RNF) 8](#_Toc363029309)

[2.2.5. Caso de Uso (UC) 8](#_Toc363029310)

[2.2.6. Regra de Negócio (RN) 9](#_Toc363029311)

[2.2.7. Mensagens (MSG) 9](#_Toc363029312)

[2.2.8. Roteiro de Teste (RT) 9](#_Toc363029313)

[2.3. Rastreabilidade de Requisitos 9](#_Toc363029314)

[2.4. Critérios de Aceitação 10](#_Toc363029315)

[3. Ferramentas, Ambiente e Infraestrutura 11](#_Toc363029316)

[4. Fornecedores de Requisitos 11](#_Toc363029317)

[5. Técnicas de Levantamento de Requisitos 11](#_Toc363029318)

[5.1. Entrevistas 11](#_Toc363029319)

[5.2. Prototipagem 12](#_Toc363029320)

[5.3. Questionários 12](#_Toc363029321)

[5.4. Workshops 13](#_Toc363029322)

[5.5. Brainstorming 13](#_Toc363029323)

[5.6. JAD 14](#_Toc363029324)

[6. Boas Práticas e Padrões 15](#_Toc363029325)

[6.1. Requisito 15](#_Toc363029326)

[6.2. Requisitos Funcionais 15](#_Toc363029327)

[6.3. Requisitos Não Funcionais 16](#_Toc363029328)

[6.3.1. Performance 16](#_Toc363029329)

[6.3.2. Integração 16](#_Toc363029330)

[6.3.3. Segurança 16](#_Toc363029331)

[6.4. Defina os atores 16](#_Toc363029332)

[6.5. Defina os Casos de Uso Principais 16](#_Toc363029333)

[6.6. Identifique Oportunidades de Reuso de Casos de Uso 16](#_Toc363029334)

[6.7. Casos de Uso 18](#_Toc363029335)

[6.7.1. Por que Criar Casos de Uso 18](#_Toc363029336)

[6.7.2. O que NÃO DEVE ser escrito em um caso de uso 18](#_Toc363029337)

[6.7.3. Regras de Negócio 18](#_Toc363029338)

[6.8. Nomenclatura dos Casos de Uso 19](#_Toc363029339)

[6.8.1. Estrutura de um caso de uso: 19](#_Toc363029340)

[6.8.2. Escrevendo um Caso de Uso 19](#_Toc363029341)

[6.8.3. Considerações Adicionais 20](#_Toc363029342)

[6.9. Pré-condições 21](#_Toc363029343)

[6.10. Pós-Condições 21](#_Toc363029344)

[6.11. Seja produtivo sem buscar a excelência 21](#_Toc363029345)

[6.12. E as Histórias de Usuário? 21](#_Toc363029346)

[6.13. Desenvolvimento Ágil com Casos de Uso 21](#_Toc363029347)

[7. Referências 22](#_Toc363029348)

1. Introdução

O objetivo deste documento é definir requisitos, apresentar padrões, conceitos e boas práticas que devem ser seguidos pela equipe no tocante à disciplina de requisitos.

A intenção deste guia não é apresentar soluções para todas as situações que serão vivenciadas durante a especificação de requisitos, o que se pretende é garantir uma maior padronização nas especificações que são elaboradas por diferentes analistas.

1. Requisitos
2. O ato de requerer; demanda. requisição.
3. Aquilo que é requerido; um comando imperativo ou autoritário; uma condição essencial; alguma coisa necessária; uma necessidade.

Webster's Revised Unabridged Dictionary, © 1996, 1998 MICRA, Inc. (tradução livre).

São objetivos ou restrições estabelecidas por clientes e usuários que definem as suas diversas propriedades do sistema.

Um conjunto de requisitos pode ser definido como uma condição ou capacidade necessária que o software deve possuir para que o usuário possa resolver um problema ou atingir um objetivo ou para atender as necessidades ou restrições da organização ou dos outros componentes do sistema.

* 1. Categorias dos Requisitos

O CMMI-DEV versão 1.3 subdivide claramente os requisitos em duas categorias:

* Requisitos do Cliente;
* Requisitos do Produto.

O primeiro diz respeito aos requisitos definidos pelo usuário ou gestor do sistema explicitamente. O RUP os determina como “Necessidades” e “Características” do produto. O segundo refere-se aos derivados dos primeiros pela equipe de analistas de requisitos que serão seguidos pela equipe de desenvolvimento. São os requisitos “Funcionais” e “Não Funcionais”.

* 1. Tipos de Requisitos

A seguir são definidos os tipos e quais atributos devem ser mapeados para cada um deles:

* + 1. Necessidades (NEC)

São as necessidades de alto nível do cliente em relação ao sistema.

Atributos:

* **ID –** Número sequencial da necessidade no projeto, ex: NEC001;
* **Nome** – Indica a descrição da necessidade;
* **Nível de Prioridade** – Indica o grau de benefício dos requisitos em relação às expectativas dos fornecedores de requisitos. Pode ser definida como:

|  |  |
| --- | --- |
| Crítico | Requisitos essenciais ou o fracasso em sua implementação significa que o sistema não irá atender as necessidades do cliente. Imprescindível que seja atendido pelo sistema, condição fundamental para o sucesso do projeto. |
| Importante | Requisitos importantes para a eficácia ou eficiência do sistema. Sua não implementação afeta a satisfação do usuário e/ou o valor agregado do produto. Afeta a satisfação do usuário significativamente, mas o não atendimento não determina o fracasso do projeto. |
| Útil | Requisitos úteis, porém menos críticos, sendo usados menos freqüentemente. Não possui muito significado para a satisfação do usuário e pode deixar de ser atendida. |

* + 1. Característica (CAR)

São as características pretendidas para o sistema.

Atributos:

* **ID –** Número sequencial da característica no projeto, ex: CAR001;
* **Descrição** – Texto explicativo da característica.
  + 1. Requisitos Funcionais (REQ)

Descreve uma função do sistema ou componente. Os requisitos funcionais podem ser cálculos, detalhes técnicos, manipulação de dados e de processamento e outras funcionalidades específicas que definem o que um sistema, idealmente, será capaz de realizar.

Atributos:

* **ID –** Número sequencial do requisito no projeto, ex: REQ001;
* **Nome**;
* **Resumo** – Descrição do requisito;
* **Justificativa** – O motivo da existência do requisito;
* **Severidade** – Importância do requisito:

|  |  |
| --- | --- |
| Alta | Devem ser especificados na primeira versão do sistema. |
| Média | Podem ser negociados, com a respectiva área, para serem criados posteriormente. |
| Baixa | Podem ser especificados em uma próxima versão do sistema. |

* + 1. Requisitos Não Funcionais (RNF)

São relacionados ao uso da aplicação em termos de desempenho, usabilidade, confiabilidade, segurança, disponibilidade, manutenibilidade e tecnologias envolvidas.

Atributos:

* **ID –** Número sequencial do requisito no projeto, ex: RNF001;
* **Descrição** – Texto explicativo do requisito, se possível, incluir exemplos.
  + 1. Caso de Uso (UC)

São narrativas em texto, descrevendo a unidade funcional, e são amplamente utilizados para descobrir e registrar requisitos de sistemas.

Atributos:

* **ID –** Número sequencial do caso de uso no projeto, ex: UC001;
* **Nome.**
  + 1. Regra de Negócio (RN)

Visam detalhar as funcionalidades particulares do software, o desenvolvimento de métodos de tratamento de exceções, particularidades que o sistema possa executar e o mais importante, limitar ações fora do processo normal de funcionamento de um sistema específico.

Atributos:

* **ID –** Número sequencial da regra no projeto, ex: RN001;
* **Descrição** – Texto explicativo da regra, se possível, incluir exemplos.
  + 1. Mensagens (MSG)

Centralizam as mensagens exibidas no sistema.

Atributos:

* **ID –** Número sequencial da mensagem no projeto, ex: MSG001;
* **Texto** – Texto explicativo da regra, se possível, incluir exemplos.
  + 1. Roteiro de Teste (RT)

Especifica a saída esperada e os resultados esperados do processamento. Numa situação ideal, no desenvolvimento de casos de teste, se espera encontrar o subconjunto dos casos de teste possíveis com a maior probabilidade de encontrar a maioria dos erros.

Atributos:

* **ID –** Número sequencial do roteiro no projeto, ex: RT001;
* **Nome.**
  1. Rastreabilidade de Requisitos

A rastreabilidade dos requisitos é uma técnica usada para avaliação do impacto de uma alteração nos requisitos do sistema.

Ela mapeia as relações entre os requisitos de derivação (rastreabilidade vertical) ou dependência (rastreabilidade horizontal) entre si e com isso permitir a avaliação dos requisitos que sofrerão impacto com uma mudança em outro requisito. Essa avaliação subsidia o cálculo do prazo e custo da alteração.

A seguir são relacionados todos os tipos de requisitos, as relações entre si e os artefatos onde são documentados:

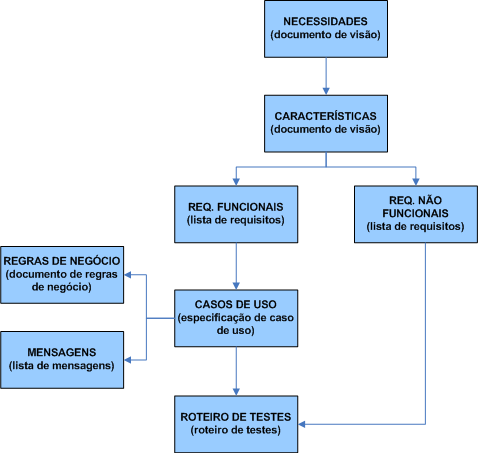


Figura - Requisitos, suas relações e artefatos

* 1. Critérios de Aceitação

Todos os requisitos de projeto devem ser analisados interna (equipe do projeto) e externamente (cliente) para que se possam identificar necessidades de um maior esclarecimento. Para isto, os seguintes critérios para a aprovação dos requisitos com a equipe de projeto e a validação com o cliente devem ser usados:

* Correto
* Não-ambíguo
* Completo
* Consistente
* Priorizado
* Verificável
* Modificável
* Rastreável

1. Ferramentas, Ambiente e Infraestrutura

A tabela abaixo apresenta uma relação de ferramentas identificadas como necessárias para o gerenciamento de requisitos do projeto:

|  |  |
| --- | --- |
| **Ferramenta** | **Finalidade** |
| Microsoft Word | Editor de textos |
| SVN | Ferramenta de gestão de configuração |

As demais ferramentas, ambientes e infra-estrutura estão descritos no documento de Plano de Projeto.

1. Fornecedores de Requisitos

Devem ser definidos no Documento de Visão.

1. Técnicas de Levantamento de Requisitos

As técnicas de levantamento de requisitos têm por objetivo superar as dificuldades relativas a esta fase. Todas as técnicas possuem um conceito próprio e suas respectivas vantagens e desvantagens, que podem ser utilizadas em conjunto pelo analista.

* 1. Entrevistas

A entrevista é uma das técnicas tradicionais mais simples de utilizar e que produz bons resultados na fase inicial de obtenção de dados. Convém que o entrevistador dê margem ao entrevistado para expor as suas idéias. É necessário ter um plano de entrevista para que não haja dispersão do assunto principal e a entrevista fique longa, deixando o entrevistado cansado e não produzindo bons resultados.

As seguintes diretrizes podem ser de grande auxilio na direção de entrevistas bem sucedidas com o usuário: desenvolver um plano geral de entrevistas, certificar-se da autorização para falar com os usuários, planejar a entrevista para fazer uso eficiente do tempo, utilizar ferramentas automatizadas que sejam adequadas, tentar descobrir que informação o usuário está mais interessado e usar um estilo adequado ao entrevistar.

Para planejar a entrevista é necessário que antes dela sejam coletados e estudados todos os dados pertinentes à discussão, como formulários, relatórios, documentos e outros. Dessa forma, o analista estará bem contextualizado e terá mais produtividade nos assuntos a serem discutidos na entrevista.

É importante determinar um escopo relativamente limitado, focalizando uma pequena parte do sistema para que a reunião não se estenda por mais de uma hora. O usuário tem dificuldade de concentração em reuniões muito longas, por isso é importante focalizar a reunião no escopo definido.

Após a entrevista é necessário validar se o que foi documentado pelo analista está de acordo com a necessidade do usuário, que o usuário não mudou de opinião e que o usuário entende a notação ou representação gráfica de suas informações.

A atitude do analista em relação à entrevista é determinar seu fracasso ou sucesso. Uma entrevista não é uma competição, deve-se evitar o uso excessivo de termos técnicos e não conduzir a entrevista em uma tentativa de persuasão. O modo como o analista fala não deve ser muito alto, nem muito baixo, tampouco indiretamente, ou seja, utilizar os termos: ele disse isso ou aquilo na reunião para o outro entrevistado. O modo melhor para agir seria, por exemplo, dizer: O João vê a solução para o projeto dessa forma. E o senhor André, qual é a sua opinião? Em uma entrevista o analista nunca deve criticar a credibilidade do entrevistado. O analista deve ter em mente que o entrevistado é o perito no assunto e fornecerá as informações necessárias ao sistema.

Para elaborar perguntas detalhadas é necessário solicitar que o usuário:

* Explique o relacionamento entre o que está em discussão e as demais partes do sistema;
* Descreva o ponto de vista de outros usuários em relação ao item que esteja sendo discutido;
* Descreva informalmente a narrativa do item em que o analista deseja obter informações;
* Perguntar ao usuário se o item em discussão depende para a sua existência de alguma outra coisa, para assim poder juntar os requisitos comuns do sistema, formando assim um escopo conciso.

Pode-se utilizar a confirmação, para tanto o analista deve dizer ao usuário o que acha que ouviu ele dizer. Neste caso, o analista deve utilizar as suas próprias palavras em lugar das do entrevistado e solicitar ao entrevistado confirmação do que foi dito.

* 1. Prototipagem

Protótipo tem por objetivo explorar aspectos críticos dos requisitos de um produto, implementando de forma rápida um pequeno subconjunto de funcionalidades deste produto. O protótipo é indicado para estudar as alternativas de interface do usuário; problemas de comunicação com outros produtos; e a viabilidade de atendimento dos requisitos de desempenho. As técnicas utilizadas na elaboração do protótipo são várias: interface de usuário, relatórios textuais, relatórios gráficos, entre outras.

Alguns dos benefícios do protótipo são as reduções dos riscos na construção do sistema, pois o usuário chave já verificou o que o analista captou nos requisitos do produto. Para ter sucesso na elaboração dos protótipos é necessária a escolha do ambiente de prototipagem, o entendimento dos objetivos do protótipo por parte de todos os interessados no projeto, a focalização em áreas menos compreendidas e a rapidez na construção.

* 1. Questionários

O uso de questionário é indicado, por exemplo, quando há diversos grupos de usuários que podem estar em diversos locais diferentes do país. Neste caso, elaboram-se pesquisas específicas de acompanhamento com usuários selecionados, que a contribuição em potencial pareça mais importante, pois não seria prático entrevistar todas as pessoas em todos os locais.

Existem vários tipos de questionários que podem ser utilizados. Entre estes podemos listar: múltipla escolha, lista de verificação e questões com espaços em branco. O questionário deve ser desenvolvido de forma a minimizar o tempo gasto em sua resposta.

Na fase de preparação do questionário deve ser indicado o tipo de informação que se deseja obter. Assim que os requisitos forem definidos o analista deve elaborar o questionário com questões de forma simples, clara e concisa, deixar espaço suficiente para as repostas que forem descritivas e agrupar as questões de tópicos específicos em um conjunto com um título especial. O questionário deve ser acompanhado por uma carta explicativa, redigida por um alto executivo, para enfatizar a importância dessa pesquisa para a organização.

Deve ser desenvolvido um controle que identifique todas as pessoas que receberão os questionários. A distribuição deve ocorrer junto com instruções detalhadas sobre como preenchê-lo e ser indicado claramente o prazo para devolução do questionário. Ao analisar as respostas dos participantes é feito uma consolidação das informações fornecidas no questionário, documentando as principais descobertas e enviando uma cópia com estas informações para o participante como forma de consideração pelo tempo dedicado a pesquisa.

* 1. Workshops

Trata-se de uma técnica de elicitação em grupo usada em uma reunião estruturada. Devem fazer parte do grupo uma equipe de analistas e uma seleção dos stakeholders que melhor representam a organização e o contexto em que o sistema será usado, obtendo assim um conjunto de requisitos bem definidos.

Ao contrário das reuniões, onde existe pouca interação entre todos os elementos presentes, o workshop tem o objetivo de acionar o trabalho em equipe. Há um facilitador neutro cujo papel é conduzir a workshop e promover a discussão entre os vários mediadores. As tomadas de decisão são baseadas em processos bem definidos e com o objetivo de obter um processo de negociação, mediado pelo facilitador.

Uma técnica utilizada em workshops é o brainstorming. Após os workshops serão produzidas documentações que refletem os requisitos e decisões tomadas sobre o sistema a ser desenvolvido.

Alguns aspectos importantes a serem considerados: a postura do condutor do seminário deve ser de mediador e observador; a convocação deve possuir dia, hora, local, horário de início e de término; assunto a ser discutido e a documentação do seminário.

* 1. Brainstorming

Brainstorming é uma técnica para geração de idéias. Ela consiste em uma ou várias reuniões que permitem que as pessoas sugiram e explorem idéias.

As principais etapas necessárias para conduzir uma sessão de brainstorming são:

* Seleção dos participantes: Os participantes devem ser selecionados em função das contribuições diretas que possam dar durante a sessão. A presença de pessoas bem informadas, vindas de diferentes grupos garantirá uma boa representação;
* Explicar a técnica e as regras a serem seguidas: O líder da sessão explica os conceitos básicos de brainstorming e as regras a serem seguidas durante a sessão;
* Produzir uma boa quantidade de idéias: Os participantes geram tantas idéias quantas forem exigidas pelos tópicos que estão sendo o objeto do brainstorming. Os participantes são convidados, um por vez, a dar uma única idéia. Se alguém tiver problema, passa a vez e espera a próxima rodada.

No brainstorming as idéias que a princípio pareçam não convencionais, são encorajadas, pois elas frequentemente estimulam os participantes, o que pode levar a soluções criativas para o problema. O número de idéias geradas deve ser bem grande, pois quanto mais idéias forem propostas, maior será a chance de aparecerem boas idéias. Os participantes também devem ser encorajados a combinar ou enriquecer as idéias de outros e, para isso, é necessário que todas as idéias permaneçam visíveis a todos os participantes.

Nesta técnica é designada uma pessoa para registrar todas as idéias em uma lousa branca ou em papel. À medida que cada folha de papel é preenchida, ela é colocada de forma que todos os participantes possam vê-la.

Analisar as idéias é a fase final do brainstorming. Nessa fase é realizada uma revisão das idéias, uma de cada vez. As consideradas valiosas pelo grupo são mantidas e classificadas em ordem de prioridade.

* 1. JAD

JAD (Joint Application Design) é uma técnica para promover cooperação, entendimento e trabalho em grupo entre os usuários desenvolvedores.

O JAD facilita a criação de uma visão compartilhada do que o produto de software deve ser. Através da sua utilização os desenvolvedores ajudam os usuários a formular problemas e explorar soluções. Dessa forma, os usuários ganham um sentimento de envolvimento, posse e responsabilidade com o sucesso do produto.

A técnica JAD tem quatro princípios básicos:

1. Dinâmica de grupo: são realizadas reuniões com um líder experiente, analista, usuários e gerentes, para despertar a força e criatividade dos participantes. O resultado final será a determinação dos objetivos e requisitos do sistema;
2. Uso de técnicas visuais: para aumentar a comunicação e o entendimento;
3. Manutenção do processo organizado e racional: o JAD emprega a análise top down e atividades bem definidas. Possibilita assim, a garantia de uma análise completa reduzindo as chances de falhas ou lacunas no projeto e cada nível de detalhe recebe a devida atenção;
4. Utilização de documentação padrão: preenchida e assinada por todos os participantes. Este documento garante a qualidade esperada do projeto e promove a confiança dos participantes.

A técnica JAD é composta de duas etapas principais: planejamento, que tem por objetivo elicitar e especificar os requisitos; e projeto, em que se lida com o projeto de software.

Cada etapa consiste em três fases: adaptação, sessão e finalização. A fase de adaptação consiste na preparação para a sessão, ou seja, organizar a equipe, adaptar o processo JAD ao produto a ser construído e preparar o material. Na fase de sessão é realizado um ou mais encontros estruturados, envolvendo desenvolvedores e usuários onde os requisitos são desenvolvidos e documentados. A fase de finalização tem por objetivo converter a informação da fase de sessão em sua forma final (um documento de especificação de requisitos).

Há seis tipos de participantes, embora nem todos participem de todas as fases:

* Líder da sessão: é responsável pelo sucesso do esforço, sendo o facilitador dos encontros. Deve ser competente, com bom relacionamento pessoal e qualidades gerenciais de liderança;
* Engenheiro de requisitos: é o participante diretamente responsável pela produção dos documentos de saída das sessões JAD. Deve ser um desenvolvedor experiente para entender as questões técnicas e detalhes que são discutidos durante as sessões e ter habilidade de organizar idéias e expressá-las com clareza;
* Executor: é o responsável pelo produto sendo construído. Tem que fornecer aos participantes uma visão geral dos pontos estratégicos do produto de software a ser construído e tomar as decisões executivas, tais como alocação de recursos, que podem afetar os requisitos e o projeto do novo produto;
* Representantes dos usuários: são as pessoas na empresa que irão utilizar o produto de software. Durante a extração de requisitos, os representantes são frequentemente gerentes ou pessoas-chave dentro da empresa que tem uma visão melhor do todo e de como ele será usado;
* Representantes de produtos de software: são pessoas que estão bastante familiarizadas com as capacidades dos produtos de software. Seu papel é ajudar os usuários a entender o que é razoável ou possível que o novo produto faça;
* Especialista: é a pessoa que pode fornecer informações detalhadas sobre um tópico específico.

O conceito do JAD de abordagem e dinâmica de grupo poderá ser utilizado para diversas finalidades, como: planejamento de atividades técnicas para um grande projeto, discussão do escopo e objetivos de um projeto e estimativa da quantidade de horas necessárias para desenvolver sistemas grandes e complexos.

A maioria das técnicas JAD funciona melhor em projetos pequenos ou médios. Para um sistema grande e complexo podem ser usadas múltiplas sessões JAD para acelerar a definição dos requisitos do sistema.

1. Boas Práticas e Padrões
   1. Requisito

Devem ser redigidos de modo a identificar o resultado a obter e não a forma de obtê-lo.

Evitar a definição de requisitos muito abrangentes, por exemplo: “o sistema deve cadastrar clientes”.

* 1. Requisitos Funcionais

Devem identificar de forma clara:

* O produto ou produtos finais e intermédios a obter;
* As funções que os mesmos devem realizar;
* As finalidades a que se destinam;
* Os objetivos, gerais e específicos, a atingir.

Exemplos:

Considere a necessidade “Fornecimento de um Sistema de Informação de Planejamento e Acompanhamento de Atividades”, o sistema deve:

* Registrar as ocorrências normais relativas à execução do plano de atividades (recepção de resultados, atualizações de datas de fim, atualização de recursos…) e obtenção de elementos de controle (avaliação custos/resultados, alertas, desvios);
* Registrar ocorrências que correspondam a alterações substanciais ao Plano – por exemplo, eliminação ou inclusão de projetos, eliminação ou desdobramento de fases de projeto, alterações com implicações orçamentárias.
* Registrar a situação final dos vários projetos/atividades no que diz respeito a consumo de meios e resultados obtidos, bem como o registro/atualização dos benefícios alcançados e a identificação dos projetos/atividades que passarão para o Plano de Atividades do ano seguinte.
  1. Requisitos Não Funcionais
     1. Performance

Definição do nível mínimo aceitável de desempenho do produto, incluindo:

* A carga de processamento a suportar numa determinada unidade de tempo. Por exemplo: o volume de transações que o sistema deve processar num determinado período de tempo.
* O nível de erro admitido, habitualmente expresso em percentagem.

Exemplos:

* Percentual de uptime do sistema.
* Tempo máximo de indisponibilidade do produto para efeitos de manutenção.
* Tempos de resposta (máximo e médio) de um sistema web.
  + 1. Integração

Aplicável para os casos em que o sistema deva interagir com outros sistemas existentes ou integrar-se num determinado contexto.

Exemplos:

* Caracterização da infra-estrutura onde determinado sistema vai ser hospedado.
* Caracterização dos sistemas e tipo de interfaces de um determinado sistema com outros.
  + 1. Segurança

Sempre que possível, fazer referência a normas técnicas nacionais e /ou internacionais.

* 1. Defina os atores

Geralmente os atores são perfis assumidos pelas pessoas que interagem com o sistema, porém, sistemas também podem ser considerados atores.

* 1. Defina os Casos de Uso Principais

São os casos de uso que provavelmente vão ser executados se tudo correr bem.

O foco inicial deverá ser nesses casos de uso, pois, você pode derivar os casos de uso menos usados depois.

Use a regra 80/20, se você desenhar uma lista detalhada de todos os possíveis casos de uso, geralmente, 20% deles serão responsáveis por 80% das atividades do sistema. Esses devem ser priorizados para o seu detalhamento, em seguida, trabalhe nos casos de uso secundários.

* 1. Identifique Oportunidades de Reuso de Casos de Uso

Reuso, por vezes, relaciona-se com as palavras “generalização”, “especialização” e “herança”. Generalização tem a ver com algo genérico, não detalhado. Ela acontece quando você “herda” de alguma coisa genérica e então adiciona detalhes, ou a especializa. Por exemplo: “Pessoa” é um termo muito genérico. “Homem” ainda é genérico, mas, não tanto quanto “Pessoa”. Pode-se dizer que “Homem” herda o comportamento e atribudos de “Pessoa”, em outras palavras “Homem” é uma “Pessoa”, mas, “Pessoa” nem sempre é um “Homem”.

Por exemplo, casos de uso que a princípio possam parecer repetidos, são bons candidatos a reuso. Analise os seguintes modelos de casos de uso de um sistema de e-commerce:



Perceba que ambos os atores possuem associação com os casos de uso “Criar Conta” e “Pesquisar Itens”. Ao invés de manter essas informações duplicadas, podemos desenhar de uma forma genérica:



Esse diagrama informa que os atores “Comprador” e “Vendedor” herdam do ator genérico “Usuário” seus relacionamentos, assim sendo, ao associar os casos de uso “Criar Conta” e “Pesquisar Itens” ao “Usuário” implica que os outros dois atores também estão associados a estes casos de uso.

* 1. Casos de Uso
     1. Por que Criar Casos de Uso

Relacionar as necessidades do cliente aos requisitos;

Definir as fronteiras do sistema;

Capturar e definir o comportamento do sistema;

Definir quem ou o que interage com o sistema;

Validar e verificar os requisitos;

É um instrumento de planejamento.

* + 1. O que NÃO DEVE ser escrito em um caso de uso

Detalhes de implementação como:

* “As informações do estudante devem ser gravadas no banco de dados”.
* “O sistema grava as informações do estudante”.

Detalhes de interface como:

* “O estudante clica no botão Selecionar para escolher um curso”.
* “O estudante escolhe um curso”.

Detalhes técnicos de processamento não relacionados ao requisito do usuário:

* “O sistema calcula o número de confirmação usando um algoritmo de hash e o número é exibido para o estudante”.
* “O sistema mostra o número de confirmação ao estudante”

Requisitos não funcionais

* “O sistema deve responder ao estudante em menos de 1 minuto”
* “O sistema deve estar disponível 24/7”
* “O sistema deve atender até 1000 usuários simultâneos”
  + 1. Regras de Negócio

Como não escrever: “O estudante informa seu número de identificação que deve ter de 6 a 8 caracteres alfanuméricos e contém um número que não pode ser o primeiro ou o último caractere”.

Mantenha o caso de uso simples:

* “O estudante informa seu número de identificação”.

Regras de negócio devem ser escritas em um artefato separado:

* “O número de identificação do estudante é usado para verificar se ele pode se registrar em algum curso. O número deve possuir de 6 a 8 caracteres alfanuméricos e contém um número que não pode ser o primeiro ou último caractere”.
  1. Nomenclatura dos Casos de Uso

O nome do caso de uso deve ser único e intuitivo, e deve identificar com clareza o resultado de valor que ele produz para algum ator do sistema. Deve seguir o formato:

***[Verbo Infinitivo + Substantivo ou Sentença]***.

Seguem algumas considerações para nomear os casos de uso:

* Não usar nomes extensos;
* Não usar nome de atores ou siglas de sistemas externos;
* Evitar o uso do plural;
* Buscar termos que reflitam o objetivo do caso de uso na totalidade.
  + 1. Estrutura de um caso de uso:

1. Introdução
2. Fluxo de Eventos
   1. Fluxo Básico
   2. Fluxos Alternativos
3. Requisitos Especiais
4. Pré-Condições
5. Pós-Condições
6. Pontos de Extensão

“Bons casos de uso são balanceados, descrevem o comportamento essencial do sistema e provêm apenas os detalhes necessários entre o sistema e seus usuários.”

* + 1. Escrevendo um Caso de Uso
       1. Faça uma breve descrição do caso de uso:

1. Parágrafo que descreve quem interage com o caso de uso e o propósito do caso de uso.
2. Não resumir o fluxo básico na descrição.

“O caso de uso Adicionar Curso é iniciado pelo Estudante e permite a inclusão de um curso ao calendário para um determinado semestre.”

* + - 1. Fazer um rascunho dos passos do caso de uso:

1. Ordenar temporalmente os passos do casos de uso (sentenças simples).
2. Concentre-se nos passos do fluxo básico.
3. Anote os maiores fluxos alternativos e exception.

O rascunho do fluxo de eventos e traz os seguintes benefícios:

* Permite uma visão da complexidade do caso de uso (quanto mais passos, mais complexos).
* Permite uma verificação inicial do caso de uso com o cliente a fim de determinar se o caso de uso está na linha certa.
* Provê insumos para a prototipação.

Exemplo:

Fluxo básico

* O Estudante efetua o login no sistema.
* O Estudante seleciona a opção de incluir um curso ao calendário.
* O Estudante informa o número do curso.
* O Sistema verifica que o estudante satisfaz os requisitos para o curso.
* O Sistema mostra uma lista de ofertas de curso abertas.
* O Estudante seleciona uma oferta de curso.
* O Estudante é registrado para o curso.

Fluxos Alternativos

* Pré-requisitos não satisfeitos.
* Não há ofertas de curso disponíveis.
* Estudante não pode ser registrado na oferta de curso.
  + - 1. Detalhar o fluxo

1. Adicione o detalhamento lembrando que cada passo deve mostrar quem faz a ação e qual o comportamento esperado do sistema.
2. O caso de uso está pronto quando seu fluxo está completo, seus termos estão descritos no glossário e todas as entradas e saídas estão descritas no documento.
3. Pergunte-se: Existe informação suficiente para a fase de prototipação, projeto e teste?
   * 1. Considerações Adicionais

* Um caso de uso não é uma função do sistema, é um processo de negócio. Por exemplo: em um sistema de e-commerce o processo de compra pode ser considerado um caso de uso único.
* Evitar a decomposição funcional.
* Avaliar se os casos de uso com poucos passos podem ser substituídos por uma ou duas regras de negócio, reduzindo a complexidade do modelo.
  1. Pré-condições
* É uma condição que deve ser verdadeira para iniciar o processo descrito no caso de uso.
* Não confundir com o evento que inicia o caso de uso.
* Não deve ser utilizada para representar carga das tabelas de apoio.
* Avaliar se um determinado fluxo alternativo pode ser simplificado como uma pré-condição.
* As pré-condições que são comuns a todos os casos de uso devem ser descritas como Requisitos Não Funcionais.
  1. Pós-Condições
* Descrevem as situações do sistema que podem estar satisfeitas ao final do caso de uso.
* Evite o óbvio, uma pós-condição não deve descrever o objetivo do caso de uso, por exemplo: No caso de uso “Manter Curso” não se aplica uma a seguinte pós-condição “Curso cadastrado com sucesso”.
  1. Seja produtivo sem buscar a excelência

Seja ágil. Seja simples. Seja completo.

Geralmente a perfeição é encarada como uma forma de impressionar os clientes, no entanto, é melhor você ter casos de uso completos, mesmo que com poucos erros, do que casos de uso excessivamente complexos e difíceis de ler.

* 1. E as Histórias de Usuário?

Algumas metodologias ágeis propõe a utilização de histórias de usuário que são, colocado de forma simplista: uma afirmação a respeito do sistema.

Como uma história de usuário é diferente de um caso de uso? Enquanto o caso de uso é um artefato bem estruturado, a historia está mais para uma necessidade do usuário. Analisando com mais detalhes as histórias podem ser consideradas insumos para os casos de uso.

Histórias de usuário são ótimas para coletar e priorizar funcionalidades de alto nível, porém, serão derivadas em requisitos e posteriormente em casos de uso.

* 1. Desenvolvimento Ágil com Casos de Uso

Um mito do desenvolvimento ágil é a necessidade de se utilizar histórias de usuário e não casos de uso. Na verdade, a agilidade está mais relacionada com iterações menores, lições aprendidas e melhorias incrementais.

Utilizar metodologias ágeis com casos de uso significa, inicialmente, fazer casos de uso mais simples que serão mais detalhados a cada iteração.

Escrever casos de uso muito grandes e complexos não estão de acordo com metodologias ágeis, pois, o desenvolvimento ficará muito extenso e pode não ser compatível com a idéia de iterações pequenas.

Mantenha os casos de uso simples.

1. Referências

* LEITE, J. C. **Requisitos de Software**. Disponível em: <engenhariadesoftware.blogspot.com.br/2007/05/requisitos-de-software.html>. Acesso em: 30 jul. 2013.
* QUATRANI, T. **Writing Good Use Cases**. Disponível em: <http://ecl.cs.ui.ac.id/PAUS/Files/Writing%20Good%20Use%20Cases.pdf>. Acesso em: 30 jul. 2013.
* MORAES, J. B. D. **Técnicas para Levantamento de Requisitos**. Disponível em: <devmedia.com.br/artigo-engenharia-de-software-2-tecnicas-para-levantamento-de-requisitos/9151>. Acesso em: 30 jul. 2013.