



# ENGENHARIA DE REQUISITOS

AULA 5



Profª Rosemari Pavan Rattmann



## CONVERSA INICIAL

Sabemos da importância em elicitar requisitos de forma clara, completa e que atenda a todas as partes interessadas para o desenvolvimento de um sistema de software e precisamos analisar os possíveis problemas que podem surgir, mesmo com este trabalho inicial muito bem definido. Por isso que abordaremos como administrar os conflitos gerados entre os envolvidos no processo, problemas e mudanças que surjam a pedido ou por necessidade, sempre buscando garantir o acordo sobre o escopo da solução, se é possível priorizar e quais seriam os requisitos, identificar a melhor forma de comunicar os requisitos e a maneira como será mantido o conhecimento obtido para uso futuro. Todo este processo trata a gerência de requisitos e existem padrões internacionais e do Brasil para ajudar nesta fase. Para um bom gerenciamento, se faz necessário um bom planejamento. Conversaremos sobre os padrões já estudados e sugeridos que garantirão uma boa solução, bem como o acompanhamento necessário de todos os planos e gestão de testes, mudanças, a importância da documentação atualizada e vários detalhes a serem mensurados sobre como gerir os requisitos. Veremos que a gestão de mudanças é o processo que avalia todos os pedidos e necessidades encontrados, avalia as autorizações de mudanças, conforme os acordos firmados e controla a comunicação com as partes interessadas, informando os requisitos alterados, os que foram rejeitados e aceitos, além de documentar tudo.

Vamos observar que um requisito precisa ter rastreabilidade para entender quem solicitou o requisito, por que o requisito existe, quais os requisitos relacionados e como os requisitos se relacionam às outras informações do sistema e, muito importante, como impactam nos diversos stakeholders, caso não seja implementado ou precisa ser alterado. Por fim, estudaremos como gerenciar os requisitos utilizando a priorização e garantindo a qualidade da especificação dos requisitos ao longo do tempo, contando com todas as possíveis mudanças que venham a aparecer.

## TEMA 1 – GERÊNCIA DE REQUISITOS

Retomando um pouco o processo de elicitação de requisitos, já estudado, desde o estudo de viabilidade, elicitação, especificação até confeccionar o documento de requisitos, durante todo este momento, os requisitos podem mudar! Como assim? A Figura 1 mostra todo este processo para refletirmos sobre o tema de gerenciamento.

Figura 1 – Processo de gerenciamento de requisitos



Fonte: Vazquez, 2016.

Ao longo da verificação com os stakeholders e com o conhecimento do negócio do cliente, algumas características dos requisitos podem ser alteradas, alguns novos requisitos podem surgir, ou ainda, alguns podem deixar de existir dependendo das mudanças. E todas as alterações relacionadas aos requisitos precisam ser bem gerenciadas porque as informações podem se perder e fazer falta no sistema a ser entregue, o produto pode ser implementado errado, ou seja, os requisitos bem especificados e atualizados determinarão o sucesso do sistema e o aceite do cliente. Então, o mecanismo que se tem para tratar, controlar e gerir as mudanças de requisitos é o que chamamos de gerência de requisitos.

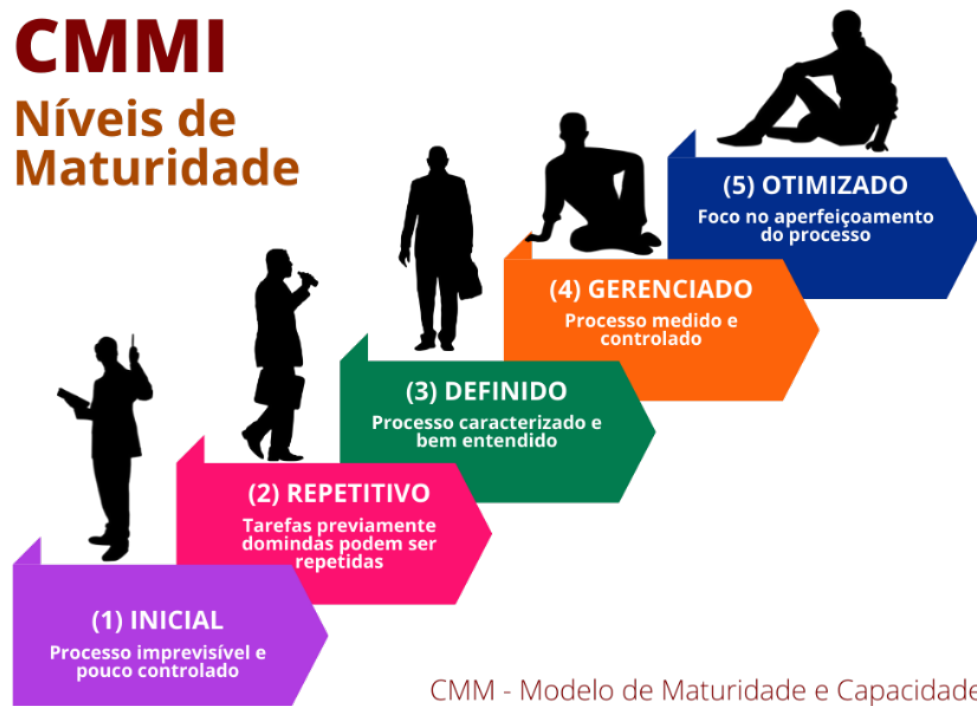
Podemos dizer que o processo de gerenciamento de requisitos é o processo que gerencia mudanças nos requisitos de um sistema. Deve ser organizado e possibilitar identificar, controlar e documentar os requisitos do sistema com suas alterações e definir e manter o acordo entre o cliente e a equipe de desenvolvimento do projeto, em relação aos requisitos variáveis do



sistema, ou seja, o que faz parte do escopo, o que foi autorizado, o que deve ser mantido e entregue como produto final.

Existem normas e padrões internacionais que as empresas adotam ou buscam nos produtos que comercializam que conferem qualidade. Vamos falar um pouco sobre este assunto para entender que o processo de gerência de requisitos é um ponto chave para avaliar a qualidade do produto desenvolvido e sua importância em todo o processo de desenvolvimento de software. As normas ISO – International Organization for Standardization (Organização Internacional para Normalização) representam uma organização internacional, não governamental, que elabora normas internacionais. O propósito das normas ISO é desenvolver e promover padrões mundiais que facilitem as negociações internacionais com base em critérios de qualidade e no Brasil, utilizamos a ABNT – Associação Brasileira de Normas Técnicas (ABNT), como representante das ISO. O CMMI surgiu durante a década de 1980 como um modelo para avaliação de risco na contratação de empresas de software pelo Departamento de Defesa dos Estados Unidos que desejava ser capaz de avaliar os processos de desenvolvimento utilizados pelas empresas que concorriam em licitações como indicação da previsibilidade. O CMMI foi desenvolvido pelo Software Engineering Institute – SEI, um departamento de pesquisa ligado à Universidade Carnegie Mellon, uma reconhecida instituição de ensino dos Estados Unidos. Embora não seja uma norma emitida por uma instituição internacional (como a ISO ou o IEEE), esta norma tem tido uma grande aceitação mundial, até mesmo fora do mercado americano. O modelo, publicado em 1992, pode ser obtido na própria Internet com facilidade. O CMMI (Figura 2) também é chamado de SW-CMM (Software CMM) e define uma área de processo chamada de Gerência de Requisitos. MPS-BR um modelo nacional, criado de acordo com a realidade das empresas brasileiras, visando a melhoria do processo de software no Brasil (Vazquez, 2016).

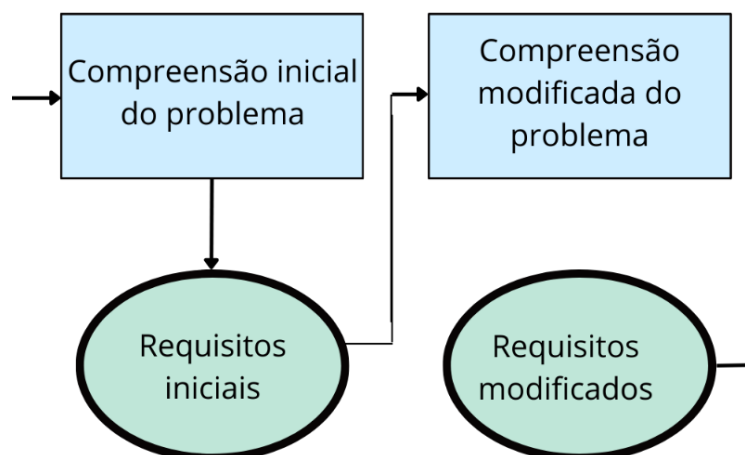
Figura 2 – Níveis de maturidade do CMMI



Crédito: Smile ilustras.

Podemos nos perguntar como podem ocorrer mudanças em requisitos uma vez que foram muito bem elicitados. Por que ocorrem, então? A Figura 3 demonstra a evolução dos requisitos.

Figura 3 – Evolução dos requisitos



Fonte: Rattmann.



Segundo Sommerville (2018), há requisitos que são mais propensos a sofrerem mudanças e outros mais estáveis e considerando uma perspectiva de evolução, os requisitos são divididos em duas classes.

- a) **Requisitos permanentes:** normalmente, fazem parte da atividade principal da organização e diretamente com o domínio do problema. Desta forma, tendem a ser mais estáveis.
- b) **Requisitos voláteis:** são os requisitos que vão se modificar durante o desenvolvimento do sistema ou depois que o sistema estiver em operação.

Mesmo sendo muito cuidadoso e realizando um bom processo de elicitação, o analista de requisitos não conseguirá evitar que existam mudanças e as razões podem ser de vários fatores, como listados:

- por existirem muitos stakeholders e várias fontes de informação, nem sempre os requisitos são óbvios e, portanto, podem não ser identificados;
- as pessoas têm dificuldades em expressar os requisitos claramente através da linguagem, seja ela escrita ou através de palavras e omitem informações relevantes;
- descoberta de erros no levantamento de informações;
- os usuários melhoram seu entendimento sobre suas próprias necessidades em relação ao software a ser desenvolvido;
- mudanças nas prioridades do cliente (da empresa);
- mudanças nas equipes do cliente que tem outros interesses de negócio;
- os requisitos têm propriedades exclusivas – por exemplo, eles não são igualmente importantes nem igualmente fáceis de cumprir;
- há várias partes interessadas, o que significa que os requisitos precisam ser gerenciados por grupos de pessoas de diferentes funções;
- há conflitos de interesse dos stakeholders e, caso o gestor não participe ativamente, a priorização dos requisitos pode atrapalhar na especificação;
- mudanças legais, novas leis a serem impostas que impactam nos requisitos já definidos;
- entre outros.

Mesmo depois de o software ser entregue aos usuários, novos requisitos surgem. Conforme o sistema vai sendo usado no dia a dia, as pessoas



descobrem novas necessidades e prioridades. Podem ser evoluções do software, novas versões, tudo para contemplar as mudanças que surgiram nos requisitos. Se não houver uma preocupação com estas mudanças, o software pode ficar sem sentido, sem uso, obsoleto, afinal, os requisitos representam as necessidades dos usuários e se não forem atendidas, perde-se o interesse.

Gerenciar requisitos é gerenciar o conhecimento adquirido sobre o software (Vazquez, 2016). Pois bem, então todas as mudanças solicitadas deverão ser levadas para a equipe de desenvolvimento? Um ponto bastante relevante neste momento é efetuar uma análise do impacto que a mudança causará, ou seja, quais requisitos estão envolvidos, qual o custo que será aumentado ou não para implementar, qual o tempo para efetivar a mudança, entre outros. Tudo deve ser analisado, bem como em que momento está o desenvolvimento do produto, se já iniciou, se está no meio ou já quase todo pronto. Para esta análise ser completa, será necessário identificar qual a fonte do requisito a ser alterado (quem solicitou) e qual a sua prioridade em relação ao todo.

Citando as tarefas do gerenciamento de requisitos, iniciamos pela necessidade de manter todos os requisitos em um repositório (como se fosse um armário de requisitos), para que nada seja perdido e todos os envolvidos tenham acesso aos mesmos requisitos. Avaliando que os recursos para os projetos são limitados, como o tempo e o custo, haverá restrições a serem consideradas, farão parte do escopo, quais requisitos manter, quais deixar para depois ou não implementar, ou seja, a priorização dos requisitos é uma tarefa do gerenciamento de requisitos. Uma grande dificuldade é avaliar quais são as relações entre os requisitos alterados e qual o impacto que causará, por isso, é tão importante rastrear os requisitos para poder avaliar estes impactos, sempre que uma mudança acontecer. O processo de validar as mudanças e aprová-las junto aos stakeholders, principalmente, daquelas que causem impactos relevantes.

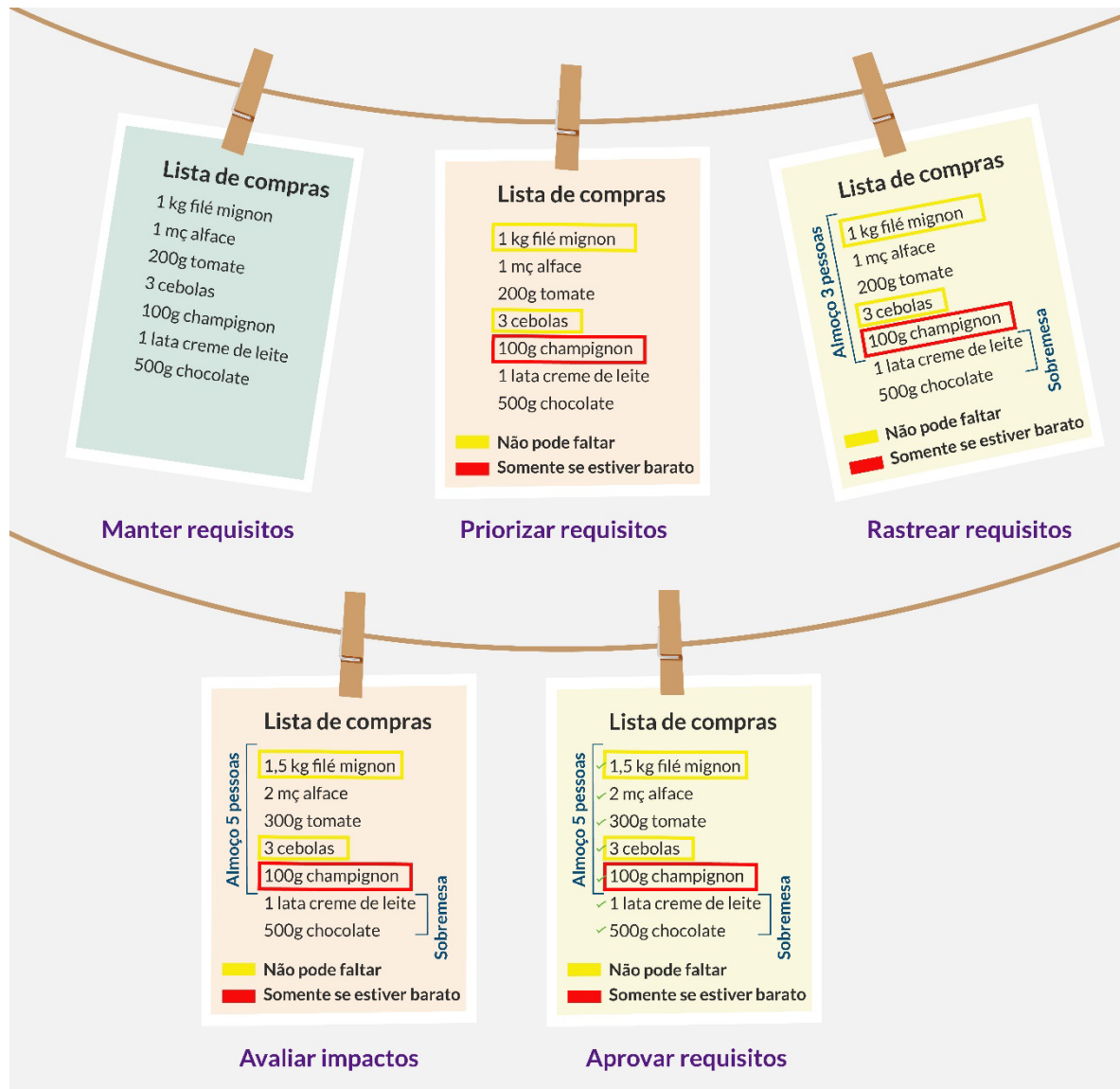
Resumindo, as tarefas da gerência de requisitos são:

1. manter os requisitos;
2. priorizar os requisitos;
3. rastrear os requisitos;
4. avaliar os impactos; e
5. aprovar os requisitos.



A Figura 4 mostra um exemplo prático de gerenciamento de mudanças nos requisitos, ao se preparar um churrasco.

Figura 4 – Manter, priorizar, rastrear, avaliar e aprovar a mudança de requisitos



Crédito: Noykosana/shutterstock.

A lista de compras inicial é definida (manter) e identificados os itens que não podem faltar (priorizar). É importante verificar como os produtos serão utilizados (rastrear). Um telefonema informou que mais dois amigos participarão, então será necessário identificar que quantidades de produtos precisam ser aumentadas (avaliar) e confirmar a lista e as quantidades para ir ao mercado (aprovar). Vale ressaltar que a cada solicitação de mudança de requisito, sendo aprovada, novas atividades de elicitação, análise e verificação deverão ser realizadas e um novo documento de especificação de requisitos será gerado,





uma versão atualizada e todos os stakeholders precisam estar cientes e envolvidos. As empresas estão em constantes mudanças, o comportamento dos usuários pode ser impactado por alguma necessidade não prevista, os mesmos usuários podem avaliar novas possibilidades com o software, algumas hipóteses levantadas nas fases iniciais podem se tornar equivocadas, enfim muitos são os motivos. Esta é a grande questão e problema enfrentado num projeto de desenvolvimento de software: como as mudanças de requisitos impactarão em outros requisitos que já foram implementados ou quais são os requisitos que ainda precisarão ser alterados em consequência da mudança solicitada.

São muitas questões a serem pensadas e negociadas. Quando não há um controle bem definido destas alterações, não se sabe os reais impactos que as mudanças podem causar e os custos e tempo envolvidos no projeto, além do controle de requisitos prioritários e outros menos importantes.

## **TEMA 2 – PLANO DE GERENCIAMENTO**

O plano de gerenciamento de requisitos responde como deve ser feita a gerência de requisitos e, normalmente, é criado pelo gerente de projetos e sua equipe de gestão do desenvolvimento do produto. Todas as ferramentas utilizadas para o desenvolvimento, o repositório, enfim, tudo que for necessário deve estar contido no plano. Mas, o responsável pela execução do plano pode ser o analista de requisitos e, também, a equipe de gestão de todo o projeto (Vazquez, 2016). Geralmente o analista de requisitos trabalha como apoio da equipe de gestão do projeto.

Quando falamos de gerenciamento de requisitos entendemos que estamos preocupados com o produto a ser gerado e que atenda ao que foi proposto/contratado. Já estudamos que os requisitos são fundamentais para este propósito. Como podemos definir que o software, após seu desenvolvimento, tem qualidade? Não há uma definição única para qualidade de software, nem um consenso entre profissionais de TI e empresas contratantes. Podemos citar algumas definições possíveis que nos levarão a pensar mais sobre o assunto, a seguir:



- Qualidade é estar em conformidade com os requisitos dos clientes;
- Qualidade é escrever tudo o que se deve fazer e fazer tudo o que foi escrito (NBR ISO 8402);
- Qualidade é atender ao escopo dentro do tempo previsto e custo estimado; e
- Qualidade é atender a todas as partes interessadas com necessidades e expectativas diferenciadas

Todo o processo de desenvolvimento de um sistema é composto por várias etapas técnicas até a entrega final e atendendo as definições especificadas pelo cliente. Há um esforço da equipe do projeto para realizar cada tarefa, no entanto, é necessário um controle e monitoramento de atividades para o cumprimento dos objetivos do projeto para cada tarefa, bem como o atendimento de prazos e custos e otimização de todos os recursos envolvidos. Assim sendo, faz-se necessário um bom gerenciamento do projeto.

Não se pretende aqui relatar todas as práticas de gerenciamento, acompanhamento e controle que um gerente de projetos deve dominar, nos deteremos no foco de gerenciamento de requisitos e seu controle de mudanças para garantir que as necessidades do cliente (e todos stakeholders) sejam atendidas, mesmo após alterações. Além das atividades fundamentais, como definição de escopo, cronograma, organização da equipe, entre outras atividades, vários planos devem ser pensados e formalizados para mitigar riscos ao projeto.

O gerenciamento de riscos para o desenvolvimento de software, formas de minimizá-los e técnicas implementadas em situações específicas são importantes para uma boa avaliação. Ainda, relativo ao tema de gerência de requisitos, abordaremos alguns planos de configuração, testes e proteção que auxiliam na identificação de riscos que podem abalar o desenvolvimento e o sucesso das entregas.

O plano de projeto é um documento que especifica o produto (software) a ser gerado e que atende às necessidades do contratante. Deve incluir, com detalhamento suficiente, o escopo e todas as atividades de cada etapa do ciclo de desenvolvimento, com estimativas de tempo, custos, responsabilidades para sua execução. Este plano também deve indicar os modelos estabelecidos para garantir a qualidade do produto e o controle de riscos envolvidos no processo. O



cliente deve ter acesso a este documento e pode acompanhar e intervir, conforme sinta necessidade, facilitando a interação e acompanhamento.

## 2.1 Plano de garantia de qualidade

O plano de qualidade descreve todas as verificações que serão impostas, as revisões e testes para avaliar e garantir o funcionamento do produto (software) dentro das especificações identificadas. Basicamente, o gerenciamento de qualidade é um acompanhamento sistemático das atividades envolvidas no desenvolvimento do produto. Como métodos podemos citar a revisões e auditorias dos requisitos propostos, reportando sempre que houver algum indício de falhas ou problemas que podem impactar em prazos e custos. No plano, deve-se incluir a definição de metas de qualidade, responsabilidades, análise de riscos, estrutura organizacional etc. (Kerr, 2015). A Figura 5 aponta importantes atributos de qualidade de software

Figura 5 – Atributos de qualidade de software

SEGURANÇA	COMPREENSIBILIDADE	PORTABILIDADE
Proteção	Tetabilidade	Usabilidade
Confiabilidade	Adaptabilidade	Resusabilidade
Resiliência	Modularidade	Eficiência
Robustez	Compexibilidade	Capacidade de aprender

Fonte: Rattmann

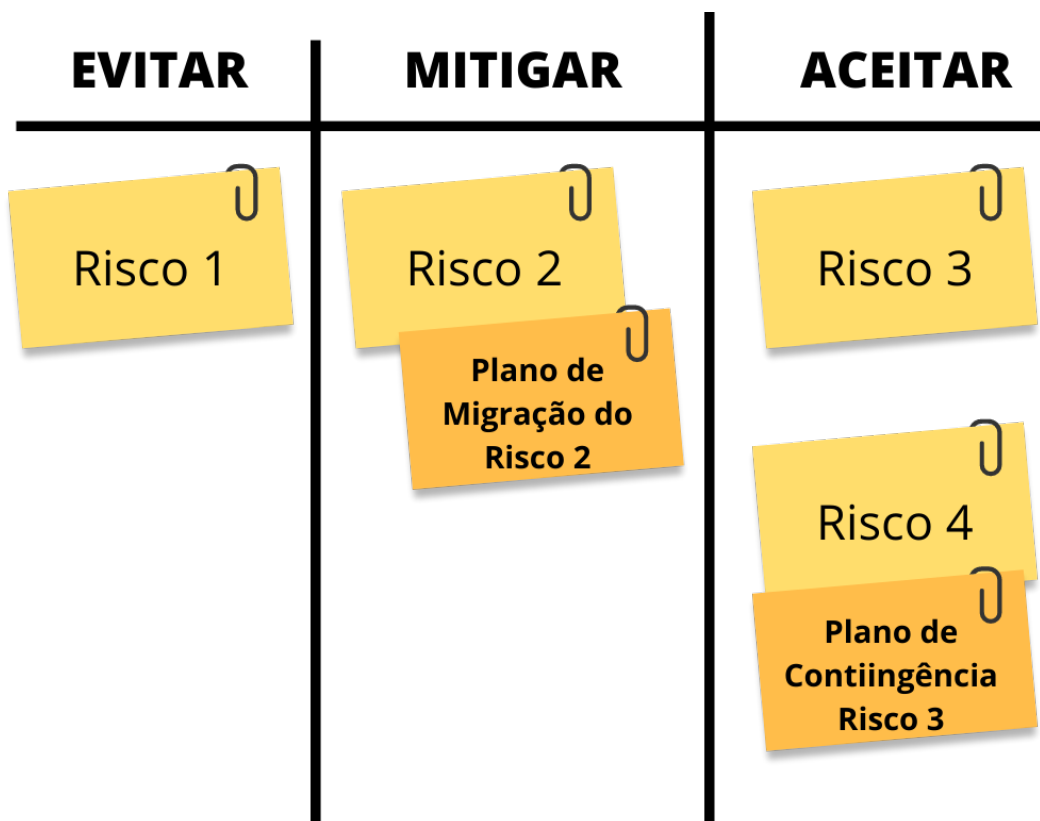
## 2.2 Plano de gerência de riscos

Todo gerente de projetos busca o cumprimento de prazos dentro do orçamento previsto e a entrega de um produto de qualidade, no entanto, vários problemas podem ocorrer ao longo do ciclo de vida do desenvolvimento que impactem no andamento esperado do projeto. Para tentar evitar estes eventos



indesejáveis, utiliza-se o plano de gerenciamento de riscos. Risco é quando se avalia e tenta quantificar as probabilidades de um problema acontecer e impactar no plano do projeto. Esta probabilidade por ser medida em zero (0-impossível de ocorrer) até um (1-certeza da ocorrência). Ao estabelecer as chances de problemas ocorrerem é possível avaliar os efeitos no projeto e produzir ações antecipadas para minimizar os impactos. A Figura 6 aponta as três formas de registrar o plano de riscos. Por exemplo, já se sabe que uma funcionalidade exigirá conhecimento profundo e especializado em computação em nuvem e, na equipe, não há este desenvolvedor com os conhecimentos necessários. O gerente do projeto pode tomar ações como antecipar um treinamento para desenvolver um integrante da equipe ou contratar um especialista, antes que o problema aconteça (atraso no desenvolvimento por falta de conhecimento técnico). Com os riscos identificados o gerente de projetos pode planejar formas de controlá-los e como reagir a eles: evitar os riscos, alterando requisitos; transferir para outras partes do sistema, isolando-os e minimizando os impactos; assumir e acompanhar e traçar estratégias para atuação quando ocorrerem os problemas (esperados).

Figura 6 – Plano de riscos

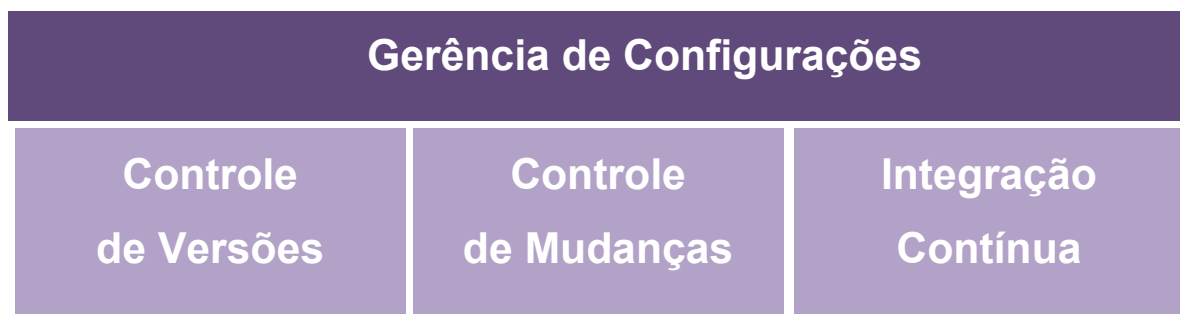




## 2.3 Plano de gerência de configuração

Descreve como as alterações nos requisitos e nos demais artefatos que tenham sido desenvolvidos serão elaboradas, ou seja, prevê como relatar as mudanças e avaliar os impactos das mesmas. Este plano se refere à capacidade de organizar e gerir como as alterações de requisitos serão resolvidas. Coordena o trabalho da equipe sobre um mesmo artefato, gerando versões do mesmo e garantindo que todas as últimas atualizações estejam válidas e ainda permitindo acesso às alterações anteriores, nas revisões. O objetivo principal é apoiar o processo para que todos os desenvolvedores possam trabalhar com os códigos e documentos do projeto de forma controlada. A Figura 7 exibe uma estrutura resumida.

Figura 7 – Plano de gerência de configuração



## 2.4 Plano de mudanças

As alterações nos requisitos podem se tornar um problema caso o sistema não esteja preparado para acomodá-las e o gerente de projeto deve evitar retrabalho da equipe e, conseqüente, perda de tempo e aumento de custo. A prevenção de potenciais alterações, como a geração de protótipos antes do desenvolvimento completo, é um bom exemplo (Figura 8).



Figura 8 – Plano de mudanças

CONTROLE DE MUDANÇAS					
Número	Solicitante	Data	Data avançada	Impacto	Observações
Número da solicitação de mudanças	(quem solicitou a mudança nos requisitos)	(data em que a mudança foi solicitada)	(data em que a solicitação de mudança foi avaliada pela equipe de analistas do projeto)	(impacto da mudança sobre o escopo, cronograma e orçamento)	(informações adicionais necessárias para esclarecer as mudanças)

No próximo tópico, analisaremos com mais detalhes o plano de mudanças e o gerenciamento.

## 2.5 Plano de testes

Descreve a forma de efetuar os testes unitários e integrais e como gerar os casos de testes de cada módulo ou funcionalidade, além de indicar os responsáveis pelas várias etapas de testes. Deve propor a forma como os testes ocorrerão e como os casos de testes serão gerados em cada parte do software e como testar a integração entre todas as partes. O documento de testes deve conter algumas informações padronizadas como (Figura 9):

- introdução – identificação e descrição dos objetivos;
- requisitos a serem testados;
- estratégias e ferramentas de teste;
- resultados esperados para cada etapa do teste; e
- relato dos resultados reais, após os testes.

### **Requisito:**

RQ1 - “ao consultar cadastro ou inserir cadastro, deve ser verificado o nível de acesso do usuário.”

### **Casos de Uso:**

UCS1 – Consultar cadastro

UCS2 – Inserir cadastro

UCS3 – Verificar permissões do usuário

### **Dependências:**

UCS1 e UCS2 possuem dependência de UCS3.

UCS3 é utilizado em UCS1 e UCS2.

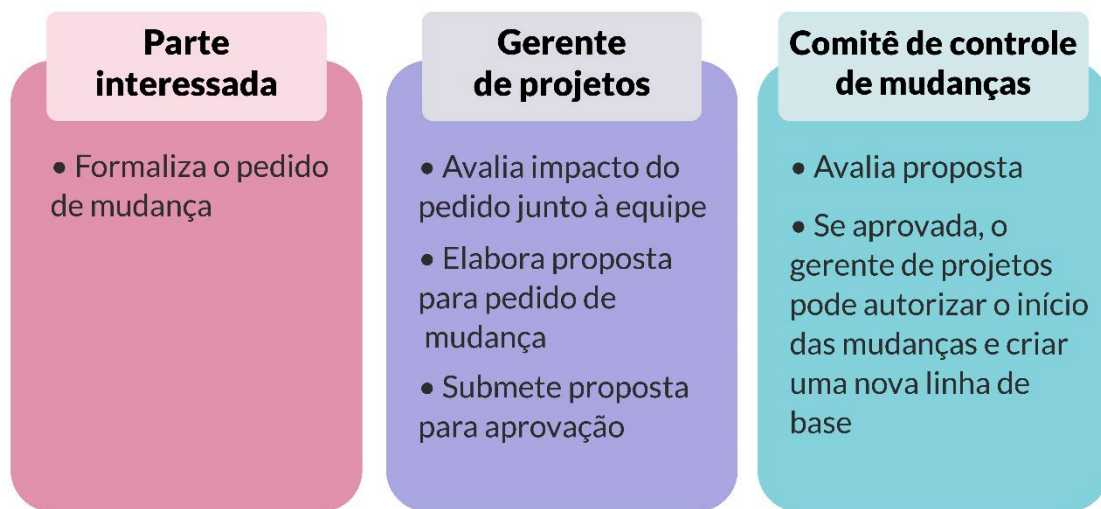
As Organizações precisam definir políticas internas de gerência de projetos que contemplem a gerência de mudanças de requisitos (estudaremos adiante os detalhes). É importante que o processo de solicitação de mudanças seja formal, documentado, que as informações necessárias para processar cada solicitação de alteração sejam completas, quem está solicitando, justificativas, custos e benefícios e, principalmente, quais os impactos em não alterar e efetuando as alterações solicitadas.

## **TEMA 3 – GESTÃO DE MUDANÇAS DE REQUISITOS**

A gestão de mudanças é o processo que avalia todos os pedidos de mudanças e controla as que foram aceitas ou rejeitadas, para comunicar à equipe de desenvolvimento, além de documentar tudo.

O pedido de mudanças pode ser feito por qualquer parte interessada e deve ser documentado e formalizado. A Figura 10 mostra alguns papéis no processo de gestão de mudanças, sugerido pelo PMBOK Guide.

Figura 10 – Papéis no processo de mudanças



É de fundamental importância que as alterações dos requisitos sejam:

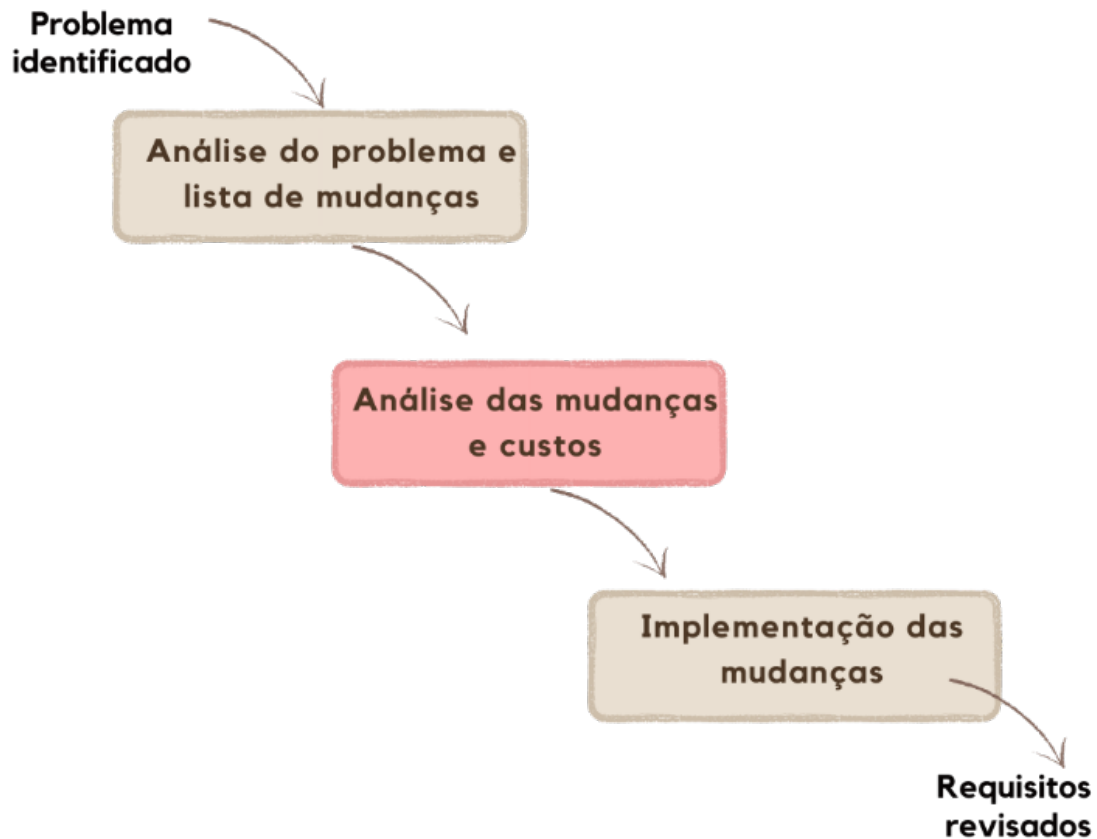
- identificadas e avaliadas;
- avaliadas sob o ponto de vista de risco;
- documentadas;
- planejadas;
- comunicadas aos grupos e indivíduos envolvidos; e
- acompanhadas até a finalização.

Segundo Vazquez (2016), para se ter uma gerência de requisitos eficaz é necessário definir um conjunto de objetivos para o processo e transmitidos para todos os integrantes da equipe. Todos os artefatos (documentos) produzidos durante o desenvolvimento do software devem ser gerados levando-se em conta padrões externos e corporativos, de modo a assegurar consistência e uniformidade das informações. Políticas bem definidas para a gerência de configuração, controle de mudanças, rastreabilidade e garantia da qualidade precisam ser colocadas em prática de modo a viabilizar um processo dinâmico e eficaz de gerência de requisitos.

Para que uma mudança possa ser aceita ou rejeitada, sempre deve ser feito uma análise do impacto sobre todo o projeto (Figura 11), mesmo que seja apenas em um requisito. Isto ocorre porque um requisito pode impactar em outros, mesmo que não estejam diretamente relacionados (Vazquez, 2016).



Figura 11 – Análise sobre o impacto das mudanças de requisitos



As solicitações de mudança aprovadas demandarão novas fases de elicitação e análise de requisitos, gerando nova versão do documento de especificação de requisitos. E, claro, conforme já estudamos, deve passar pela aprovação das partes interessadas e uma revisão do escopo a ser atendido, tempo e custo, consequentemente.

O processo para obter a aprovação sobre os requisitos alterados não é simples e exige da equipe que gerencia esta fase, um grande esforço. E por que é tão necessário obter esta aprovação junto ao cliente? A resposta está na necessidade de se manter um contrato entre as partes para que todas as mudanças e o que está sendo desenvolvido esteja claro e haja um consenso formal para evitar que, no futuro, algum stakeholder venha a questionar as alterações. Como, geralmente, há muitas partes interessadas e cada uma com suas necessidades, podem acontecer novos conflitos quando se fala em alterações que podem levar a novas mudanças, inclusive no escopo inicial do projeto. Se o projeto for pequeno e uma quantidade menor de partes interessadas, tendo em vista que o analista de requisitos já trabalhou com a equipe e há uma boa comunicação, este processo de aprovação tende a ser



tranquilo e rápido. Porém, em projetos maiores, faz-se necessário formalizar todas as aprovações e as dúvidas geradas. Todo este processo precisa estar anotado, documentado e com o resumo do que foi decidido.

Existe uma técnica chamada *controle de questões* que visa garantir que todas as perguntas feitas tenham respostas e que cada mudança verifique novas perguntas a serem respondidas. Este processo auxilia no entendimento das necessidades de todas as partes interessadas. Esta técnica ajuda a rastrear, gerenciar e resolver questões e outras preocupações que precisam ser resolvidas em relação aos requisitos do projeto e possíveis mudanças (Vazquez, 2016). Algumas questões representam conflitos entre as partes interessadas que precisam ser resolvidos, defeitos a corrigir na especificação ou premissas que precisem ser confirmadas.

#### TEMA 4 – RASTREABILIDADE DE REQUISITOS

O desenvolvimento de software, principalmente aqueles mais complexos e de criticidade alta, precisam ser confiáveis e a exigência de qualidade é cada vez maior. A rastreabilidade é um importante instrumento neste sentido, sendo um processo para identificar e documentar os vínculos que envolvem os requisitos, como uns dependem ou impactam nos demais, desta forma, rastreando a origem, os artefatos derivados e todos os demais envolvidos (Vazquez, 2016). Segundo Kerr (2015), *rastreabilidade* refere-se à possibilidade de identificar a fonte e as consequências dos requisitos. Isto é muito útil quando ocorrem alterações na especificação dos requisitos, pois, a partir da rastreabilidade é possível localizar a relação entre os requisitos alterados e o ponto onde se deve alterar. A Figura 12 ilustra uma mudança no meio de um processo da já estabelecido, justamente, o que ocorre no ciclo de vida do projeto e que precisa ser rastreado para avaliar os impactos.



Figura 12 – Mudanças sempre acontecem nos requisitos



Créditos: AndriiYalanskyi/Shutterstock.

O propósito da rastreabilidade é estabelecer um processo que ajude a:

- compreender a origem dos requisitos;
- gerenciar o escopo do projeto;
- gerenciar mudanças nos requisitos;
- avaliar o impacto no projeto da mudança em um requisito;
- avaliar o impacto de um defeito de teste nos requisitos;
- verificar se todos os requisitos do sistema são desempenhados pela implementação; e
- verificar se o aplicativo faz apenas o que era esperado que ele fizesse.

Outro benefício importante da rastreabilidade é que ajuda a garantir a conformidade dos requisitos em relação ao escopo do projeto e mudanças solicitadas, permite uma visão mais clara dos riscos, tempo custo e da comunicação entre as várias partes interessadas e equipe de desenvolvimento.

Um requisito é rastreável se for possível identificar quem solicitou o requisito, por que o requisito existe, quais os requisitos relacionados e como os requisitos se relacionam às outras informações como design de sistemas, implementações e documentos do usuário.



Boas práticas de gerenciamento de requisitos, como uma manutenção de dependências entre requisitos, têm benefícios em longo prazo, como maior satisfação do cliente e custos de desenvolvimento mais baixos. Uma vez que os retornos não são imediatos, o gerenciamento de requisitos pode parecer uma despesa desnecessária. Entretanto, sem a gerência, a economia de curto prazo será devastada pelos custos em longo prazo. Logo, todo sistema deve ser desenvolvido de modo que as alterações sofridas ao longo do seu desenvolvimento sejam o menos impactante possível. O processo de mudança dos requisitos precisa ser controlado de modo a garantir a qualidade do sistema. O impacto destas mudanças precisa ser avaliado e compreendido de modo que a sua implementação seja feita de maneira eficiente e a baixo custo (Vazquez, 2016).

Temos dois tipos de rastreabilidade de requisitos:


- horizontal e vertical; e
- pré e pós-rastreabilidade.

Segundo Vazquez (2016), a capacidade de rastrear um requisito até seus refinamentos é definida como rastrear para frente (horizontal) e a capacidade de rastrear um requisito até sua origem, ou seja, para trás (vertical). O nome conhecido para estes processos é de rastreabilidade bidirecional e sempre deve ser executado desta forma para cumprir os seus objetivos.

A rastreabilidade horizontal estabelece a dependência entre os requisitos em um mesmo nível, por exemplo, rastreabilidade dos requisitos entre si ou rastreabilidade entre códigos, ou seja, avalia as suas versões ao longo do ciclo de vida do projeto, pois requisitos são refinados e possuem versões e tem a capacidade de avaliar impactos sobre as mudanças pretendidas nestes requisitos.

A rastreabilidade vertical auxilia a determinar se todos os requisitos fonte foram completamente tratados e se todos os requisitos de mais baixo nível ou códigos de unidade podem ser rastreados para um requisito fonte válido.

Na rastreabilidade vertical são relacionados os requisitos de distintas fases ao longo do ciclo de vida do projeto, partindo-se de uma origem e listando todos os requisitos relacionados a este primeiro. Pensando no exemplo da Figura 13, uma solicitação de alteração em um requisito implementado em um caso de uso (funcionalidade), devem ser verificadas as consequências para UCS1 e



UCS2. Dependendo das alterações em UCS1 ou UCS2, pode ser necessário alterar UCS3.

Figura 13 – Exemplo de rastreabilidade bidirecional vertical

### **Requisito:**

RQ1 - “ao consultar cadastro ou inserir cadastro, deve ser verificado o nível de acesso do usuário.”

### **Casos de Uso:**

UCS1 – Consultar cadastro

UCS2 – Inserir cadastro

UCS3 – Verificar permissões do usuário

### **Dependências:**

UCS1 e UCS2 possuem dependência de UCS3.

UCS3 é utilizado em UCS1 e UCS2.

A rastreabilidade vertical bidirecional possibilita rastrear requisitos a códigos implementados. Este tipo de mecanismo de rastreabilidade é essencial para uma análise de impacto de mudanças de requisitos, identificando de que forma uma mudança de requisito impacta nos planos do projeto que contêm as estimativas aprovadas de esforço e custo para os produtos de trabalho e tarefas, e em códigos já implementados que precisem ser modificados (PDS).

Outro tipo de rastreabilidade é a pré e pós. A pré-rastreabilidade está concentrada no ciclo de vida dos requisitos antes de serem incluídos no documento de especificação de requisitos. A pós-rastreabilidade, está concentrada no ciclo de vida dos requisitos após serem incluídos na especificação de requisitos. A principal diferença entre elas envolve as informações com que podem lidar. A pré-rastreabilidade foca em identificar a origem dos requisitos que compõem a especificação de requisitos. A pós-rastreabilidade foca em identificar quais componentes implementam determinado requisito. Como exemplo, o documento de arquitetura e os casos de teste que são típicas da pós-rastreabilidade, ou seja, tem-se informações sobre como e onde os requisitos foram implementados (ou estão sendo implementados).



A análise do impacto causado por mudanças nos requisitos tem o objetivo de avaliar quantidade de esforço e custo a ser despendido, além de identificar quais partes do software precisam ser alteradas e estes processos de rastreabilidade podem ser mais facilmente visualizados através de uma matriz de rastreabilidade.

Colocam-se os requisitos avaliados e os casos de uso relacionados numa tabela e marcam-se os eixos de interseção. Isto facilita a visualização da rastreabilidade documentada entre os requisitos e outros objetos. No entanto, caso o projeto seja muito grande e com muitos requisitos, é provável que seja necessário o uso de uma ferramenta especializada devido ao esforço de avaliação e manutenção da matriz.

São vários tipos de matriz de rastreabilidade.

- **Matriz de rastreabilidade entre funcionalidades:** mostra o relacionamento entre partes do sistema visíveis aos clientes/usuários.
- **Matriz de rastreabilidade de fontes:** permite identificar a fonte, a origem de cada requisito.
- **Matriz de rastreabilidade de dependências:** essa é a forma mais comum da matriz, e identifica os relacionamentos entre os requisitos.
- **Matriz de rastreabilidade de subsistemas:** relaciona os requisitos pelos subsistemas a que estão relacionados.
- **Matriz de rastreabilidade de interfaces:** identifica como os requisitos se relacionam com as interfaces internas e externas do sistema.

A matriz de rastreabilidade mais utilizada é a de dependências entre requisitos e casos de uso, como no exemplo da Figura 14.



Figura 14 – Exemplo de matriz de rastreabilidade da dependência entre requisitos e casos de uso

RELAÇÃO ENTRE REQUISITOS	RF-001	RF-002	RF-003	RF-004
RF-001	✓		✓	✓
RF-002		✓	✓	
RF-003	✓			✓
RF-004		✓	✓	
RF-005		✓		

O responsável pela gerência do projeto, tendo estas informações, pode estabelecer negociações com o cliente para atender as solicitações de mudanças de requisitos, com informações sobre custos e prazos que são riscos inerentes de mudanças.

## TEMA 5 – PRIORIZAÇÃO

É bem provável que os requisitos tenham graus de importâncias diferentes e a especificação de requisitos atende a este item de qualidade quando explicita uma priorização de requisitos em função de sua importância ou pelas chances de possíveis alterações que venham a ocorrer (Kerr, 2015). Esta verificação de prioridade envolve mediação de interesses conflitantes entre os stakeholders e, provavelmente, será necessário convocar reuniões entre os envolvidos para obter um consenso sobre os requisitos obrigatórios e aqueles que podem ser implementados posteriormente. Assim, as decisões a serem tomadas tornam-se formais e com todos os envolvidos cientes, evitando queixas nas entregas futuras.

O principal objetivo da priorização de requisitos é ter informações sobre a decisão de quais são as coisas mais importantes para serem tratadas em primeiro lugar.

Vários critérios podem ser utilizados para ajudar na priorização, como exhibe a Figura 15.





Figura 15 – Critérios para a priorização de requisitos

CRITÉRIOS DE PRIORIZAÇÃO	COMO PRIORIZAR REQUISITOS?	QUANDO UTILIZAR?
Valor de negócio (benefício)	Com base na análise de custo-benefício	Projetos com limitações orçamentárias
Risco	Com base maiores riscos de falha para o projeto	Quando há conflitos de requisitos
Custo		
Perdas	Com base nos custos de implementar os requisitos	As partes interessadas podem mudar prioridades
Dependência com outros requisitos	Com base nas perdas ocasionadas por não implementar o requisito	Maximizar a probabilidade de sucesso do projeto
Estabilidade	Com base na dependência e importância do requisito	
Sensibilidade temporal	Com base no consenso obtido pelas partes interessadas e tempo	

Falaremos sobre algumas técnicas utilizadas para priorização:

- Técnica timeboxing/budgeting;
- Técnica Votação; e
- Técnica Análise de Moscow.

## 5.1 Técnica timeboxing/budgeting

Quando não se impõem limites nos tempos, certas tarefas podem sair do controle e, o que deveria ser realizado em algumas horas, acaba utilizando um dia inteiro. A técnica *timeboxing* se baseia em priorizar requisitos que possam ser implementados pela equipe num determinado período de tempo, enquanto *budgeting* prioriza dentro de um orçamento fixo. Segundo Vazquez (2016), uma estratégia para selecionar os requisitos é identificar os de mais alta prioridade e adicionar ou remover da lista, até que se encontre uma combinação que cumpra os limites de prazo e orçamento. Um exemplo comum de priorização por *timeboxing* e na metodologia Scrum, muito usada em projetos com métodos ágeis, que estudaremos adiante. A Figura 16 apenas ilustra a importância do



equilíbrio necessário entre o planejamento, o tempo e o custo e ainda uma priorização adequada aos requisitos de maior importância no escopo do projeto.

Figura 16 – Equilíbrio de tempo, custo e priorização de requisitos



Créditos: BRO.vector/Shutterstock.

## 5.2 Técnica Votação

A técnica de priorização por votação considera o voto de cada participante (stakeholder) entre os requisitos propostos em relação ao tempo e outros elementos. Os requisitos que receberem uma maior quantidade de recursos serão priorizados. As áreas de negócio da empresa devem conversar para identificar as prioridades, sempre dependendo do tamanho do projeto e da palavra final do dono do projeto, no caso a diretoria ou o proprietário da empresa. Esta técnica não é muito comum em desenvolvimento de software, mas pode ser muito útil quando se pesquisa entre vários clientes ou consumidores, quais funcionalidades devem ser implementadas nas próximas versões de um produto já distribuído.

### 5.3 Técnica Análise de Moscow

O termo *MoSCoW* (*Must have, Should have, Could have and Won't have*) — Figura 17 — é uma das técnicas de priorização, geralmente utilizada com *timeboxing/budgeting*, na qual é fixado e o foco nos requisitos mais importantes. Ajuda a comunicar o que será feito de imediato ou não.

Traduzindo os termos:

- *Must Have* (Tenho que fazer);
- *Should Have* (Devo fazer);
- *Could Have* (Poderia fazer); e
- *Won't Have* (Não vou fazer).

Figura 17 – Técnica de priorização de requisitos Moscow



A técnica é atribuir aos requisitos de um software os valores possíveis, conforme o nome da técnica.



- **M (deve ter):** deve ser um requisito ou grupo de requisitos obrigatórios para o projeto, ou seja, sem estes requisitos o projeto não terá sucesso. Por exemplo: não se consegue entregar o projeto sem este requisito.
- **S (deveria ter):** requisito ou grupo de requisitos importantes, mas não essenciais para o projeto. No entanto, é um requisito que precisa ser implementado, pois faz parte diretamente do escopo e do que os stakeholders precisam. A solução entregue, sem este requisito, é viável.
- **C (poderia ter):** é um requisito ou grupo de requisitos desejável, mas não imprescindível para o projeto. É um requisito que será incluído somente se houver tempo e dentro do orçamento e não influencia diretamente no resultado do projeto, não afetando muito a satisfação do cliente.
- **W (não terá agora):** é um requisito ou grupo de requisitos dispensável no momento e que poderá ser incluído em outros ciclos do projeto.

A priorização pode mudar ao longo do ciclo de vida do projeto, bem como os requisitos e a maior dificuldade não está nas técnicas utilizadas, mas nas decisões necessárias, pois fazer escolhas pode impactar em deixar funcionalidades para trás. Se perguntar ao cliente (e partes interessadas), praticamente tudo é prioritário, mas os requisitos precisam ser identificados em relação ao seu nível de importância e, muitas vezes, a priorização fica a cargo da equipe de projetos. Priorizar tudo é não priorizar.

## FINALIZANDO

Vimos que, por mais bem especificada a documentação de requisitos, ainda acontecerão solicitação de mudanças, requisitos que não haviam sido identificados, ou que estavam escondidos. A compreensão dos stakeholders ao longo do ciclo do projeto evolui e novos requisitos podem surgir, e por este e outros motivos observamos que é necessária uma gestão sobre estas mudanças. Sem esta preocupação, ao final do desenvolvimento, pode ser que o software já não faça mais sentido.

Abordamos a necessidade de planejar o projeto e criar várias opções a serem consideradas de forma a obter um checklist de verificação, como os planos de qualidade, de riscos iminentes, de configuração, de mudanças e de testes. Vimos como estes planejamentos são importantes e a documentação



formal das mudanças sugeridas para minimizar impactos e comunicar corretamente a todos os envolvidos.

Todo o processo de mudanças necessita de controle e o que chamamos de gestão de mudanças incorpora as autorizações dos stakeholders, quais requisitos e como rastreá-los a fim de manter tudo que foi solicitado sendo planejado para ser realizado.

Aprendemos que realizar a rastreabilidade do projeto é importante para controlar as mudanças de requisitos, perceber se existem inconsistências nos requisitos e verificar se o sistema atende o que foi solicitado pelo cliente e isto tende a ganhar eficiência e qualidade no produto desenvolvido.

E, por fim, vimos que a priorização de requisitos é muito importante para um bom gerenciamento dos recursos envolvidos, do tempo, de manter o custo dentro do estimado e de garantir a qualidade da especificação dos requisitos ao longo do tempo.



## REFERÊNCIAS

KERR, E. S. **Gerenciamento de Requisitos**. 1. ed. São Paulo: Pearson, 2015.

PMKB – Project Management Knowledge Base. **Principais Causas de Fracasso em Projetos**. Belo Horizonte, 29 dez. 2015. Disponível em: <<https://pmkb.com.br/artigos/principais-causas-de-fracasso-em-projetos/>>.

Acesso em: 10 jul. 2021.

SOMMERVILLE, I. **Engenharia de software**. Pearson. 6. ed. São Paulo, 2018.

VAZQUEZ, C.; SIMÕES, G. **Engenharia de Requisitos: Software Orientado ao Negócio**. 1. ed. Rio de Janeiro: Brasport, 2016.