

# Análise e definição de requisitos funcionais e não-funcionais do sistema

Conceitos, técnicas de levantamento, exemplos práticos e aplicação

---

## 1. O que são requisitos de software?

Requisitos de software são as condições e capacidades que um sistema precisa atender para resolver o problema para o qual foi criado. Eles representam a ponte entre a necessidade do cliente ou usuário e o que efetivamente será construído pela equipe de desenvolvimento. Definir requisitos de forma clara, completa e verificável é uma das atividades mais importantes — e mais desafiadoras — de todo o desenvolvimento de software, já que erros nessa etapa tendem a se propagar e se tornar cada vez mais caros de corrigir nas fases seguintes.

Os requisitos são tradicionalmente divididos em duas grandes categorias: requisitos funcionais e requisitos não-funcionais.

## 2. Requisitos funcionais

Requisitos funcionais descrevem **o que o sistema deve fazer**: as funcionalidades, ações e comportamentos específicos que ele precisa oferecer aos usuários. Eles respondem à pergunta "o que o sistema faz?" e normalmente são descritos como ações, como "o sistema deve permitir que o usuário...".

**Exemplo prático:** em um sistema de e-commerce, exemplos de requisitos funcionais incluem: "o sistema deve permitir que o cliente adicione produtos ao carrinho", "o sistema deve enviar um e-mail de confirmação após a finalização do pedido" e "o sistema deve permitir que o administrador cadastre novos produtos com nome, preço e descrição".

Requisitos funcionais costumam ser levantados a partir de entrevistas com usuários e stakeholders, análise de processos de negócio existentes, questionários e observação direta de como o trabalho é realizado atualmente (muitas vezes de forma manual ou com ferramentas inadequadas).

## 3. Requisitos não-funcionais

Requisitos não-funcionais descrevem **como o sistema deve se comportar**, definindo qualidades e restrições que não estão diretamente ligadas a uma funcionalidade específica, mas que afetam a experiência geral de uso e a viabilidade técnica do sistema. Eles respondem à pergunta "com que qualidade o sistema deve funcionar?".

### 3.1 Desempenho

Define o tempo de resposta esperado, a capacidade de processar um determinado volume de dados ou requisições simultâneas.

**Exemplo prático:** "o sistema deve responder a consultas de busca de produtos em até 2 segundos, mesmo com 10 mil usuários simultâneos conectados".

### 3.2 Segurança

Define requisitos relacionados à proteção de dados, controle de acesso e conformidade com regulações de privacidade.

**Exemplo prático:** "todos os dados de cartão de crédito devem ser criptografados tanto em trânsito quanto em repouso, seguindo o padrão PCI-DSS".

### 3.3 Usabilidade

Define expectativas relacionadas à facilidade de uso, acessibilidade e experiência do usuário.

**Exemplo prático:** "um novo usuário deve conseguir concluir o cadastro em até 3 minutos, sem necessidade de suporte", ou "o sistema deve atender aos critérios de acessibilidade WCAG nível AA".

### 3.4 Disponibilidade e confiabilidade

Define o tempo esperado de funcionamento do sistema sem interrupções e como ele deve se comportar diante de falhas.

**Exemplo prático:** "o sistema deve ter disponibilidade mínima de 99,5% ao mês", o que equivale a pouco mais de 3 horas de indisponibilidade tolerável por mês.

### 3.5 Escalabilidade e manutenibilidade

Define a capacidade do sistema de crescer (em usuários, dados ou funcionalidades) e a facilidade com que poderá ser modificado no futuro.

**Exemplo prático:** "a arquitetura deve permitir a adição de novos módulos de pagamento sem exigir alterações no núcleo do sistema de pedidos".

## 4. Resumo comparativo: funcionais x não-funcionais

| Aspecto                     | Requisitos funcionais                              | Requisitos não-funcionais                                |
|-----------------------------|----------------------------------------------------|----------------------------------------------------------|
| Pergunta que respondem      | O que o sistema faz?                               | Com que qualidade o sistema funciona?                    |
| Exemplo                     | Permitir login com e-mail e senha                  | Login deve responder em até 1 segundo                    |
| Verificação                 | Testado por meio de casos de uso e funcionalidades | Testado por meio de métricas e testes de carga/segurança |
| Visibilidade para o usuário | Direta e explícita                                 | Muitas vezes percebida indiretamente                     |

## 5. Processo de análise e definição de requisitos

## 5.1 Levantamento (elicitação)

Coleta inicial de informações por meio de entrevistas, questionários, workshops e observação, buscando entender as necessidades reais dos usuários e stakeholders.

**Exemplo prático:** em um projeto para uma clínica veterinária, a equipe entrevista recepcionistas e veterinários para entender como funciona hoje o agendamento manual de consultas, identificando gargalos e necessidades não atendidas.

## 5.2 Análise e priorização

Os requisitos levantados são analisados quanto à sua viabilidade, importância e prioridade, muitas vezes utilizando técnicas como MoSCoW (Must, Should, Could, Won't have) para organizar o que será desenvolvido primeiro.

**Exemplo prático:** "permitir agendamento online de consultas" é classificado como essencial (Must have), enquanto "enviar lembretes automáticos por SMS" é classificado como desejável (Could have) para uma versão inicial do sistema.

## 5.3 Especificação e documentação

Os requisitos são formalmente documentados, seja em um documento tradicional de especificação de requisitos, seja como histórias de usuário em metodologias ágeis, incluindo critérios de aceitação claros para cada um.

**Exemplo prático:** a história de usuário "Como recepcionista, quero agendar uma consulta em menos de um minuto, para atender o cliente rapidamente" é documentada com o critério de aceitação "o sistema deve preencher automaticamente o histórico do pet ao selecionar seu tutor".

## 5.4 Validação com stakeholders

Os requisitos documentados são revisados junto ao cliente e demais interessados, confirmando que refletem corretamente as necessidades identificadas antes de avançar para o design e desenvolvimento.

**Exemplo prático:** a equipe apresenta o documento de requisitos para a diretoria da clínica, que identifica um requisito ausente: a necessidade de bloquear horários para procedimentos cirúrgicos, que é então incluído antes do início do desenvolvimento.

## 6. Boas práticas na definição de requisitos

- Escrever requisitos de forma clara, específica e verificável, evitando termos vagos como "o sistema deve ser rápido" sem definir o que "rápido" significa em números concretos.
- Envolver diferentes tipos de usuários no levantamento, não apenas gestores, já que quem opera o sistema no dia a dia frequentemente identifica necessidades que a liderança não percebe.
- Registrar não apenas o requisito, mas também a justificativa por trás dele, facilitando decisões futuras quando o contexto do negócio mudar.

- Revisar e atualizar os requisitos ao longo do projeto, especialmente em contextos ágeis, em vez de tratá-los como um documento fixo e imutável definido apenas no início.

## 7. Aplicação prática no dia a dia dos projetos

A análise e definição de requisitos é aplicada por diferentes profissionais ao longo do projeto:

- **Analistas de negócio** conduzem entrevistas e workshops para levantar requisitos funcionais diretamente com os usuários finais e áreas de negócio.
- **Arquitetos de software** utilizam requisitos não-funcionais, como desempenho e escalabilidade, para tomar decisões importantes de arquitetura antes mesmo de iniciar o desenvolvimento.
- **Product Owners**, em times ágeis, traduzem requisitos em histórias de usuário priorizadas no backlog, equilibrando funcionalidades novas com melhorias de qualidade não-funcionais.
- **Equipes de qualidade (QA)** utilizam requisitos funcionais e não-funcionais como base para elaborar casos de teste, incluindo testes de carga e segurança específicos para validar requisitos não-funcionais críticos.

## 8. Considerações finais

A análise e definição cuidadosa de requisitos funcionais e não-funcionais é a base para a construção de sistemas que realmente atendam às necessidades dos usuários, tanto em termos do que fazem quanto da qualidade com que funcionam. Negligenciar os requisitos não-funcionais, em particular, é um erro comum que pode comprometer seriamente a experiência do usuário e a viabilidade do sistema em produção, mesmo quando todas as funcionalidades planejadas foram corretamente implementadas.