ENGENHARIA DE REQUISITOS

AULA 2

Profª Rosemari Pavan Rattmann

CONVERSA INICIAL

Anteriormente, vimos os motivos pelos quais a engenharia de requisitos é a primeira atividade do desenvolvimento de um projeto, no âmbito da engenharia de *software*, e a sua importância no desenvolvimento de um produto de qualidade. Nesta etapa, abordaremos os tipos de requisitos, suas características e detalharemos algumas técnicas para a sua obtenção em face dos representantes do cliente, usuários, equipe do projeto e todas aquelas pessoas ou grupos nominados *partes interessadas* ou *stakeholders*. Definiremos o que são essas partes interessadas, seu significado para os fins da engenharia de requisitos e qual sua importância para o levantamento de requisitos. A classificação dos requisitos e de suas finalidades, contextualizada com o domínio do problema e da solução, faz parte desse estudo.

Consideraremos ainda as restrições ou regras de negócio que as partes interessadas informam como base para um projeto, bem como as premissas identificadas como possíveis acontecimentos e que possam ser elencadas como medidas preventivas, para se gerenciar riscos, caso existam. Veremos a diferença entre esses fundamentos para o conceito de requisito e como o analista de requisitos deve trabalhar com tais informações. A análise de riscos não faz parte da etapa de elicitação de requisitos, porém o analista de requisitos deve trabalhar junto com o analista de sistemas.

Veremos, por fim, que requisitos de alta qualidade devem ser claros, completos, sem ambiguidade, consistentes e testáveis; por isso, uma boa elicitação e especificação documental é muito importante para o resultado final da engenharia de requisitos. Ainda sobre a elicitação de requisitos, avaliaremos um conjunto de técnicas empregadas para levantar, detalhar, documentar e validar os requisitos de um projeto, sempre focando no escopo definido. O escopo do problema define exatamente as fronteiras sobre o que será implementado no produto e o que não será implementado. Concluiremos com uma explanação sobre o que é o escopo do projeto, lembrando sempre que uma boa engenharia de requisitos é um passo fundamental para um projeto de *software* de qualidade.

TEMA 1 – DEFINIÇÃO DO ESCOPO: DOMÍNIO DO PROBLEMA E REQUISITOS DE NEGÓCIO

Segundo o dicionário *Michaelis*, a definição de escopo é: “1 Ponto de mira; alvo. 2 Algo que se pretende conseguir ou atingir; intenção, objetivo [...]. 3 Espaço que circunda ou envolve; âmbito.” (Escopo, 2015).

Figura 1 – Escopo: o alvo a ser atingido, em um projeto

Créditos: Visual Generation/Adobe Stock.

Segundo o guia *Project Management Body of Knowledge* (Pmbok), a declaração do escopo do projeto é o documento final sobre o que será entregue, que detalha prazos, custos, atividades e proporciona um entendimento comum entre o cliente e a equipe de desenvolvimento, a respeito do projeto (PMI, 2018). A Figura 1 retrata que, após ouvir todas as partes interessadas, o alvo precisa ser definido, o escopo precisa ser descrito e aceito. A declaração do escopo do projeto deve garantir que o projeto contemple todo o trabalho necessário, e apenas o necessário, para aquele terminar com sucesso. O escopo deve explicitar o que será feito e o que não constará no projeto, como um contrato com o cliente, de forma a não restar dúvidas, registrando um entendimento comum, das partes interessadas, sobre o projeto. Embora pareça tratar de informações redundantes, a elaboração do escopo é um processo necessário para explicitar a percepção do que o cliente espera do produto e do que a equipe de desenvolvimento deve projetar e implementar.

O escopo positivo refere-se a tudo que o projeto deve atender e entregar como produto final e deve descrever todos os produtos de um projeto, os serviços necessários para realizá-los e os resultados finais esperados. O escopo negativo identifica e deixa claro tudo aquilo que o projeto não irá atender ou não será entregue no produto final. É bastante importante explicitar as ressalvas sobre o que o projeto não irá contemplar, deixá-lo claro para as partes envolvidas. Com isso, em última análise, o escopo como um todo relata os produtos que serão desenvolvidos e os que não serão desenvolvidos, em um projeto. A omissão de ressalvas pode significar que está implícita a realização de serviços não citados no contrato, o que nem sempre se aplica (Castro et al., 2015).

O domínio do problema é uma área em análise (Vazquez; Simões, 2016). Corresponde às fronteiras (decisões, resoluções, políticas) em que o *software* estará imerso, com que terá que interagir. Em uma empresa, as decisões são tomadas por quem tem autoridade e responsabilidade de estabelecer o domínio do problema e os requisitos ou necessidades de negócio. O domínio implica as metas da organização, descreve as razões pelas quais o projeto foi iniciado, os problemas a se resolver com ele.

Uma boa declaração de escopo deve:

- descrever o escopo do *software* a ser desenvolvido;
- descrever as características do produto contratado e em acordo com o cliente e a equipe de desenvolvimento, referenciando a especificação de requisitos;
- definir os critérios para aceitação do projeto;
- descrever como serão efetuadas as verificações de atendimento aos requisitos e critérios de aceitação;
- estabelecer o cronograma de entregas do *software*;
- detalhar todas as entregas a serem feitas, mesmo que parciais, quais relatórios, documentos devem ser elaborados, com as respectivas datas e cujo nível de detalhe depende do tamanho do *software* ou da sua complexidade;
- descrever o escopo negativo, identificando, de forma clara, o que não será implementado, declarando explicitamente o que está fora do escopo do projeto, ajudando no gerenciamento das expectativas das partes interessadas;

- descrever as restrições do projeto, que limitem as opções da equipe, incorporando cláusulas contratuais, se for o caso;
- descrever as premissas específicas ao projeto, com seus riscos e seu impacto potencial, se aquelas forem provadas falsas.

O escopo compreende, assim, todo o trabalho necessário para desenvolver o projeto e reunir informações relevantes para isso, como os objetivos específicos, as entregas de produtos e em que datas, quais tarefas serão necessárias, as responsabilidades de cada integrante do projeto, prazos e custos. Quanto melhor e mais completa a definição do escopo, menos problemas e conflitos tendem a ocorrer entre as partes interessadas e a equipe de desenvolvimento do projeto.

TEMA 2 – RESTRIÇÕES E PREMISSAS

2.1 Restrições

Restrições são limitações às possíveis soluções de desenvolvimento do *software*. Além disso, elas afetam o projeto em termos de implementação, testes, validação, até sua implantação. Não são requisitos, mas geram impactos no desenvolvimento. As restrições não são negociadas e não podem ser modificadas, apenas pensadas, identificadas, tratadas e validadas (Figura 2).

Figura 2 – Restrições, que devem ser validadas

Créditos: BRO.vector/Shutterstock.

Um exemplo de restrição é um prazo para que o *software* fique pronto, por imposição legal. Pode ser que essa restrição propicie uma solução mais rápida, porém não a ideal, porque essa última precisaria de mais tempo. Cabe ao analista de requisitos avaliar e validar essa restrição como verdadeira e considerá-la no plano do projeto.

Além de restrições de negócio, podem existir várias restrições associadas à tecnologia, como decisões sobre a arquitetura do *software*, uso de servidores, capacidade de memória e de processadores, componentes, entre outros elementos. Essas informações são muito necessárias para identificação e avaliação da solução a ser desenvolvida, evitando-se o risco de se criar uma solução inadequada.

2.2 Premissas

O analista de requisitos ouve todas as partes interessadas sobre os requisitos e pressupõe muitas situações, gerando indagações e premissas. Essas são suposições que devem ser validadas e confirmadas para se seguir com a especificação dos requisitos. Segundo o guia Pmbok, “Premissas são fatores associados ao escopo do projeto que, para fins de planejamento, são

assumidos como verdadeiros, reais e certos sem a necessidade de prova ou demonstração" (PMI, 2018).

Um exemplo de premissa é o uso do número no Cadastro de Pessoas Físicas (CPF) como chave de *login* no sistema, porque cada usuário do *software* possui CPF único e diferente um do outro. Se, no decorrer do desenvolvimento, for descoberto que nem todos têm CPF (por exemplo, menores de idade que utilizarão o *software*), muitas funcionalidades terão que ser refeitas para se fornecer um novo *login* que seja adequado a uma chave diferente do CPF.

A seguir, falaremos um pouco sobre riscos, pois premissas podem configurar riscos ao projeto. A Figura 3 simboliza a necessidade de se investigar e planejar todos os fatores que possam indicar necessidades de uma solução, mesmo que ainda não declaradas. Assim, podemos dizer que as premissas devem ser tratadas como riscos ao projeto e, portanto, devem ser gerenciadas e documentadas.

Figura 3 – Premissas, que podem configurar riscos ao projeto

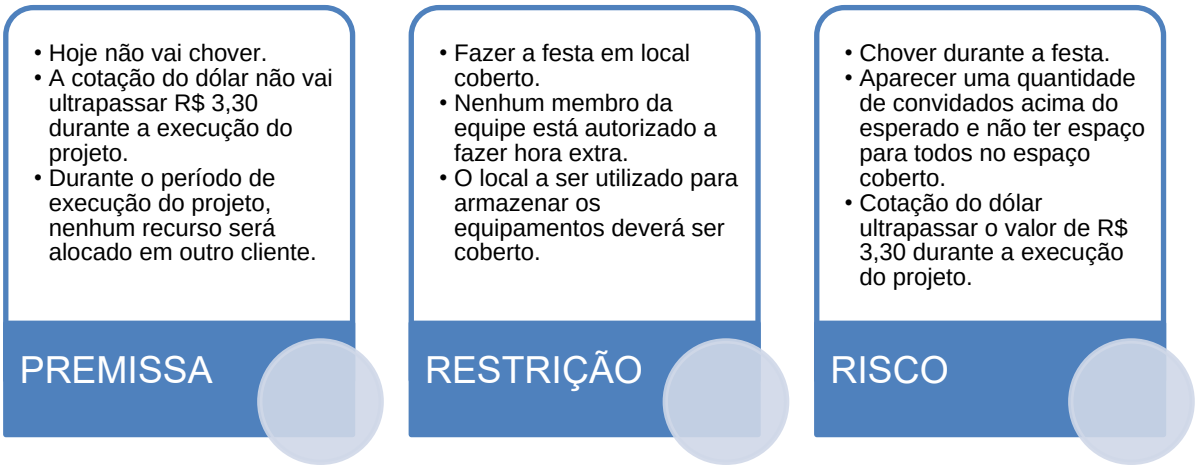
Crédito: VectorMine/Shutterstock.

2.3 Riscos

Quando o analista de requisitos observa e identifica situações que possam impactar os requisitos do projeto, como as premissas que avaliamos anteriormente, é necessário listar os riscos possíveis, ou melhor, os eventos incertos cuja ocorrência possa provocar efeitos negativos no projeto. Evitar que

problemas aconteçam é praticamente impossível; no entanto, identificar possíveis riscos, ajudar a gerenciá-los de forma antecipada e minimizar maiores problemas podem ser muito úteis. Uma avaliação de riscos determina a probabilidade de ocorrência e o grau do impacto que podem causar no desenvolvimento do projeto, o que permite que o analista de sistemas e sua equipe planejem ações preventivas para reagir a esses problemas, quando e se ocorrerem.

Figura 4 – Diferenças entre premissas, restrições e riscos



TEMA 3 – PARTES INTERESSADAS (*STAKEHOLDERS*)

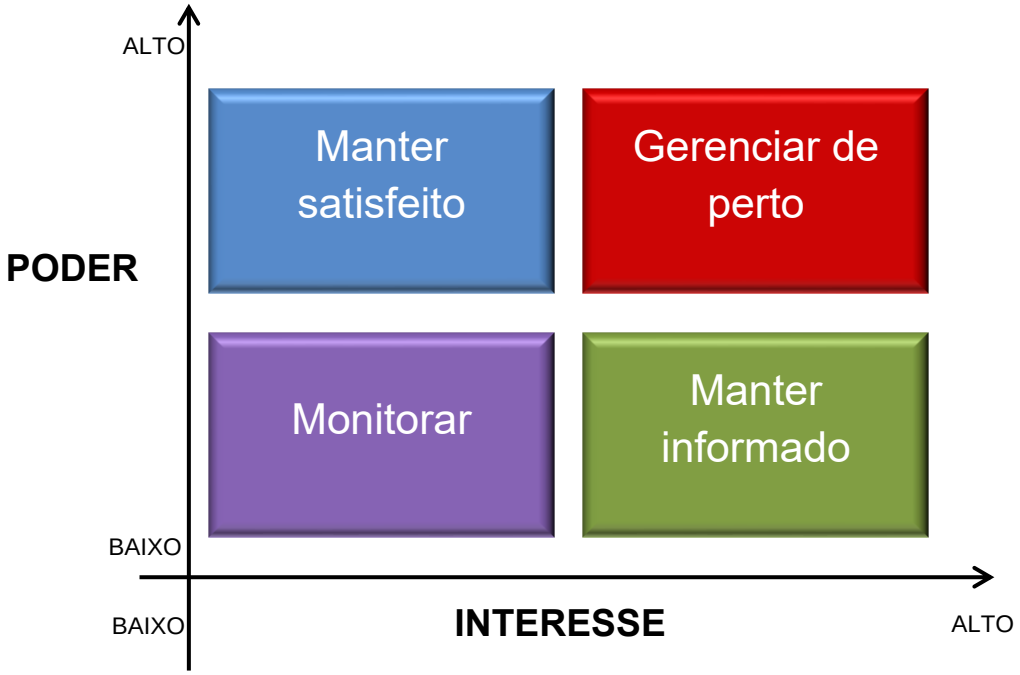
Utilizaremos o termo *stakeholders* para nos referirmos às partes interessadas de um projeto. O trecho a seguir explica como esse termo oriundo do inglês foi introduzido na engenharia de requisitos.

> O termo em inglês surgiu nos anos sessenta da derivação de “stake” + “holder” (tradução livre: aquele que possui estava/delimitador) e da palavra stakeholder (acionista), com a intenção de classificar os proprietários de ações que podiam influenciar a tomada de decisões em uma organização. Com o passar dos anos o termo foi incorporando conceitos mais amplos e difundido na área de estudo da administração de negócios. Na Engenharia de Requisitos o termo foi adotado para determinar aqueles que podem possuir outros tipos de participação que não sejam apenas cliente e usuário de um software, ou seja, o stakeholder influencia os requisitos de um sistema e é afetado por esse sistema. (Ramos, 2019)

Genericamente, a designação que passou a ser adotada foi *partes interessadas*. No entanto, segundo Ramos (2019), nem sempre o *stakeholder* é uma parte que está efetivamente interessada no desenvolvimento da solução ou na mudança que isso causará. O nível de interesse, bem como o grau de

influência (poder) na organização, são atributos dos *stakeholders* que precisam ser considerados e gerenciados durante o desenvolvimento do projeto, conforme mostrado na Figura 5.

Figura 5 – *Stakeholders*: sua influência e poder



3.1 Cliente *vs.* usuário

Cliente e usuário são casos específicos de partes interessadas. Cliente é quem solicitou o serviço e, geralmente, quem paga pelo trabalho e espera ser atendido plenamente em todas as suas necessidades, no prazo mais curto e com custo reduzido. Usuário é aquele que usará o produto desenvolvido e, efetivamente, realizará o trabalho da organização (cujo cliente é o **dono**) e que, por isso mesmo, espera que o produto entregue lhe seja útil, fácil de usar e execute todas as operações que o usuário precisa para realizar um trabalho eficiente, rápido e correto.

No entanto, como relatam Vazquez e Simões (2016), na maioria dos projetos, uma boa parte dos requisitos será obtida das partes interessadas que não são clientes ou usuários. Assim, consistem em alguns exemplos de *stakeholders*, em um projeto:

- Acionistas
- Colaboradores
- Gestores da organização

- Clientes
- Concorrentes
- Fornecedores
- Governos
- Mídia

O nível de interesse e o grau de poder afetam como as partes interessadas se comunicam e interagem com a nova solução de *software* a ser entregue. Por exemplo, em uma empresa cujos funcionários realizam suas funções com base em processos manuais, esses provavelmente ficarão preocupados com a possibilidade de perder seus postos de trabalho, caso a proposta de desenvolvimento de um sistema (*software*) possa causar sua substituição. Esses funcionários são considerados *stakeholders* porque seu conhecimento e experiência identificarão muitos requisitos; no entanto, eles podem ser pessoas não interessadas na efetividade dessa solução. Cabe ao analista de requisitos, com todas as suas habilidades desenvolvidas, conversar com essas pessoas e conseguir coletar delas todas as informações necessárias.

Os diferentes tipos de leitores dos requisitos são os *stakeholders* (partes interessadas) do software. A identificação desses *stakeholders* pode ser difícil e pode ser alterada ao longo do ciclo de vida de um projeto; porém, o entendimento de seu grau relativo de influência, e se negativa ou positiva, em um projeto é um ponto crítico. Alguns se beneficiam de um projeto bem-sucedido, enquanto outros enxergam que o sucesso de um projeto pode lhes causar resultados negativos. Negligenciar a existência e o papel dos *stakeholders* negativos pode aumentar a probabilidade de falha do projeto.

O documento com a especificação dos requisitos deve ser escrito de forma clara e objetiva, sem muitos termos de tecnologia nem explicações demasiadas sobre o processo de implementação do *software*, para que o cliente o entenda. Segundo Vazquez e Simões (2016), diversos autores que analisam documentos de projetos de desenvolvimento de *software* concluíram que um erro muito comum é a elaboração da especificação de requisitos com uma linguagem adequada apenas à equipe de desenvolvimento. Ou seja: ainda que esse documento útil possa ser útil para os programadores de código, seu objetivo é que o cliente o entenda e consiga, por ele, avaliar se suas necessidades se encontram contempladas no projeto.

TEMA 4 – REQUISITOS

4.1 Definição de requisitos

Na fase inicial de um projeto, o seu foco está no levantamento, ou seja, na compreensão do que o produto de *software* deve ser capaz de realizar para que satisfaça as necessidades de quem o solicitou. A Figura 6 exemplifica o objetivo da etapa de levantamento de requisitos. É um olhar à frente, buscando o máximo de informações, mesmo aquelas que ainda não tenham sido elencadas. Além disso, é fundamental entender a natureza do *software* a ser construído, o domínio do problema e o comportamento esperado para o sistema.

Figura 6 – A definição de requisitos

Créditos: Rudall30/Shutterstock.

Tendo todos os requisitos elicitados, outras atividades da engenharia de requisitos serão iniciadas, como a modelagem, a avaliação de cada requisito e, por fim, a documentação dos requisitos. Uma parte vital dessa fase é a elaboração de modelos conceituais que descrevam o que o *software* tem de

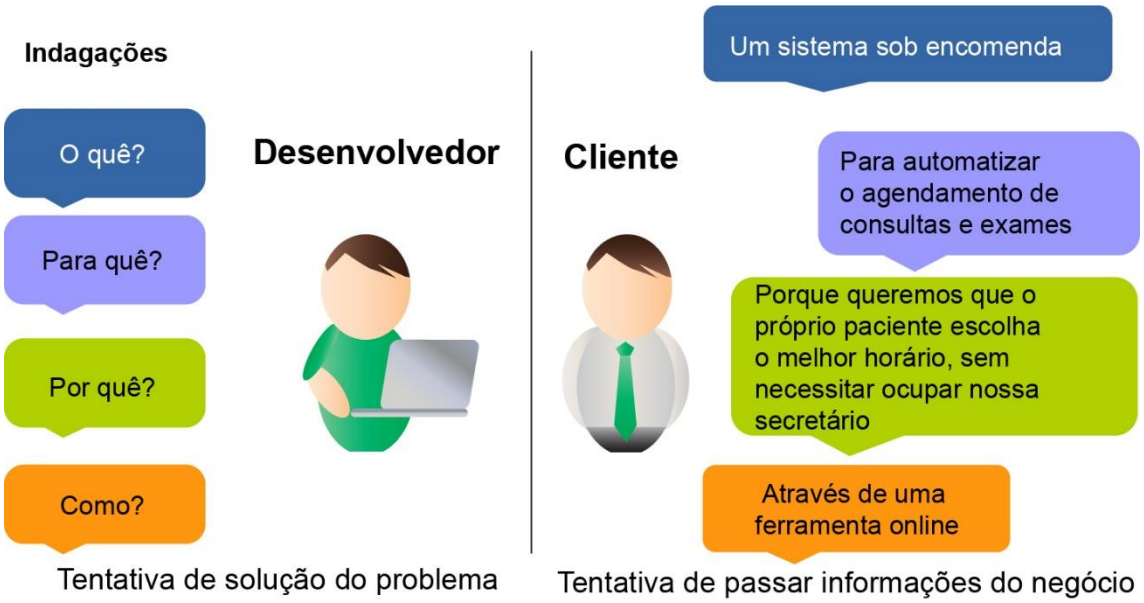
fazer. Esses modelos são muito úteis para o completo entendimento e validação do que foi levantado.

Uma das principais medidas do sucesso de um *software* é o grau do quanto ele atende aos requisitos para os quais foi construído. Os requisitos são as descrições das necessidades e propósitos do cliente de um sistema. Então, qual a definição de requisito? O Instituto de Engenheiros Eletricistas e Eletrônicos (IEEE) – uma organização profissional sem fins lucrativos, fundada nos Estados Unidos, que consiste na maior organização profissional do mundo dedicada ao avanço da tecnologia em benefício da humanidade –, define requisito como:

> – Uma condição ou capacidade do sistema, solicitada por um usuário, para resolver um problema o atingir um objetivo;
> – Uma condição ou capacidade que deve ser atendida por uma solução para satisfazer um contrato, especificação, padrão ou quaisquer outros documentos formais impostos.
> – Documentação na representação das condições ou capacidades apresentadas nos dois itens anteriores.
> – Uma condição ou capacidade que deve ser alcançada ou possuída por um sistema, produto, serviço, resultado ou componente para satisfazer um contrato, padrão, especificação ou outro documento formalmente imposto. Requisitos incluem as necessidades quantificadas e documentadas, desejos e expectativas do patrocinador, clientes e outras parte interessadas.

Quando uma empresa ou equipe de desenvolvimento de *software* está responsável por um pedido do cliente, precisa que o analista de requisitos entregue o documento com todos os requisitos especificados para que a equipe os entenda e possa iniciar as atividades de análise e o projeto da solução. Vários são os questionamentos a se fazer até se conseguir entender todas as necessidades de todas as partes interessadas, como exemplifica a Figura 7.

Figura 7 – Indagações sobre a solução de um problema



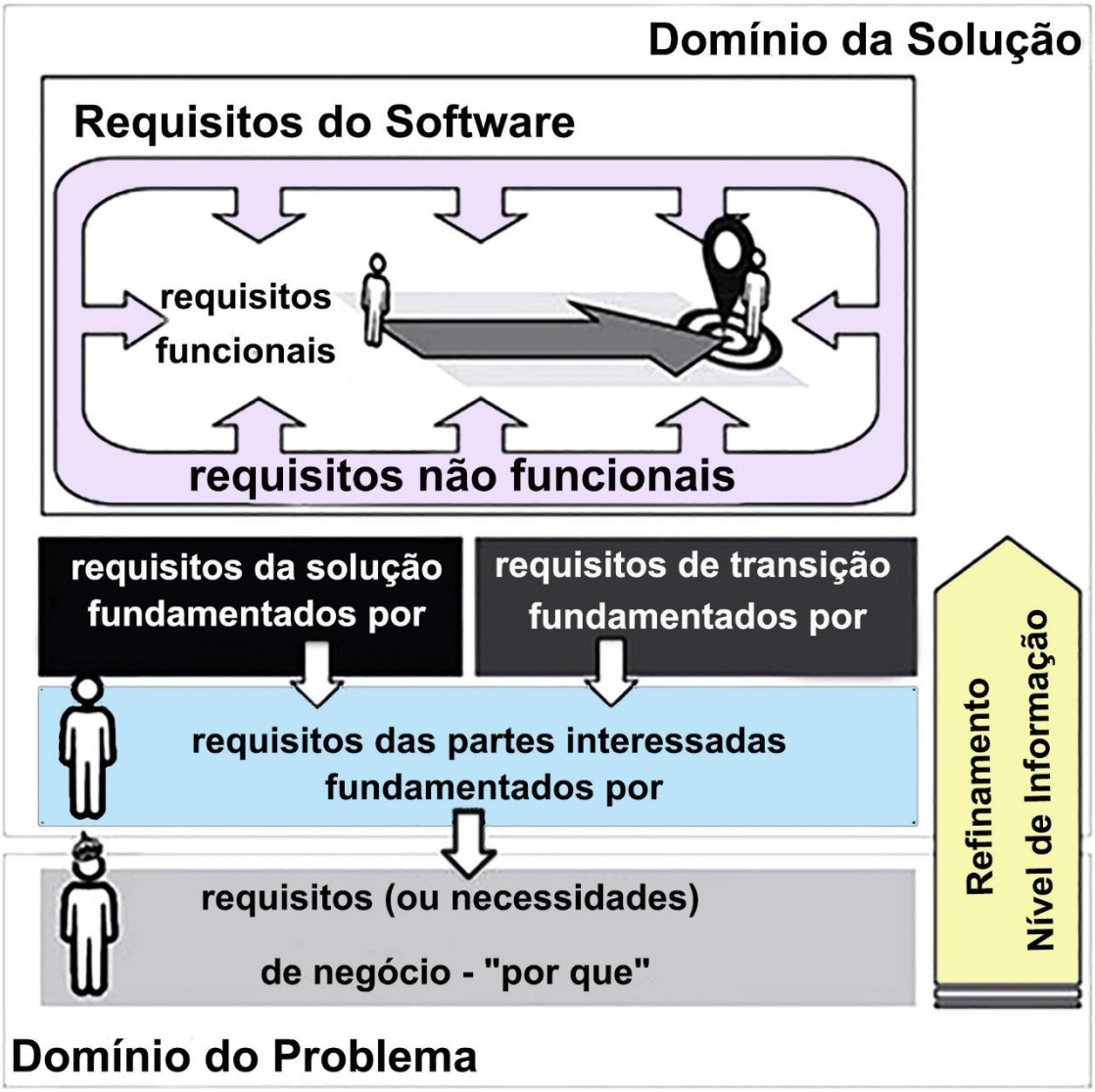
Fonte: S.Gomes/SlideShare.

A definição dos requisitos deve ser suficientemente abstrata, para que não seja direcionada para uma solução específica, antes de se avaliar o todo. Por exemplo: se o detalhamento dos requisitos for no nível de linguagem de programação, a equipe acabará tendo que utilizar a linguagem A ou B, sem, no entanto, ter avaliado se é a mais indicada para o caso. Muitas são as condições para a decisão de qual tecnologia se utilizar, e não é no momento de definição de requisitos que isso deve ser pensado. O motivo disso é que se faz necessário entender centralmente as reais necessidades do cliente e como deverá ser o comportamento do *software* para atendê-las, não como ele será construído.

4.2 Tipos de requisitos

Há várias classificações de requisitos que utilizam a hierarquia que detalharemos a seguir, resumida na Figura 8.

Figura 8 – Tipos de requisitos



Fonte: Elaborado por Leitão, 2022 com base em Sommerville, 2018.

4.2.1 Requisitos de negócio

Requisito de negócio, também conhecido como *regra de negócio*, é o que define a forma de fazer o negócio do cliente, reflete a sua política interna, os seus processos existentes e que não serão alterados. Compreende ainda as normativas da empresa e do cliente a que os seus funcionários (futuros usuários do *software*) já obedecem e que o *software* (sistema) a ser desenvolvido deve contemplar. Uma regra de negócio não se torna uma funcionalidade do *software*, necessariamente, mas guia o comportamento das suas funcionalidades. Sistemas não existem sem regras de negócio e nem todas as regras de negócio são automatizadas, em um sistema. Um sistema sem regras de negócio permitirá

muitas coisas e sem qualquer critério. Por outro lado, existem regras de negócio não automatizáveis por um *software*, por exemplo, um funcionário pegar café no escritório sempre que ficar com sono após o almoço. De todo modo, egras de negócio sempre existirão para organizar os processos das empresas.

São exemplos de requisitos de negócio regras expressas como:

- *Para se conceder limite de cheque especial, o indivíduo deve ser cliente do banco há mais de um ano, nunca ter emitido cheques sem fundo e sobre ele não constarem restrições nos sistemas de crédito.*
- *Somente serão aceitos pagamentos em espécie. Não serão aceitos cartões de crédito, cartões de débito ou cheques.*
- *Somente serão aceitos pedidos com datas do mês anterior.*

4.2.2 Requisitos das partes interessadas

Requisitos das partes interessadas são as necessidades e expectativas das partes interessadas em relação à organização, que devem ser avaliadas, monitoradas e atendidas. Com base nos requisitos das partes interessadas, surgem oportunidades para identificar conflitos a serem resolvidos e oportunidades de consolidar requisitos semelhantes.

Vejamos o exemplo de requisito das partes interessadas sugerido pela ISO 9001 (ABNT, 2015): numa **festa de casamento**, as expectativas são diferentes para cada envolvido. Nesse caso, os noivos seriam a organização e os *stakeholders* seriam os pais dos noivos e seus parentes, amigos e fornecedores. Entre esses *stakeholders* da festa de casamento,

- os pais dos noivos esperam que todos os parentes do casal sejam convidados;
- os parentes e amigos dos noivos esperam que a festa supere as suas expectativas quanto à qualidade de música, ambiente e alimentação.
- os fornecedores da organização do evento, como a empresa de serviço de *buffet*, o *disk jockey* (DJ) ou os músicos, a empresa de decoração, certamente querem que os pagamentos sejam quitados antes de se iniciar a festa.

Identificar quem são e o que esperam as partes interessadas é uma tarefa primordial para se entender o contexto da organização, as suas necessidades e expectativas e como o não atendimento dos seus interesses afetará a

organização na busca de prover produtos e serviços conformes ao mercado em que aquela atua.

É importante observar que devemos definir os requisitos pertinentes, ou seja, os requisitos que estejam de acordo com o sistema de gestão da qualidade da organização (empresa). Como estamos falando de uma norma de qualidade (ABNT, 2015), as partes interessadas serão aquelas que influenciem ou sejam influenciadas pela gestão e pela política de qualidade. A melhor maneira de defini-las é fazer um levantamento para saber quem são aqueles que afetam o sistema de gestão da qualidade. A Figura 8 lista esses níveis de requisitos.

Figura 9 – Níveis de requisitos

PARTES INTERESSADAS	REQUISITOS
Corretores	Infraestrutura de atendimento ao cliente. Novas oportunidades criadas.
Acionistas	Retorno sobre o investimento. Redução de prejuízos.
Construtoras	Parceria em novos negócios. Prazos reduzidos
Cliente Presidente da Empresa	Gerenciar toda a produção da empresa. Relatórios de cada setor.
Cliente Funcionário – opera o sistema	Software forneça todas as informações e seja fácil de usar. Integre com outros setores da empresa.
Gerente do Projeto	Todos os prazos cumpridos. Custo dentro do esperado. Cliente satisfeito

São exemplos de requisitos das partes interessadas:

a. Os funcionários da empresa aguardam que seus salários sejam pagos nos prazos previstos.
b. Os órgãos reguladores preveem que a companhia cumpra a legislação vigente.
c. A prefeitura da cidade onde a empresa está sediada tem a expectativa de que as contribuições e impostos de âmbito municipal sejam recolhidos pela empresa na data e na forma corretas.
d. As associações de classe/sindicatos cobram que os direitos dos trabalhadores sejam atendidos integralmente.

4.2.3 Requisitos de transição

Os requisitos de transição são um conjunto de requisitos temporários, importantes para a implantação da solução, mas necessários somente para que ela seja possível. Quando um *software* é implementado para substituir um outro, é necessário pensar em tudo que será demandado para que o processo todo aconteça sem causar impactos para a empresa. Pode ser preciso executar processos em paralelo, até que tudo que esteja determinado para tal seja substituído e apenas o novo *software* reste em funcionamento. Esses requisitos somente podem ser previstos quando já se tem o projeto do novo *software*. Além disso, são relevantes somente durante o período de transição entre as soluções. No entanto, para elicitar esses tipos de requisitos, o trabalho é o mesmo dos demais requisitos – a diferença não está nos métodos para defini-los, mas nas entradas e no fato de que eles deixam de ser relevantes assim que o *software* existente for desativado.

Geralmente, esses requisitos envolvem tarefas como conversão de informações (transmissão de dados de um sistema antigo para um sistema novo), treinamentos para que a nova solução possa ser operada e capacidades como redundâncias e trabalhos paralelos. Requisitos de transição podem ser comparados com andaimes de uma obra: eles precisam existir para que as paredes sejam construídas, mas depois são descartados por não terem mais utilidade.

São exemplos de requisitos de solução e transição:

a. recrutamentos e qualificações;
b. mudanças de setor;
c. migração de dados entre servidores;
d. condução de treinamentos para usuários, no novo *software*;
e. migração de dados de uma base de dados para outra;
f. *backup* de códigos e dados.

4.2.4 Requisitos da solução

Os requisitos da solução descrevem as características de uma solução que atenda aos requisitos do negócio e aos requisitos das partes interessadas, capturando as decisões tomadas sobre o escopo e o comportamento esperado para o *software*. Tanto os requisitos da solução como os requisitos de transição

se subdividem em **requisitos funcionais** e **não funcionais**. Os requisitos funcionais descrevem o funcionamento da solução, o seu comportamento e as informações que ela irá gerenciar; os requisitos não funcionais, conhecidos como *requisitos de qualidade* ou *suplementares*, descrevem condições como eficiência, velocidade, disponibilidade, aparência e ambiente, sob as quais a solução irá operar. A Figura 10 apresenta exemplos de requisitos de solução.

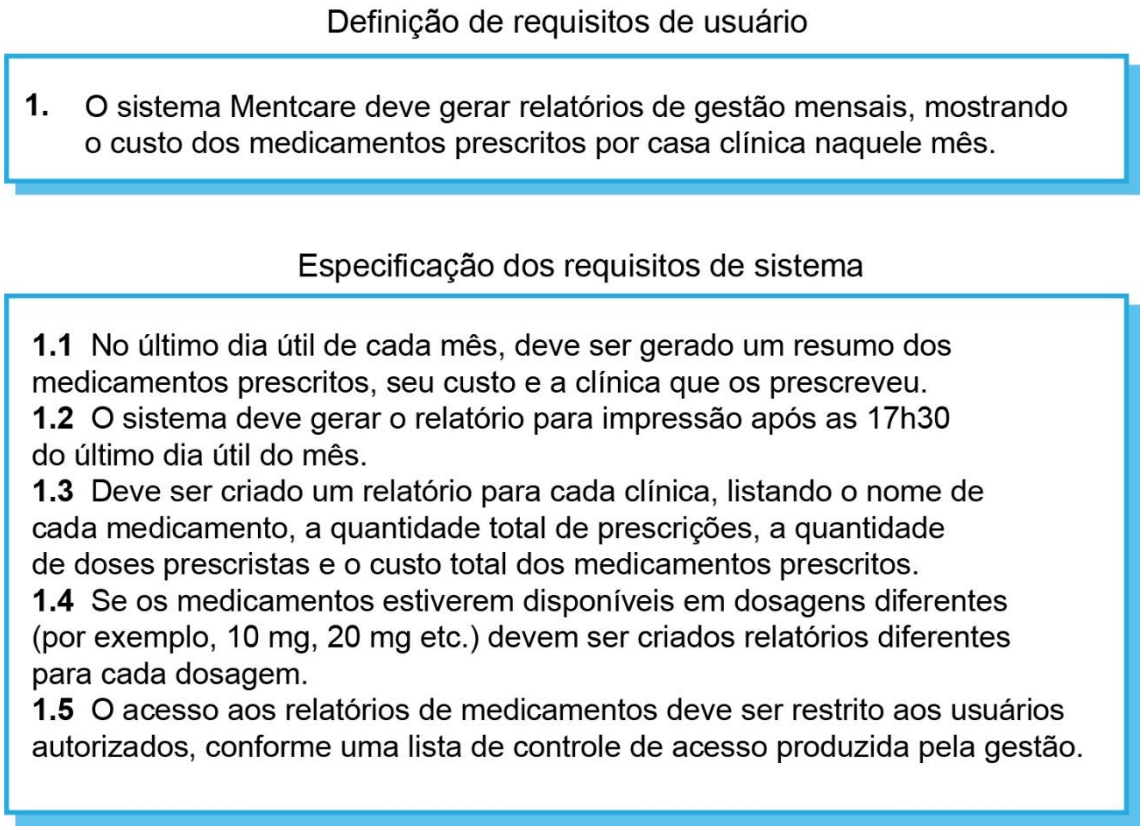
Figura 10 – Exemplos de requisitos de solução

Item / Descrição da Característica
1.1.1 A SOLUÇÃO deve possuir um único ambiente de administração e controle de acessos de usuários.
1.1.2 A SOLUÇÃO a ser ofertada deve possuir módulos componentes integrados de forma única e nativa entre si, ou seja, a sua integração deve ser provida em suas versões originais integradas e possuir as mesmas características tecnológicas. Não será admitida SOLUÇÃO COMPOSTA por módulos de diferentes empresas detentoras de direitos autorais e de comercialização.
1.1.3 A SOLUÇÃO deve ser uma aplicação Web
1.1.4 A SOLUÇÃO deve obrigatoriamente ser compatível com o ambiente de execução de aplicação IBM WebSphere Application Server (WAS) for z/OS Versão 7 .x ou Microsoft Internet Information Services 7 .x, ou SAP Netweaver 7.4 de acordo com o dispositivo no Anexo IV – Ambiente Computacional do Banco do Nordeste.

4.2.4.1 Requisitos funcionais

Um requisito funcional define uma função de um sistema de *software* ou seu componente. Representa o que o *software* faz, em termos de tarefas e serviços. Uma função é descrita como um conjunto de entradas, seu comportamento e saídas. Requisitos funcionais descrevem o comportamento, as tarefas e os serviços que o usuário terá para interagir com o *software* para que opere o sistema (Figura 11). O comportamento esperado e descrito pelo requisito funcional referencia todo o processo de comunicação com o usuário e os meios de armazenamento até que um determinado objetivo seja alcançado.

Figura 11 – Requisitos funcionais: o que o *software* executa



Fonte: Vazquez; Simões, 2016.

São exemplos de requisitos funcionais expressões como:

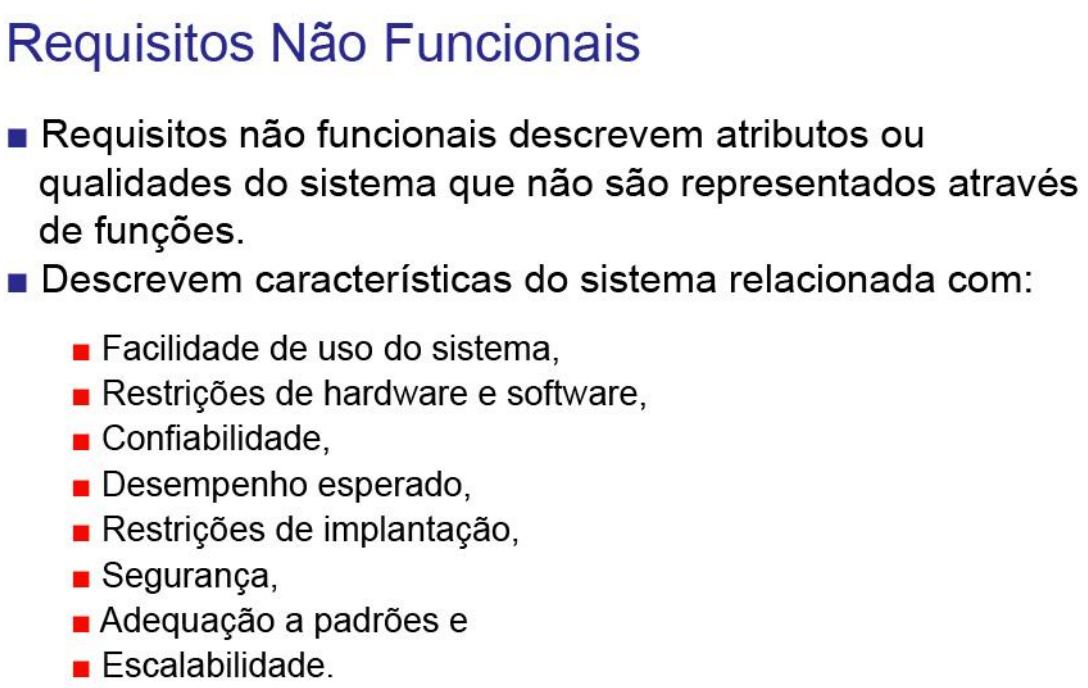
- *O sistema deve manter o cadastro de alunos.*
- *O sistema deve emitir a lista de todas as turmas e horários de aula.*
- *O sistema deve garantir que somente usuários da secretaria tenham acesso ao cadastro de alunos.*
- *O sistema deve possibilitar o registro de notas por disciplina.*

4.2.4.2 Requisitos não funcionais

Os requisitos não funcionais descrevem limitações aos requisitos funcionais para que, além de executar o que foi proposto pelo cliente, a aplicação funcione bem considerando-se termos de desempenho, usabilidade, confiabilidade, segurança, disponibilidade, manutenção e tecnologias envolvidas. Esses requisitos dizem respeito a como as funcionalidades serão entregues ao usuário do *software*. São necessidades que podem impactar o objetivo final do *software* se não forem contempladas em tempo de análise e desenvolvimento do projeto. Um único requisito não funcional pode gerar ou

limitar uma série de requisitos funcionais existentes. Por exemplo: um sistema foi projetado para possibilitar buscar as informações do endereço do cliente pela pesquisa de Código de Endereçamento Postal (CEP) na página dos Correios; no entanto, o tempo de resposta da pesquisa ultrapassa 5 minutos. Esse é um exemplo de requisito não funcional que não foi descrito ou considerado. A Figura 12 descreve tipos de requisitos não funcionais.

Figura 12 – Requisitos não funcionais



Os requisitos não funcionais surgem por várias razões, normalmente por necessidade dos usuários, orçamento limitado, políticas organizacionais ou fatores externos, como legislação e segurança. A definição de um requisito não funcional deve constar e ser bem descrita no documento de especificação do projeto, com dados como identificador do requisito, a que categoria atende (confiabilidade, usabilidade etc.), graus de prioridade e importância e outros detalhes.

São exemplos de requisitos não funcionais, com expressões que os ilustram:

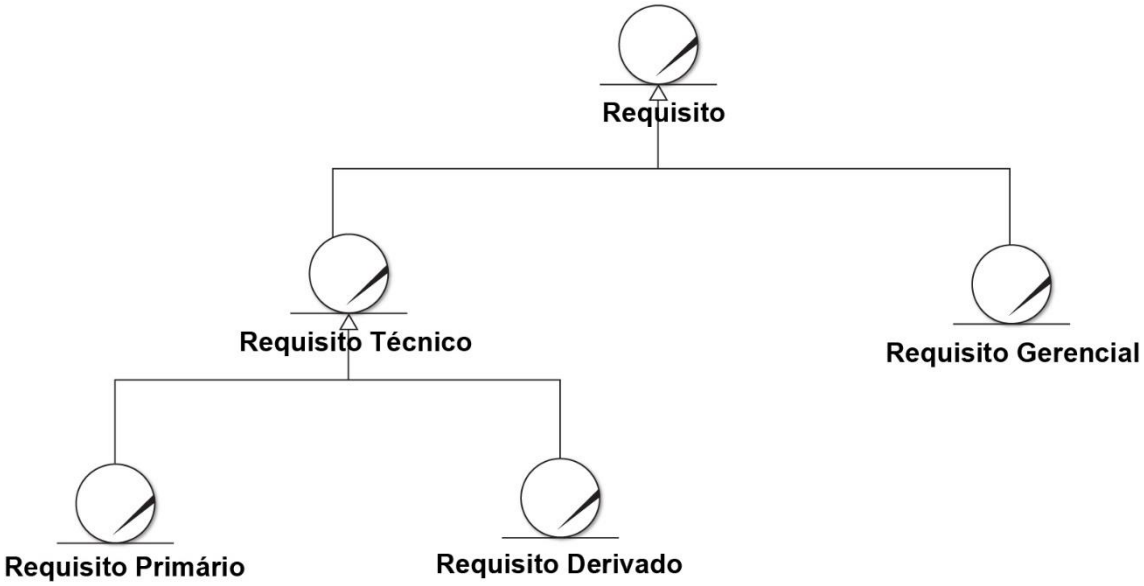
- desempenho (*Ao se registrar um item sendo vendido, a descrição e o preço do item devem aparecer em, no máximo, 2 segundos*);
- disponibilidade (*O sistema estará disponível 24 horas por dia e 7 dias por semana*);

- integridade/segurança (*Apenas usuários com privilégios de acesso de administradores poderão visualizar históricos de transações de clientes*);
- necessidades de internacionalização (*O produto será disponibilizado em inglês, mas de forma a permitir que versões em línguas latinas possam ser produzidas*);
- manutenibilidade (*Modificações de qualquer relatório deverão ser implementadas em até 24 horas*).

Em engenharia de requisitos, buscamos descrever os requisitos técnicos que representam as características técnicas do produto a ser desenvolvido, do que se excluem os requisitos gerenciais, que serão tratados no âmbito da gestão de projetos. Tais requisitos técnicos são obtidos das necessidades das partes interessadas, como já vimos.

Existem outros requisitos, chamados *derivados*, que fazem parte do modelo de solução adotado no desenvolvimento do projeto, conforme Figura 13. Os requisitos derivados são elementos importantes para a etapa de codificação ou implementação do produto. Como exemplo, imagine um requisito descrito assim: *O sistema deve funcionar 24 horas, no Alasca*. Isso pressupõe requisitos derivados, como: *O sistema deve funcionar na neve* e *O sistema deve ter baterias para nunca ficar sem energia elétrica*. Essas duas definições serão fundamentais quando da construção do código que atenda à demanda do projeto.

Figura 13 – Hierarquia de requisitos



Fonte: Paula Filho, 2019.

TEMA 5 – DOCUMENTO DE ESPECIFICAÇÃO DE REQUISITOS

A documentação de requisitos, também chamada de *especificação*, é uma necessidade do projeto de registrar os requisitos para ele selecionados. Trata-se de um contrato entre cliente e equipe de desenvolvimento, que estabelece tudo que deverá ser criado e executado pelo *software*, bem como todas as restrições e regras de negócio a que o projeto deve atender.

É importante descrever os requisitos em vários níveis de detalhe, para seu entendimento por cada parte interessada (*stakeholder*), como o cliente, o programador, entre outros (Figura 14). Para o cliente, geralmente, o interesse não reside em como o *software* será construído, mas em como serão as suas interfaces. Já o programador estará mais interessado nos recursos detalhados e em como implementá-los. Veremos isso mais detalhadamente, adiante.

Figura 14 – Documento de requisitos, cujo entendimento deve contemplar as várias partes interessadas de um projeto

Créditos: Bakhtiar Zein/Shutterstock.

As partes interessadas de um projeto são as várias unidades organizacionais de uma empresa, ou seja, os vários profissionais que atuarão com o *software*, os seus diversos usuários, e o desafio será construir uma visão de especificação de requisitos que todos compreendam e se sintam contemplados em sua perspectiva. O documento de especificação de requisitos será utilizado e lido pelas partes interessadas (clientes) e equipe de desenvolvimento do produto. Pensando no ponto de vista do cliente, ao avaliar as partes interessadas, perceberemos que há vários tipos de usuários do *software* que será produzido e, portanto, várias visões de necessidades que tenderão a direcionar a requisitos diferentes ou duplicados.

O analista de requisitos tem um grande desafio, que é, justamente compreender, entre as várias pessoas a quem se dirige, cada perspectiva e necessidade, se há omissões de requisitos, se há inconsistências, se há falhas no entendimento dos processos, antes que o projeto avance e certos requisitos tenham sido esquecidos ou não identificados, o que ocasionará muito retrabalho e perda de tempo e de recursos envolvidos.

Na equipe de desenvolvimento, há vários papéis a serem desempenhados. O gerente de projetos utiliza a especificação de requisitos para seu planejamento, cronograma, definição de tarefas, medição do tamanho funcional do *software*, custos, entre outros processos de gerenciamento. O analista projetista precisa avaliar os requisitos e elaborar como será a arquitetura

do *software* e que recursos tecnológicos essa arquitetura demandará. Os analistas de teste utilizarão os requisitos para criarem os planos de testes e validarem o resultado do código construído. Ou seja, cada integrante se servirá do documento de especificação dos requisitos para desenvolver suas tarefas na produção do *software*.

Nesse sentido, cabe avaliar que erros nessa especificação, como falta de requisitos ou requisitos trocados, causarão erros, também, no desenvolvimento de todos os papéis dos integrantes da equipe e, consequentemente, impactarão algum tipo de prejuízo para o projeto. Dado esse contexto, avaliamos que o analista de requisitos atua reunindo informações coletadas sobre o sistema e traduzindo as necessidades das partes interessadas para a equipe de desenvolvimento. Segundo Kerr (2015), para que isso ocorra com sucesso, é preciso que o analista de requisitos se atente a alguns aspectos como:

- **Preparação**: o analista de requisitos deve se preparar para as atividades, por meio de várias técnicas, que depois veremos detalhadamente. Por envolverem outras pessoas, as atividades devem ser planejadas com antecedência, estudadas em relação às pessoas envolvidas e estruturadas para que todas as informações relevantes sejam levantadas.
- **Identificação dos *stakeholders***: é fundamental que se identifique quem são os *stakeholders* do projeto, ou seja, quais são as partes interessadas que o impactam, qual seu papel e influência no desenvolvimento do projeto.
- **Desenvolvimento das competências interpessoais**: o analista de requisitos deve ser capaz de mediar conflitos, uma vez que estará em contato com várias pessoas com interesses distintos, formas de comunicação diferentes, opiniões divergentes, e deverá demonstrar poder de persuasão para conciliar esses diferentes interesses existentes em torno do projeto, além de determinar quais informações são, de fato, relevantes como requisitos para o projeto. Também deve possuir habilidades, como empatia, para entender os *stakeholders*, que nem sempre sabem se expressar com clareza e uma comunicação clara e objetiva.
- **Desenvolvimento da capacidade de escuta e compreensão**: o foco do analista de requisitos deve ser a compreensão do problema a ser resolvido e, para isso, ele deve saber escutar e compreender as pessoas,

tendo a capacidade de descrever todas as informações, traduzindo-as para a lista de requisitos técnicos do sistema.

Fora isso, alguns padrões na escrita do documento de requisitos devem ser seguidos para garantir que ele seja, ao final, claro, objetivo, detalhado e completo, tais como:

- Iniciar o documento com construções verbais como *O sistema deve...*
- Usar frases curtas como *O sistema deve permitir validar o CPF do usuário*.
- Cada requisito deve ter um identificador único.
- Precisa-se separar os requisitos funcionais dos não funcionais.
- Os requisitos não devem conter detalhes de implementação do projeto.
- Deve-se manter consistente uso dos termos do domínio de aplicação, ou seja, utilizar as mesmas palavras ao se abordar um mesmo assunto, em todo o documento.

Como já avaliamos, o documento de definição de requisitos deve ser escrito de maneira que o cliente possa entendê-lo, com uma descrição completa de tudo que se espera do sistema (*software*) proposto. Por outro lado, a especificação de requisitos é escrita em linguagem técnica a mais apropriada para o entendimento dos desenvolvedores da equipe. Pode ser criado um único documento, também, com essas duas visões; mas é mais indicado que se elaborem dois documentos distintos (Pfleeger, 2004).

Cada item descrito na definição de requisitos deve ter imediata correspondência na especificação para que nada seja perdido ou esquecido e, também, uma possibilidade de acompanhamento da implementação de todos os requisitos. Essa correta associação entre a definição e a especificação dos requisitos é o que possibilita projetar os casos de testes que serão realizados ao final do desenvolvimento para validar tanto o correto funcionamento, como os requisitos do projeto. Esse conjunto de métodos é conhecido como *gerência de configurações* e também permite determinar o impacto de mudanças, caso aconteçam.

A Figura 15 ilustra um documento de especificação de requisitos utilizando a linguagem natural do cliente. Claro que o exemplo apresentado não consiste na única maneira de se escrever esse tipo de documento e não contempla todos os problemas que existem.

Figura 15 – Exemplo de documento de especificação de requisitos

Requisitos Funcionais				
ID	**Descrição do Requisito**	**Importância**	**Fonte**	**Alocado**
M-1	O sistema deve permitir ao médico receber as informações do Atendimento, evoluir o paciente e alterar as informações: Incidente, Endereço e Prioridade	Essencial	Cliente	Sim
M-10	O sistema deve permitir ao médico colocar uma ocorrência “Em Espera” para atendimento posterior ou agendado	Pouca	Cliente	Não
M-11	O sistema deve permitir ao médico Rejeitar Envolvidos e informar o motivo (falta de recurso, infraestrutura ou leito), indicando o tipo de estabelecimento e o local e o responsável pela rejeição.	Média para baixa	Cliente	Sim
M-12	O sistema deve permitir ao médico Encerrar uma ocorrência após a finalização de sua execução, com ou sem alocação de recurso	Essencial	Cliente	Sim

Um dos problemas que podem ocorrer com o documento de especificação de requisitos é ele gerar confusão sobre o seu entendimento, uma vez que o cliente tem uma visão e o analista desenvolvedor pode pensar diferente daquela. O papel do analista de requisitos, utilizando-se de todo o seu conhecimento adquirido no levantamento de informações, nas reuniões com as partes interessadas e na sua própria experiência, é garantir que o documento de requisitos esteja completo e coerente com todos os envolvidos e interessados e que possa ser uma ferramenta útil, tanto para o desenvolvimento, quanto para as validações que serão solicitadas ao final do desenvolvimento do projeto. Esse documento é chamado como tal justo porque será a comprovação de todas as solicitações, do escopo do projeto, das suas definições, ou seja, será o documento que comprovará tudo que foi acordado para que o projeto seja executado e entregue ao cliente.

FINALIZANDO

Nesta etapa, abordamos os principais conceitos sobre requisitos, escopo do problema a ser resolvido, considerações para se garantir o entendimento do

projeto por todos os envolvidos e como expressar de forma clara e detalhadamente todas as descrições do *software*. Num primeiro momento, entendemos a importância da declaração do escopo para todo o processo de desenvolvimento e que, quanto mais completo for esse detalhamento, menos problemas e conflitos poderão ocorrer. Compreendemos, a seguir, como avaliar fatores limitantes e que possam afetar o projeto da solução, identificando situações que porventura impactem os requisitos e que não tenham sido declaradas formalmente, para não se correr o risco de criar uma solução inadequada.

Aprendemos ainda a considerar todos os envolvidos como **partes interessadas** e o quanto é imprescindível que o analista entenda o nível de interesse, bem como o grau de influência (poder) de cada parte interessada, na organização. Por meio de várias dicas e exemplos de requisitos, analisamos os tipos de requisitos e como eles são utilizados no projeto da solução. E compreendemos que cada requisito funcional identifica uma funcionalidade de que o usuário necessita e que o requisito não funcional pode ser tão importante a ponto de inutilizar um *software*, caso não tenha sido identificado. Por fim, estudamos como descrever um documento de especificação, utilizando alguns padrões formais de linguagem natural, mas também com um nível técnico, de forma clara, para que todas as partes interessadas consigam entender e aprovar os requisitos encontrados.

O mais importante desta etapa é perceber o quão importante é despender tempo e realizar uma boa análise de requisitos para se garantir um cliente satisfeito ou, pelo menos, minimizar problemas futuros em relação ao *software*.

REFERÊNCIAS

ABNT – Associação Brasileira de Normas Técnicas. **ABNT NBR ISO 9001:2015**: sistemas de gestão da qualidade – requisitos. Rio de Janeiro, 30 set. 2015.

CASTRO, M. C. V. de et al.; Principais causas de fracasso em projetos. **PMKB**, Belo Horizonte, 29 dez. 2015. Disponível em: <https://pmkb.com.br/artigos/principais-causas-de-fracasso-em-projetos/>. Acesso em 18 abr. 2022.

ESCOPO. In: MICHAELIS dicionário brasileiro da língua portuguesa. São Paulo: Melhoramentos, 2015. Disponível em: <https://michaelis.uol.com.br/moderno-portugues/busca/portugues-brasileiro/escopo/>. Acesso em: 18 abr. 2022.

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

PAULA FILHO, W. P. **Engenharia de software**: produtos. 4. ed. Rio de Janeiro: LTC, 2019. v. 1.

PFLEEGER, S. L. **Engenharia de software**: teoria e prática. 2. ed. São Paulo: Prentice Hall, 2004.

PMI – Project Management Institute. **Um guia do conhecimento em gerenciamento de projetos**: guia Pmbok. 6. ed. Newtown Square, 2018.

RAMOS, E. S. **Lidando com stakeholders**: boas práticas em análise de negócios e engenharia de requisitos. 1. ed. São Paulo: KDP, 2019.

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

VAZQUEZ, C.; SIMÕES, G. **Engenharia de requisitos**: software orientado ao negócio. 1. ed. Rio de Janeiro: Brasport, 2016.