



ENGENHARIA DE REQUISITOS

AULA 1

Profª Rosemari Pavan Rattmann



CONVERSA INICIAL

Seja bem-vindo(a)! Esta aula apresentará vários conteúdos que são fundamentais para os nossos estudos, compondo os conhecimentos necessários para a sua formação profissional dentro da engenharia de software.

Entenderemos o que é a engenharia de requisitos, os motivos para ser classificada como uma “engenharia”, que significado isso agrega, por que tem esse nome e como está inserida dentro da engenharia de software, sendo uma disciplina tão importante e que pode trazer benefícios e prejuízos, dependendo de sua aplicação e qualidade, podendo também impactar o custo do projeto.

Várias podem ser as causas que levam um projeto de software a fracassar ou a possibilitar falhas de operação ou rejeição do cliente. Veremos os principais erros, além de aprendermos como mitigar (minimizar) as causas que levam ao fracasso, mesmo para um projeto bem desenvolvido e, aparentemente, sem falhas de operação.

Conduziremos reflexões sobre as funções desempenhadas pelo analista de requisitos e de outros membros da equipe de desenvolvimento, suas principais habilidades e qual o seu papel dentro da engenharia de requisitos. Avaliaremos o valor de uma comunicação clara para o processo de desenvolvimento e descobriremos quais são as aptidões e características necessárias para um bom analista de requisitos.

Na execução de todas as atividades de elicitação de requisitos, veremos que existem dificuldades e elegeremos as mais comuns e como superá-las, ao especificar todos os requisitos para uma solução de software, reduzindo os impactos negativos quando essa fase não é bem detalhada.

TEMA 1 – ENGENHARIA DE REQUISITOS

O termo *engenharia* é descrito pelo dicionário Priberam da Língua Portuguesa como um “Conjunto de técnicas e métodos para aplicar o conhecimento técnico e científico na planificação, criação e manutenção de estruturas, máquinas e sistemas para benefício do ser humano”. A engenharia consiste em efetuar tarefas contínuas com modelos matemáticos, técnicos e científicos, para obter resultados objetivos e assertivos sobre um determinado problema.



Dessa forma, é adequado utilizar o termo *engenharia de requisitos* para identificar, organizar, documentar e rastrear os requisitos de um software, não apenas num primeiro momento, mas continuamente, e levando em consideração o cliente e suas necessidades que podem mudar antes mesmo que uma solução esteja completamente definida.

Esse processo chamado de engenharia de requisitos visa exatamente entender e explicitar o que é necessário para entregar um software adequado e que atenda ao negócio do cliente.

A engenharia de requisitos é a primeira disciplina a ser pensada quando se propõe estudar o desenvolvimento de um novo software e é fundamental para todo o trabalho a ser empregado no desenvolvimento do mesmo, como demonstra a Figura 1.

Os requisitos estão ao centro de todo o planejamento do ciclo de vida do projeto (veremos a seguir esse conceito). As fases de planejamento (*engineering*), projeto (*design*) e construção (*construction*) interagem entre si, baseadas nos requisitos apresentados e, ainda, interferem nos testes (*testing*), nas necessidades de configuração do ambiente (*configuration*) e nas futuras manutenções (*maintenance*) ou mudanças necessárias após a implantação e entrega.

Em todas as fases de desenvolvimento, a constante avaliação sobre os requisitos possibilita confirmar que todas as necessidades estejam contempladas na solução que está sendo construída.

Engenharia de requisitos é um processo que engloba todas as atividades que contribuem para a produção de um documento de requisitos e sua manutenção ao longo do tempo (Vazquez, [S.d.]) e tendo identificado todos os requisitos, os projetistas poderão projetar a melhor solução que atenda ao solicitado pelo cliente ou dono do software.

A identificação de requisitos é o primeiro passo no desenvolvimento de um sistema (de um software).



Figura 1 – A importância dos requisitos no desenvolvimento da solução



Créditos: EtiAmmos/Adobe Stock.

1.1 Fundamentos de requisitos de software

No mundo atual, tudo envolve softwares, desde grandes sistemas controladores de equipamentos industriais até pequenos aplicativos que oferecem dicas de receitas. Para que o funcionamento seja eficiente e que as pessoas queiram adquiri-los, obviamente, precisam ser executar os objetivos aos quais se propõem.



Se você quiser obter um endereço, espera que o software lhe responda rapidamente, com dicas sobre as redondezas, a melhor rota e, claro, que o ajude a chegar no exato endereço que solicitou. Estes são exemplos de requisitos que vocês espera e o que o analista de requisitos que irá desenvolver esse software, precisa identificar para que tenha sucesso e o cliente – você – fique satisfeito com o resultado.

Os sistemas estão trazendo problemas mais complexos para serem resolvidos por software e envolvem equipes, não sendo suficiente apenas um analista desenvolvedor. Cada integrante tem um papel a ser desempenhado e partes a serem especificadas e projetadas que se interconectam ao final, ou seja, se uma parte não ficou bem resolvida, pode afetar todo o trabalho de uma equipe e o resultado final ficar comprometido.

O guia *Software Engineering Body of Knowledge* (Conjunto de conhecimentos em engenharia de software), conhecido pela sigla SWEBOK, é um documento criado sob o patrocínio da IEEE Computer Society e publicado por esta a fim de promover a profissionalização e criar um consenso sobre as áreas de conhecimento da Engenharia de Software e seu escopo (Vazquez, [S.d.]).

Vários estudos e pesquisas de entidades que visam a qualidade de software, como o PMI – PMBOK, apontam como a maior causa para o fracasso do projeto foi a definição incompleta ou mal feita sobre os objetivos essenciais dele, ou seja, requisitos mal especificados levaram ao fracasso.

O PMI – Project Management Institute – é uma instituição internacional que não tem fins lucrativos e reúne profissionais de gestão de projeto do mundo todo, preocupando-se em acompanhar, identificar e compartilhar as melhores práticas. Publicaram um guia sobre gestão de projetos chamado PMBOK, que é amplamente divulgado e estudado pelas organizações (PKMB).

Podemos avaliar a qualidade de um software pela quantidade de erros de operação ou execução, no entanto, mesmo que não existam falhas, se não executar o que se esperava, tende a não ser utilizado e a ter um cliente insatisfeito. Deve existir uma boa comunicação com todos os envolvidos, aqueles que utilizarão o software para se ter uma lista completa sobre todas as necessidades, ou seja, todos os requisitos daquele software.

O desenvolvimento de um sistema exige um plano sobre o qual se definem todas as tarefas, e elas são organizadas ao longo do tempo, com o



objetivo final de entregar o sistema pronto, funcionando e que atenda as necessidades de quem o contratou, dentro do prazo e custo contratados. A Figura 2 nos leva a entender que ocorrem várias reuniões com o cliente e entre os membros da equipe para definições e entendimento das necessidades daquele projeto.

Figura 2 – Reunião do projeto



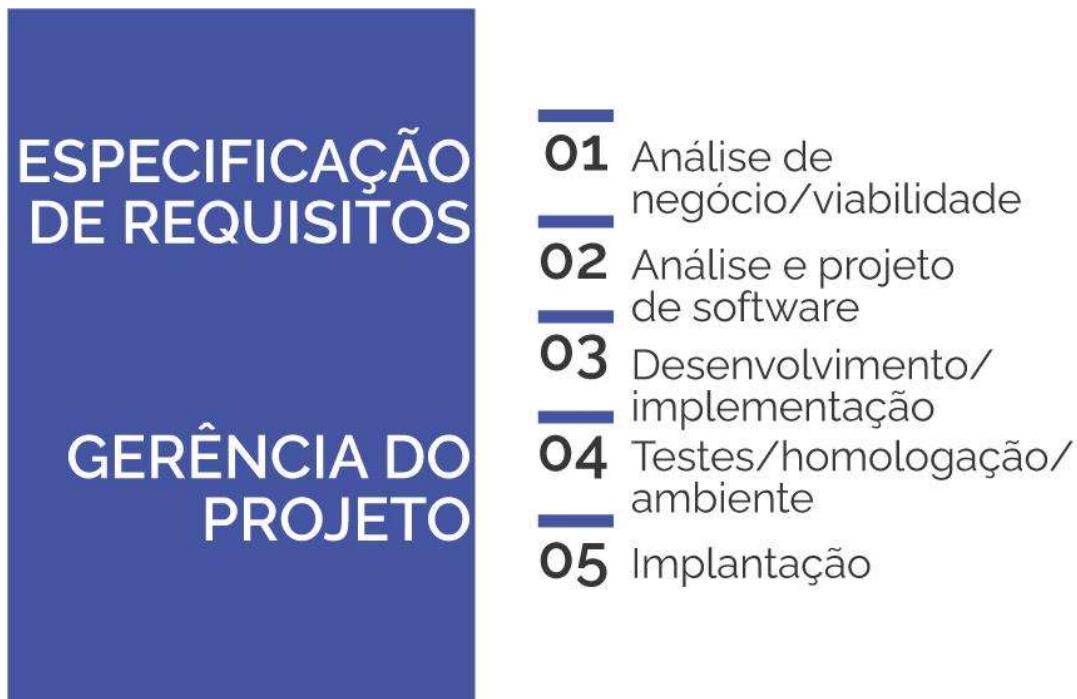
Créditos: memyjo/Adobe Stock.

Essa organização de tarefas é o ciclo de desenvolvimento do projeto, ou ciclo de vida, como vários autores citam, e há várias abordagens atualmente. Apresentaremos algumas metodologias nesse momento para exemplificar que, independentemente da maneira como a equipe trabalhará, a documentação de cada tarefa (fase), bem como a ciência dos requisitos a serem atendidos, deverão ser fortemente buscados.

É importante ressaltar que o ciclo de vida de um projeto apenas organiza as atividades, mas ainda cabe ao analista de sistemas ou líder do projeto, gerenciar cada fase, negociar com usuários, tomar decisões, motivar a equipe, entre outras responsabilidades (Yourdon, [S.d.]).



Figura 3 – Ciclo de vida de um projeto



Fonte: Rattmann, 2021.

O ciclo de vida de um software representa todas as tarefas necessárias a serem executadas para que seja concluído. Basicamente, podemos elencar as fases que a Figura 3 indica.

Cada item representa várias atividades similares que resultam em uma entrega de parte do software. A engenharia de requisitos ajuda a organizar todas essas atividades com o objetivo de produzir um software confiável e de qualidade.

Para organizar as atividades, podem ser adotadas estratégias diferentes, dependendo do grau de complexidade técnica do produto a ser produzido, da equipe disponível, do cliente, entre outros. Isto determinará se precisará de maior rigor formal no planejamento e execução de cada atividade ou se a preocupação maior é o fato do cliente ser instável e alterar os requisitos com maior frequência.

A seguir, descrevemos as estratégias mais conhecidas para organizar um ciclo de vida de projetos:

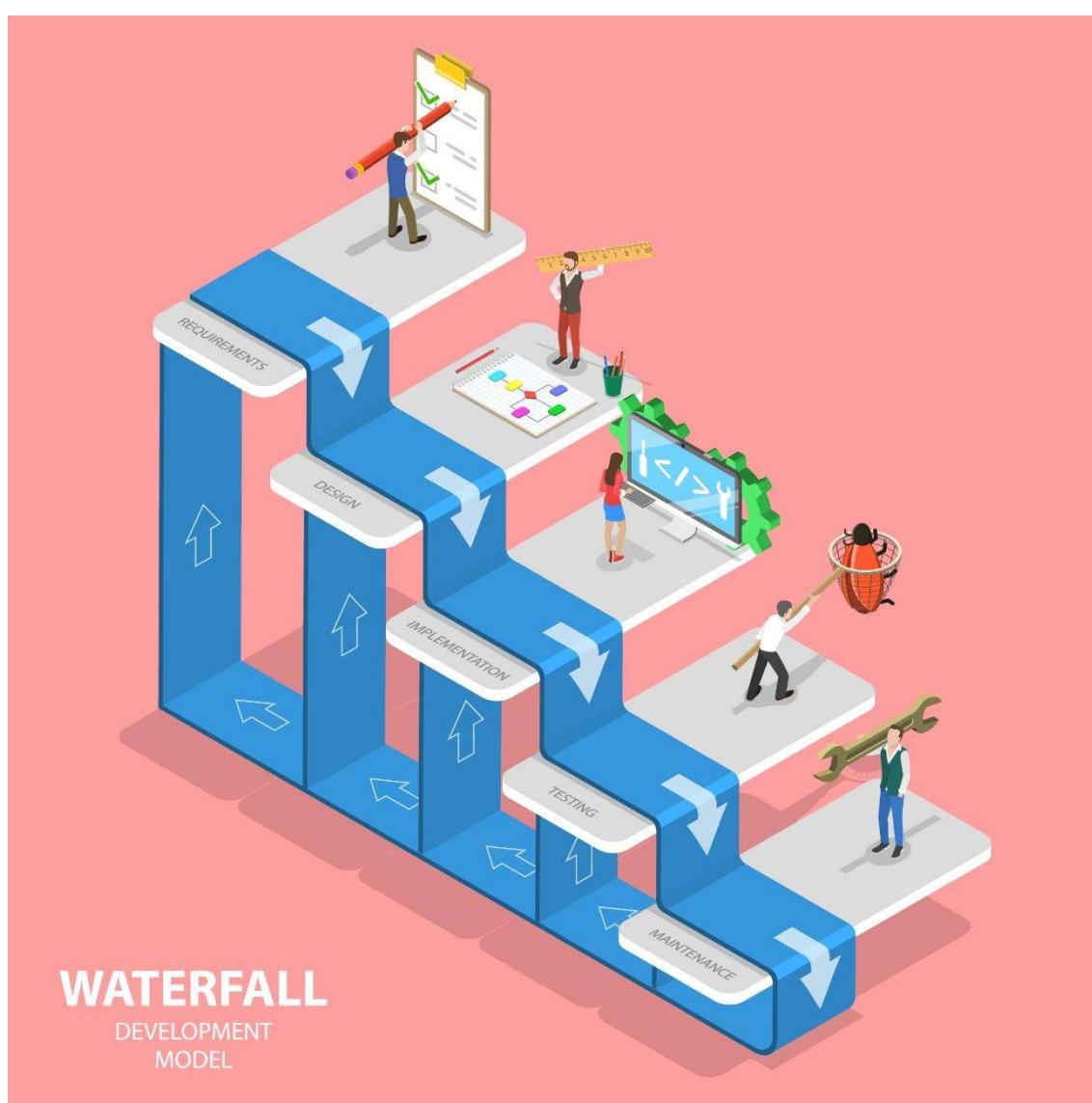
- **Estratégia sequencial:** também é conhecida por *modelo sequencial* ou *desenvolvimento em cascata*, em que as atividades são executadas sequencialmente, sempre concluindo uma antes de iniciar a seguinte. É a



forma mais tradicional de desenvolvimento de sistemas e adotado ainda hoje por várias empresas, apesar de algumas desvantagens.

É notório observar que se for descoberto um requisito mal especificado após a execução de várias atividades, provavelmente, será necessário voltar atrás para corrigir a falha. Esse tipo de abordagem de desenvolvimento do projeto é mais indicado quando se tem requisitos bem definidos e possui um grande foco no planejamento e na conferência constante dos requisitos elicitados no início do projeto.

Figura 4 – Estratégia sequencial



Créditos: TarikVision/Shutterstock.

- **Estratégia Iterativa/Incremental:** o projeto é desenvolvido por versões completas de partes do software. A cada versão, é feita uma entrega para



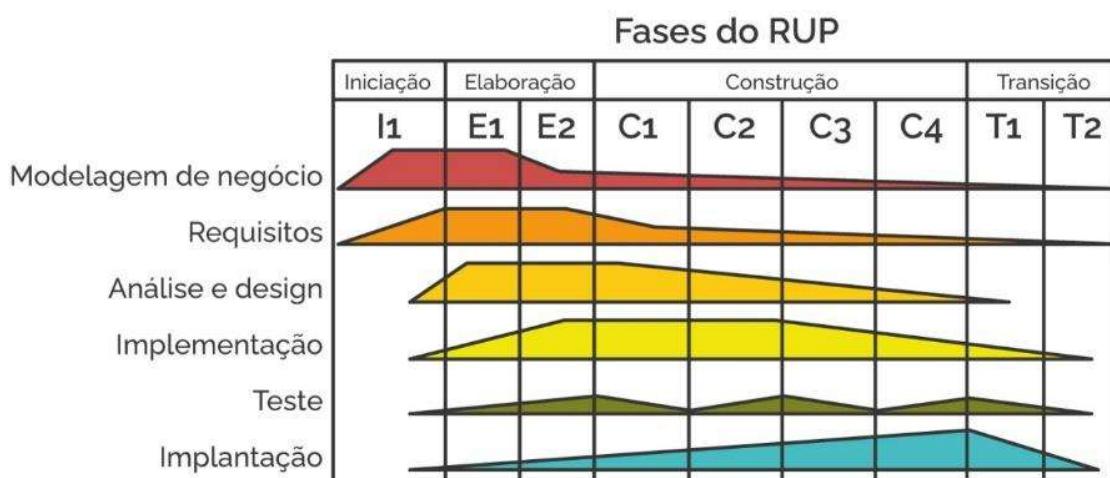
o cliente e se torna mais vantajosa por possibilitar incluir novos requisitos que não tenham sido identificados na fase inicial.

Caso essa versão apresente falhas, também é mais simples corrigi-la, sem necessariamente afetar as próximas versões. Nesse tipo de estratégia, os requisitos são organizados por funcionalidades daquela versão, por esse motivo podem ser sucessivamente refinados, garantindo maior qualidade ao produto final.

Um bom exemplo desse tipo de abordagem é o modelo RUP – *Rational Unified Process* (Processo Unificado da Rational) – que organiza o processo de desenvolvimento de software em quatro fases: Iniciação, Elaboração, Construção e Transição, que são relacionadas e tratadas durante o plano do projeto, o levantamento de requisitos, na análise e projeto, implementação (codificação), testes e implantação.

Observa-se na Figura 5 que, na fase de iniciação, há um grande esforço no levantamento de requisitos, que ocorre justamente porque este é o momento em que se busca entender as necessidades a serem atendidas para o software que será construído.

Figura 5 – Modelo Iterativo/Incremental – Exemplo RUP



Fonte: InfoEscola, [S.d.].

No entanto, as atividades sobre requisitos se mantém nas outras fases, até a Transição, quando ocorre a implantação e entrega final. O objetivo disso é melhorar o gerenciamento de mudanças de requisitos ao longo do



desenvolvimento do sistema. A cada fase e, sucessivamente, partes do produto final são concluídas e refinadas.

- **Metodologia ágil:** proposta para outros tipos de desenvolvimento que diminuam o tempo gasto em planejamento e agilidade na entrega. Muito indicado para projetos de porte menor. No começo de 2001, um grupo de 17 desenvolvedores reconhecidos se juntou em Utah, nos EUA, para discutir maneiras de desenvolvimento mais leves com base em suas experiências.

Foi criado o Manifesto Ágil que se baseou em quatro fundamentos-chave: (Foggetti).

- I. Indivíduos e interações acima de processos e ferramentas.
- II. Software funcionando acima de documentação abrangente.
- III. Colaboração com o consumidor/cliente acima de negociação de contratos.
- IV. Resposta às transformações/mudanças, mais do que seguir um plano.

Essa estratégia de desenvolvimento será explanada em detalhes nesta disciplina de engenharia de requisitos, mais adiante. A metodologia ágil tem por objetivo fazer entregas com rapidez e com maior frequência, conforme surgem as necessidades do cliente.

A cada ciclo de desenvolvimento, chamado *sprint*, todo o processo de desenvolvimento de software é realizado. Faz-se o levantamento dos requisitos para aquele trecho da solução, a análise e projeto, a implementação e testes e a entrega.



Figura 6 – Estratégia Ágil



Créditos: Elnur/adobe stock.

Na sequência, recomeça uma nova *sprint*, com todas as fases, até a conclusão de todas as sprints planejadas, como apresentar a Figura 6.

A adoção de uma ou outra estratégia não impede que se utilize outras no mesmo projeto. Aliás, para projetos de grande porte, é recomendável que se utilize mais de uma abordagem, conforme os requisitos, tempo disponível, recursos, ou seja, são muitos os fatores a se avaliar, para organizar cada etapa do projeto.

Vimos as atividades do ciclo de vida de um projeto e estratégias de organização para o desenvolvimento e, agora, entenderemos as etapas da engenharia de requisitos que ajudam em todo o processo, independentemente da estratégia utilizada.

Na Figura 7, ficam evidentes dois grupos principais: o desenvolvimento e o gerenciamento de requisitos. No desenvolvimento, estão reunidas as atividades de elicitação, análise, especificação e validação de requisitos (Kerr, [S.d.]).



Figura 7 – Etapas da engenharia de requisitos



Fonte: Elaborado com base em Kerr, 2015.

A identificação dos requisitos necessários para o software devem ser definidos em acordo como cliente, que é o principal interessado. O “cliente” pode ser uma empresa, uma equipe, ou seja, são as organizações envolvidas, aos quais são chamados de *stakeholders*. Eles influenciam os requisitos, de alguma forma, direta ou indiretamente, e são eles que precisam ser atendidos quando o projeto for concluído.

Essa fase em que se levantam informações e identificam-se necessidades chamamos de elicitação de requisitos. “Elicitação é a técnica para obtenção de dados por meio de clientes ou usuários que detenham as informações necessárias (*stakeholders*) para construir um produto” (Kerr, [S.d.]).

Há várias técnicas para auxiliar o analista de requisitos nessa etapa que serão estudadas adiante. A análise de requisitos tem como foco avaliar os possíveis conflitos, identificação das relações com o contexto, definição de requisitos. A fase de especificação de requisitos realiza a documentação e detalhamento das especificações dos requisitos.

E, por fim, a validação de requisitos compara em relação aos propósitos do produto de software. Tem a intenção de mostrar que um sistema está em conformidade com a sua especificação e que atende aos requisitos do cliente.



Observando a Figura 7, a etapa desenvolvimento contempla as demais fases de elicitação, análise, especificação e validação de requisitos continuar.

O gerenciamento de todas as etapas fazem parte do processo de desenvolvimento de software. Essa etapa tem por objetivo garantir que todas as necessidades tenham sido levantadas e devidamente elencadas na documentação de requisitos.

TEMA 2 – PAPEL DO ANALISTA DE REQUISITOS

2.1 Analista de negócios e analista de requisitos

Trabalho em equipe é a forma mais interessante para o desenvolvimento de sistemas, sejam pequenos e simples ou grandes e complexos. Os integrantes da equipe têm seus papéis definidos, conforme suas especialidades, formação e fase do projeto.

As funções mais comuns são: analista de negócio, analista de sistemas, analista de requisitos , programadores (desenvolvedores), testadores e gerente de projeto. Neste tema que estamos estudando que se refere à engenharia de requisitos, avaliaremos os analistas de negócio, de sistemas e de requisitos.

O analista de negócios trabalha com a equipe de desenvolvimento durante todo o processo de implementação e deve possuir o conhecimento funcional relevante na área em que o trabalho é atribuído. Tem a visão do todo e ajuda a organização a atingir seus objetivos. O analista de negócios também pode eliciar (levantar) requisitos e documentá-los e pode assumir o papel de analista de requisitos.

O analista de sistemas deve ter um conhecimento sólido em software de computador, hardware e rede, tendo como principais responsabilidades: interagir com usuários finais e clientes, planejar o fluxo do sistema, gerenciar as considerações de design e implementação enquanto gerencia os prazos, como sugere a Figura 8.

Também participa ativamente da fase de testes e homologação e da implantação, planejando o treinamento, configuração de todos os ambientes de produção e testes. O analista de sistemas é o principal responsável pelo desenvolvimento do projeto e responde diretamente ao gerente de projetos.



Entender quais são as funcionalidades que o sistema deve fornecer para o cliente é apenas uma parte de todo o trabalho de um analista de requisitos, nas primeiras reuniões com o cliente e todos os usuários.

O analista de requisitos tem a função de trazer as necessidades do cliente e de todas as partes interessadas. Deve ser especialista em descobrir as necessidades do cliente, para isso, tem habilidades de comunicação e conhece bem as técnicas utilizadas para negociar e interargir com os clientes e tem a capacidade de documentar os requisitos, tornar possível que os requisitos sejam elicitados.

O analista de negócios tem a visão do todo e ajuda a organização a atingir seus objetivos. Também pode eliciar requisitos e documentá-los e pode assumir o papel de analista de requisitos. O analista de requisitos tem a função de trazer as necessidades do cliente e de todas as partes interessadas, documentar os requisitos, tornar possível que os requisitos sejam elicitados.

Outros integrantes da equipe precisam conhecer os requisitos também. Os desenvolvedores codificam e constroem as funcionalidades e interfaces projetos e pensam em características baseadas nos requisitos do cliente. Os testadores geram os planos de testes com essas informações, para validar se os requisitos foram atendidos, além de validar falhas de funcionamento. O gerente do projeto precisa das informações para conferir e gerenciar possíveis mudanças nos requisitos e no escopo definido para o projeto. Enfim, é relevante que toda a equipe do projeto tenha o documento de requisitos em mãos.



Figura 8 – Analista de requisitos e analista de sistemas



Créditos: LanKogal/Shutterstock.



Créditos: PCH.Vector/Shutterstock.

O analista de requisitos conversa com o cliente (e todos os envolvidos), utiliza técnicas e habilidades de comunicação para conhecer as necessidades



para atender ao que o cliente deseja e contrata, como sugerem as responsabilidades e habilidades específicas do analista de requisitos:

- Ter facilidade para se adaptar e conhecer profundamente ao negócio do cliente.
- Levantar informações detalhadas sobre as necessidades do cliente.
- Familiaridade com aplicações e tecnologias relacionadas ao negócio.
- Interagir bem com gerentes de projetos, analistas, programadores e pessoas envolvidas.
- Boa habilidade analítica.
- Uma mente desafiadora e curiosa.
- Atenção aos detalhes e tenacidade.
- É essencial ter boa escrita e expressão para registro de não conformidades.
- Informar ao cliente sobre todos os requisitos levantadas e o andamento do projeto.

Nas empresas, pode ocorrer que o mesmo profissional acumule as funções de analista de negócios, de sistemas e de requisitos, por ter as habilidades, experiência ou especialização técnica. No entanto, há desvantagens nesta situação, além da sobrecarga de atividades, porque o analista de sistemas tem como principais atividades executar o projeto em todas as suas fases e ele deve obedecer e fazer cumprir o cronograma.

Uma vez que também exerce o papel de analista de requisitos, pode ser que acabe adiantando um pouco o tempo utilizado para a elicitação de requisitos, como meio de não atrasar a próxima fase de análise. Então, o ideal é que existam profissionais distintos que possam cooperar nas atividades e tê-las separadas e cada um com suas responsabilidades.

TEMA 3 – IMPORTÂNCIA DA ENGENHARIA DE REQUISITOS

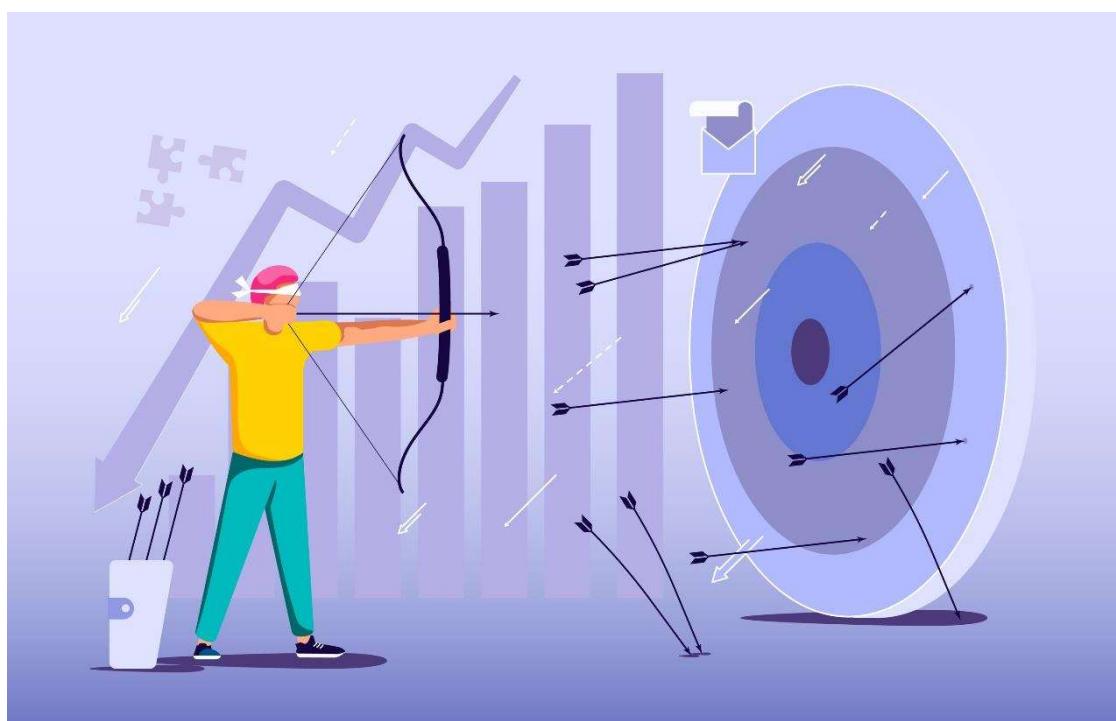
Podemos avaliar alguns sintomas de falhas no processo de desenvolvimento de software que podem ser causadas por requisitos mal elicitados, ou seja, que não são completamente atendidos em relação às necessidades do cliente. Por exemplo, quando ocorrem atrasos das entregas planejadas, insatisfação do cliente, aumento de custos, muito retrabalho, falhas



no funcionamento do software entregue, reclamações do cliente, equipe desmotivada.

"Não há nada tão inútil quanto fazer eficientemente o que não deveria ser feito" (Drucker, [S.d.]). Esse pensamento, representado na Figura 9, traduz muito bem a importância dos requisitos no desenvolvimento de um sistema. Imaginemos uma equipe com analista de negócios, de sistemas, de requisitos, um ou dois programadores e um projeto médio a ser projetado e construído em cinco meses.

Figura 9 – Conhecer as necessidades do cliente



Créditos: Tatiana Stulbo/shutterstock.

As necessidades do cliente que contratou o projeto são levantadas pelo analista de requisitos. Em seguida, é passada para a equipe, que, ao longo dos meses, trabalhou intensamente para concluir e entregar o software.

Tão importante quanto conseguir novos clientes é manter os clientes satisfeitos, para que realizem novos negócios, indique a empresa para outros, aumentando as chances de novos negócios.

No entanto, na apresentação do produto pronto, o cliente reclama que não era bem aquilo que queria e que, apesar de ter muitas funcionalidades interessantes, falta justamente uma que não existe no software entregue. Provavelmente, a equipe será cobrada para refazer algumas partes do software



e o analista de requisitos precisará entender qual funcionalidade faltou. Podem ocorrer tantas mudanças na estrutura do banco ou mudanças estruturais que inviabilizem a manutenção, por ser necessário reconstruir quase tudo.

Além do prejuízo financeiro, um cliente mal atendido pode trazer um marketing negativo irreversível. Não saber exatamente o que o cliente quer, o quê precisa e espera é fundamental para um resultado eficiente e lucrativo para uma empresa.

A engenharia de requisitos provê informações fundamentais em todas as fases de desenvolvimento do projeto, independentemente do tipo de estratégia empregada.

Buscando melhores resultados e agilidade nas entregas de seus produtos, as empresas investem em treinamentos de pessoas, em ferramentas de desenvolvimento de software e gerenciamento, buscam novas técnicas e formam boas equipes.

Voltando ao pensamento de Peter Drucker, quando se tem uma equipe bem treinada, especializada tecnicamente em softwares, em conhecimento e bem gerenciada, é possível que se produza boas soluções com rapidez. No entanto, o problema pode ser outro, ela pode produzir a coisa errada e não o que deveria ser entregue, quando não se dá a devida atenção à engenharia de requisitos, que direciona a equipe para atender as necessidades do cliente e que o satisfaça (Vazquez, [S.d.]).

TEMA 4 – IMPACTOS NEGATIVOS DA FALHA EM REQUISITOS

E qual o motivo de ainda terem retrabalho, precisarem refazer trechos enormes de códigos, atrasar cronogramas e não entregar produtos cujos clientes fiquem plenamente satisfeitos? (Vazquez, [S.d.]).

O *Project Management Institute* (PMI) apresenta uma constatação preocupante: 47% dos projetos que fracassam são causados por uma deficiência no exercício da engenharia de requisitos (PMBK). Esse estudo não é restrito a projetos de software, mas a projetos de forma geral. É possível que o resultado seja ainda pior para o contexto de projetos de software, pois, as falhas de requisitos também vão gerar retrabalho, mas isso não costuma ser tão visível.

Além de clientes insatisfeitos, as empresas responsáveis pelo desenvolvimento de projetos de software arcaram com muito prejuízo e retrabalho. Como representa a Figura 10, quanto mais tarde são descobertos requisitos



incompletos ou errados, maior o custo para recuperar o desenvolvimento do projeto.

Figura 10 – Impactos de falhas em requisitos



Créditos: small smiles/Adobe Stock.

Prejuízos financeiros chegam a valores expressivos, como apresentado no estudo de Barry Boehm (engenheiro de software americano do Centro de Engenharia de Software da Universidade do Sul da Califórnia em Los Angeles), no qual o custo para corrigir um defeito após a entrega chega a custar 100 vezes mais do que corrigir no início, durante a elaboração dos requisitos, exibido pela Figura 11 (Vazquez, [S.d.]).

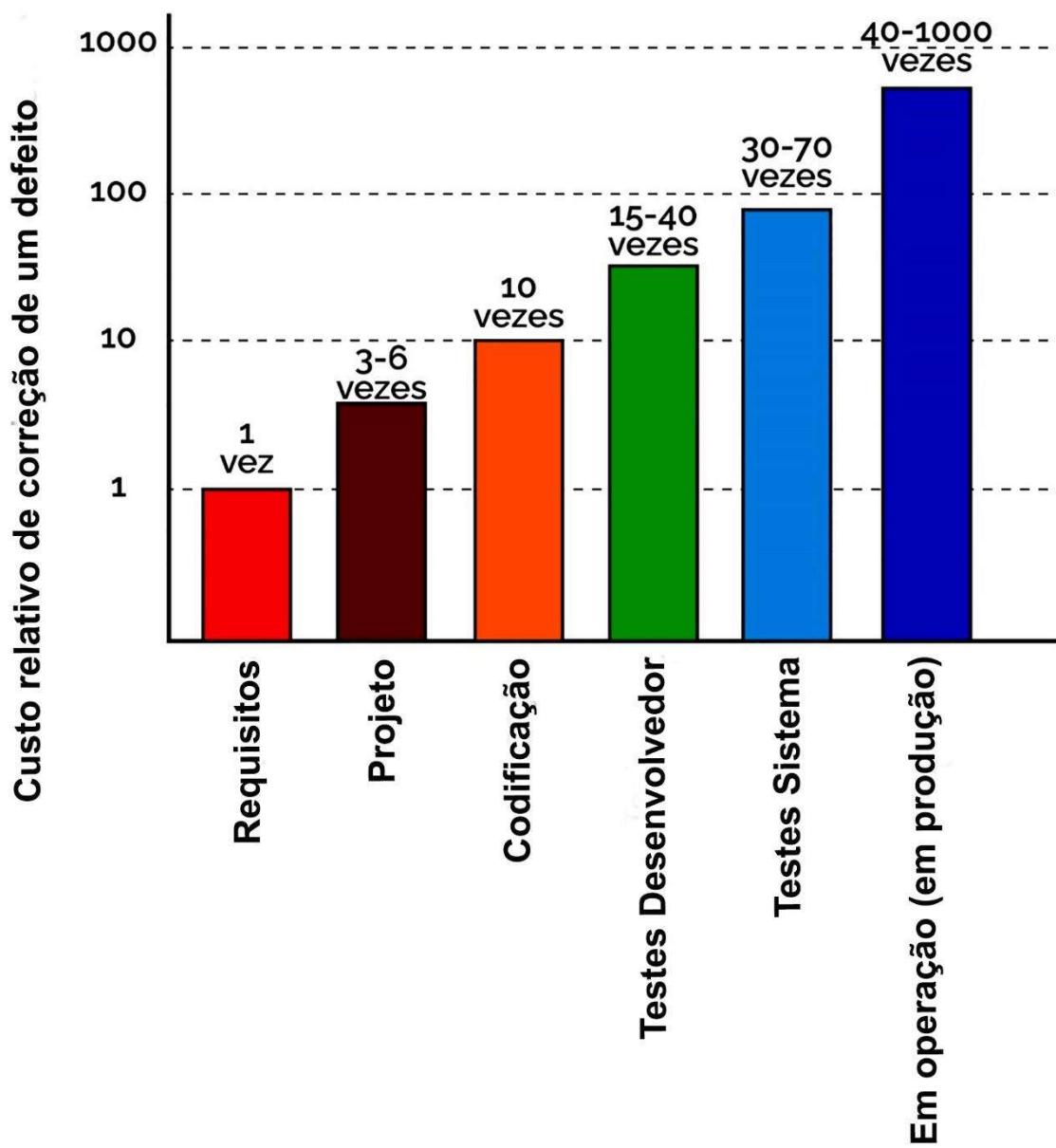
Capers Jones (2013) apresenta que, nos EUA, um em cada cinco defeitos em potencial são originados de requisitos, ou seja, 20% do total. Dean Leffingwell (1997) argumenta que, em projetos mais complexos, erros em requisitos são os mais comuns. Pohl (2011) afirma que erros em requisitos podem corresponder a até 60% do total de erros no projeto, isso considerando também como erro requisitos que faltaram ser implementados no produto. E este tipo de erro só é detectado em etapas mais avançadas do trabalho, não raro na homologação do produto.

A documentação de requisitos é bastante importante porque reúne conhecimento de negócio da empresa e, portanto, além de ser necessário uma boa especificação, deve ser mantido e entregue às equipes sempre que



mudarem as pessoas. Esses documentos minizam os impactos e perdas de informações, que também causam prejuízos.

Figura 11 – Custo relativo para correção de defeitos, conforme o ponto que é identificado por Barry Boehm



Fonte: Elaborado com base em Boehm, [S.d.].

De acordo com estudos realizados pela Escola de Gerenciamento de Cranfield no Reino Unido, 68% dos projetos são destinados ao insucesso (PKMB).

Citaremos alguns exemplos de causas para tanto insucesso:



- **Planejamento inadequado:** o projeto deve ter seu planejamento detalhado com as metas a serem atigidas e seus marcos de datas. Deve também ter definido um cronograma de atividades e quais tarefas cada integrante irá desenvolver, além de todos os recursos necessários, sejam humanos ou ferramentas e treinamentos. Grandes desvios sobre esse plano de projeto que pudessem ter sido previstos e serem mitigados caracterizam falha no planejamento.
- **Definição do projeto:** o projeto deve atender às necessidades do cliente e, quando não atende ou quando foram definidos requisitos errados ou incompletos, todo o projeto fica comprometido.
- **Definição do escopo:** alterações sobre o escopo, ou seja, sobre os itens que foram definidos como essenciais, podem atrasar o cronograma, inviabilizar a continuidade, aumentar o custo etc.
- **Alteração das especificações do projeto:** todas as alterações efetuadas, após o início das atividades, terão impacto direto em todas as outras tarefas a serem iniciadas ou na manutenção do que já foi produzido. Custos maiores, atrasos etc.
- **Inexperiência dos gestores do projeto:** o gerenciamento das atividades de todos e do plano do projeto precisam ser eficientes e atribuídas a profissional experiente, caso contrário a inabilidade ou incompetência no gerenciamento de custos e recursos mal alocados podem comprometer a conclusão do projeto e desmotivar a equipe que não produzirá tão bem.
- **Cronograma e expectativas irreais:** quando há um cronograma apertado com muitas tarefas e pouco tempo para execução, o estresse abrange toda a equipe, impactando na qualidade do que é produzido e no mal-estar entre as pessoas, o que pode gerar falhas graves e perda de produtividade.
- **Falta de suporte e envolvimento da alta gestão:** o patrocínio de gestor a um projeto é muito importante para projetos corporativos para fins de tomada de decisões que precisam ser assertivas e rápidas.
- **Falta de envolvimento do cliente:** o cliente deve ser envolvido em todo o desenvolvimento e estar satisfeito é um sinalizador para o sucesso do produto final.



Observando os efeitos dos problemas relacionados aos requisitos, percebe-se vários problemas que chegam ao desenvolvimento do sistema. Em que impacta o problema de requisitos incompletos no desenvolvimento do projetos.

No lado do cliente, piora a aceitação do software, cancela a assinatura do serviço que tinham contratado, registram notas baixas para seu sistema, podem cancelar o projeto ou não renovar o serviço.

Em relação ao produto, quando se tem requisitos incompletos, o software terá baixa qualidade, os usuários sentirão falta de funcionalidades ou de funcionamento diferente do esperado.

TEMA 5 – DIFICULDADES COMUNS COM REQUISITOS

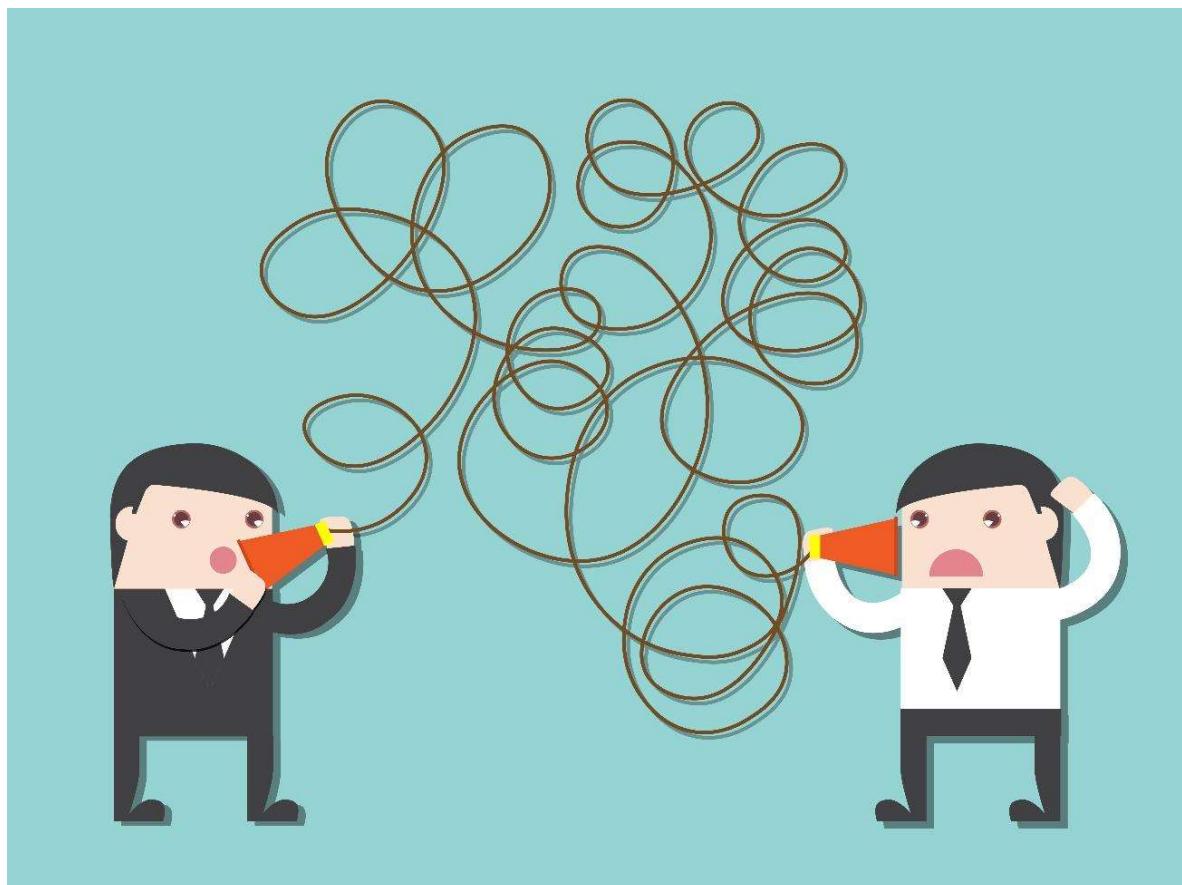
A coleta de requisitos pode parecer uma tarefa bem precisa, porém, são encontradas muitas dificuldades na prática:

- Nem sempre os requisitos são óbvios e podem vir de várias fontes (clientes).
- Nem sempre é fácil expressar os requisitos claramente em palavras e muitos clientes não sabem explicar ou não têm conhecimento de que aquela necessidade é um requisito que precisa ser informado.
- Existem diversos tipos de requisitos em diferentes níveis de detalhe.
- O crescimento desordenado do número de requisitos poderá impossibilitar o gerenciamento adequado.
- Há várias partes interessadas (*stakeholders*), o que significa que os requisitos precisam ser gerenciados por grupos de pessoas de diferentes funções.
- Os requisitos são alterados ao longo do desenvolvimento do projeto – mudam ou aumentam o escopo que foi definido no início.
- Entre outros.

Nem sempre fazer exatamente o que o cliente quer garante que ele ficará satisfeito, pois sua capacidade de se expressar é uma das maiores dificuldades. A Figura 12 demonstra uma conversa com um cliente que não sabe se expressar objetivamente. O cliente não sabe se expressar ou não sabe o que quer, e o analista de requisitos tem o papel de compreender corretamente suas necessidades, mesmo que ele não consiga informar adequadamente.



Figura 12 – Comunicação dos requisitos



Créditos: Chuenmanuse/shutterstock.

Pode-se afirmar que, em relação aos requisitos de um software, os maiores desafios no desenvolvimento estão ligados à comunicação, não a questões técnicas, como representado na Figura 13.



Figura 13 – Dificuldades de comunicação entre analistas e clientes (Fernandes)

Dificuldade	Solução
O cliente não é capaz de verbalizar o que quer	Observar os usuários a desempenhar as suas funções no contexto real
O cliente só percebe que descreveu mal o problema quando recebe uma solução que não o resolveu satisfatoriamente	Garantir que, antes de iniciar o desenvolvimento do sistema, o problema a tratar esteja bem formulado e corresponda à realidade
O engenheiro pensa saber mais do problema do cliente do que o próprio cliente	Submeter o engenheiro às dificuldades sentidas pelos usuários no contexto real
Cada parte interessada acredita ingenuamente que todas as outras estão motivadas e de boa-fé	Identificar todos aqueles que possam estar contra o desenvolvimento do sistema e encontrar formas de os motivar a contribuir positivamente

Fonte: Fernandes, 2017.

No entanto, o analista de requisitos precisa aprender sobre o negócio do cliente, obter conhecimento suficiente mesmo antes de interagir com ele. Dessa forma, poderá selecionar perguntas mais assertivas, observar se há erros nas informações ou se estão incompletas e perceber o que é óbvio pelo cliente, mas que ele não disse.

As pessoas envolvidas no projeto, no lado do cliente, são chamadas de partes interessadas ou *stakeholders* e precisam ser ouvidas para entender as suas necessidades.

As partes interessadas são pessoas ou organizações envolvidas no projeto ou cujos interesses podem ser afetados pelo projeto e, também, podem influenciar nas decisões do projeto. A não identificação de uma parte interessada pode aumentar substancialmente os custos.

Por exemplo, o reconhecimento tardio de que o departamento jurídico é uma parte interessada significativa, que gera atrasos e aumenta as despesas, devido a requisitos legais (PMKB).



Quando não se tem essas informações, há uma chance grande de não definir requisitos completos, o que pode produzir uma solução final não compatível com todas as partes interessadas, e não será útil.

As mudanças sempre acontecem e o analista de requisitos precisa estar atento e não desperdiçar a chance de incorporar ou, pelo menos, avaliar a mudança o quanto antes, evitando problemas mais para frente.

Embora dificuldades aconteçam, cabe ao analista de requisitos estar preparado com ferramentas, técnicas, habilidades e bom relacionamento interpessoal para conseguir eliciar todas as necessidades de um produto de software, que é sua principal meta.

FINALIZANDO

Nesta aula, entendemos a importância da engenharia de requisitos dentro do processo de desenvolvimento da engenharia de software, independentemente do tipo de estratégia de desenvolvimento no ciclo de vida do projeto. Pontuamos os problemas que podem acontecer quando não temos requisitos bem definidos e os documentos bem especificados, que podem impactar em tempo e custo do desenvolvimento, além de prejuízos financeiros.

Avaliamos o ciclo de vida de um projeto de software e algumas estratégias de abordagem para a condução do planejamento de todas as etapas. As especialidades dos membros da equipe definem seus papéis e tarefas, como o analista de requisitos que precisa ser um bom comunicador para obter do cliente quais são suas necessidades e, assim, repassar para o analista de sistemas, de maneira mais assertiva.

Também discutimos os impactos negativos, prejuízos, perdas e retrabalho que podem causar as falhas na elicitação de requisitos e as principais dificuldades que um analista de requisitos enfrenta ao levantar todas as possibilidades que satisfaçam ao cliente, do ponto de vista dele, para o produto a ser entregue.

Embora seja responsabilidade do analista de requisitos compreender as necessidades do cliente, mesmo que não sejam bem expressas, vimos que nem sempre entregar exatamente o que o cliente quer garante a satisfação e a qualidade do produto.



O analista de requisitos deve estar preparado com ferramentas, técnicas, habilidades e bom relacionamento interpessoal para conseguir eliciar todas as necessidades de um produto de software, que é sua principal meta.



REFERÊNCIAS

- FERNANDES, J.; MACHADO, M.; RICARDO, J. **Requisitos em Projetos de Software e de Sistemas de Informação.** 1. ed. São Paulo: Novatec, 2017.
- FOGGETTI, C. **Gestão Ágil de Projetos.** 1. ed. São Paulo: Pearson, 2014.
- KERR, E. S. **Gerenciamento de Requisitos.** 1. ed. São Paulo: Pearson, 2015.
- PMKB. **Principais Causas de Fracasso em Projetos.** Belo Horizonte: 2015. Disponível em: <<https://pmkb.com.br/artigos/principais-causas-de-fracasso-em-projetos/>>. Acesso em: 2 out. 2021.
- PRIBERAM Dicionário Brasileiro da Língua Portuguesa. Dicionário online, 2008. Disponível em: <<https://dicionario.priberam.org/>>. Acesso em: 2 out. 2021.
- VAZQUEZ, C.; SIMÕES, G. **Engenharia de Requisitos:** Software Orientado ao Negócio. 1. ed. Rio de Janeiro: Brasport, 2016.
- YOURDON, E. **Análise Estruturada Moderna.** 3. ed. Rio de Janeiro: Campus, 1990.