De: O Mentor

Requisitos claros reduzem retrabalho, mas não exigem prever tudo no início. Neste capítulo, veremos como itens de backlog, critérios de aceitação, Definition of Done (DoD) e automação podem apoiar conversas sobre valor e qualidade. Também distinguiremos práticas úteis de requisitos do Scrum.

Acesse a versão completa em: https://alessiojr.com/disciplines/gps/conteudo/00-livro/

Versão compilada em 20/9/2026 – Capítulo isolado do livro.

Acesse a versão completa em: https://alessiojr.com/disciplines/gps/conteudo/00-livro/

1 Requisitos evolutivos e documentação útil

A Figure 1 introduz a relação entre requisitos e automação. Em projetos de software, necessidades, restrições e hipóteses técnicas podem ser esclarecidas ao longo do trabalho. Documentação detalhada produzida muito cedo pode precisar de revisão; por outro lado, o Manifesto Ágil não elimina documentação, planejamento ou contratos. Seus valores indicam que os itens à direita também têm valor e devem ser usados na medida necessária ao contexto.



Figure 1: Ilustração de abertura: integração, automação e entrega contínua. *Fonte: Imagem gerada por IA, com supervisão do autor.*

O nível adequado de documentação depende, por exemplo, de risco, regulação, integração com terceiros, manutenção e acordos contratuais. Decisões de arquitetura, interfaces, critérios de qualidade e responsabilidades podem precisar de registro persistente. Itens do Product Backlog ajudam a manter requisitos visíveis e a orientar conversas, mas não substituem automaticamente esses artefatos.

2 Épicos, itens de backlog e Histórias de Usuário

Equipes organizam necessidades em diferentes níveis de detalhe. Um **épico** é uma iniciativa ampla que normalmente precisa ser refinada antes de caber em uma Sprint. Por exemplo, “oferecer apoio socrático ao estudante” pode envolver memória de conversas, políticas de resposta, interface e observabilidade.

O *Scrum Guide* não prescreve épicos, features ou Histórias de Usuário. São técnicas de gestão de requisitos que uma equipe pode adotar para tornar itens do Product Backlog compreensíveis e negociáveis. Um item selecionado para a Sprint deve ser suficientemente entendido para os Developers elaborarem um plano rumo ao Sprint Goal; seu formato pode variar entre uma história, uma hipótese, um defeito ou trabalho técnico.



Figure 2: Decomposição didática de um épico em capacidades e itens de backlog. *Fonte: Elaboração do autor.*

Versão compilada em 20/9/2026 – Capítulo isolado do livro.

A Figura 2 sintetiza visualmente o conceito discutido nesta seção.

3 Técnicas opcionais para detalhar itens

Uma **História de Usuário** pode expressar uma necessidade do ponto de vista de quem a utiliza. O padrão Connextra é uma convenção frequente, não uma exigência do Scrum:

Como `[persona]`, **quero** `[capacidade]` **para** `[benefício]`.

Por exemplo: “Como estudante, quero receber uma dica sobre meu código para identificar o próximo passo sem receber a solução completa.” Uma tarefa como “criar endpoint de dicas” pode continuar necessária, mas torna explícito o meio técnico, não o benefício por si só. Ela pode ser vinculada à história ou ser um item técnico independente.

INVEST é outra técnica opcional, proposta por Bill Wake, para discutir a qualidade de histórias: independência, negociação, valor, estimabilidade, tamanho e testabilidade. Ela não é uma Definition of Ready (DoR), e nem INVEST nem DoR são elementos prescritos pelo Scrum. Se a equipe estabelecer uma DoR, deve tratá-la como acordo revisável para apoiar o refinamento, sem impedir a adaptação do Product Backlog.

• Teoria: Perguntar pelo benefício

Considere o pedido de exportar conversas em PDF. Perguntar para que o relatório será usado pode revelar que o objetivo é acompanhar a evolução do estudante. A equipe pode então comparar alternativas, como um resumo na interface, mantendo registro das restrições de privacidade, auditoria e acesso. A conversa melhora a decisão; não elimina a necessidade de documentar escolhas importantes.

4 Critérios de aceitação, RNF e Definition of Done

Critérios de aceitação descrevem condições observáveis para um item específico e apoiam o entendimento compartilhado. Eles podem ser registrados em uma Issue, em exemplos executáveis ou em outro meio escolhido pela equipe. Um requisito não funcional (RNF) precisa de contexto mensurável, em vez de formulações vagas como “rápido” ou “seguro”.

Por exemplo, para uma tela de login: “no ambiente de homologação, com 100 usuários concorrentes e base de dados de teste, 95% das respostas de autenticação devem terminar em até 2 segundos, medidos do recebimento da requisição até a resposta HTTP, durante uma execução de 15 minutos.” Esse critério informa ambiente, carga, percentil, limite e método; ainda pode exigir revisão quando o contexto mudar.

Acesse a versão completa em: https://alessiojr.com/disciplines/gps/conteudo/00-livro/

Versão compilada em 20/9/2026 – Capítulo isolado do livro.

A **Definition of Done (DoD)** é o compromisso do Incremento no Scrum: um entendimento compartilhado que define quando ele atende ao nível de qualidade exigido. Se um item não atende à DoD, ele não faz parte de um Incremento utilizável. Critérios de aceitação podem ajudar a avaliar um item, mas não substituem a DoD; tampouco a DoD é um mecanismo de aprovação individual.

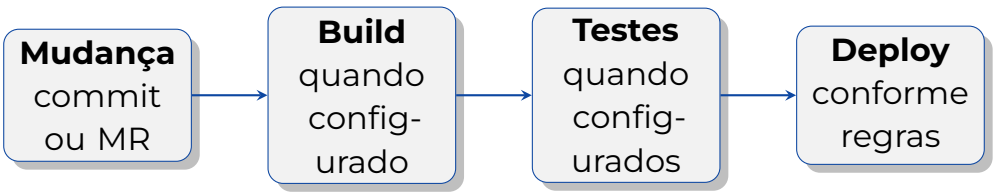
▷ Exemplo: Exemplo de DoD acordada pela equipe

Uma equipe pode acordar que um Incremento inclui código revisado, testes automatizados adequados ao risco, análise estática sem falhas bloqueadoras, documentação de uso atualizada quando afetada e validação no ambiente definido. O número de revisores, a cobertura mínima, os cenários manuais e o uso de Merge Requests são escolhas de contexto. Devem ser configurados e periodicamente revisados pela equipe.

Na **Sprint Review**, o Scrum Team e stakeholders inspecionam o resultado da Sprint e discutem mudanças de contexto para adaptar o Product Backlog. Não é uma etapa formal de aceite ou rejeição. A avaliação de que o Incremento atende à DoD ocorre durante o trabalho; feedback da Review pode gerar novos itens ou alterações de prioridade.
Também não concluir todos os itens selecionados não significa, automaticamente, que a Sprint falhou. A inspeção considera o Sprint Goal e o Incremento efetivamente pronto. Itens incompletos voltam ao Product Backlog para serem reconsiderados; Developers e Product Owner examinam o que aprenderam e adaptam o planejamento futuro.

5 Da integração ao deploy: CI/CD

Integração Contínua (CI) é uma prática de integrar mudanças frequentemente e validá-las com automação adequada, como build, testes e análise estática. Um pipeline é uma implementação possível dessa prática. Seus gatilhos, jobs, variáveis, regras de falha e ambientes dependem da configuração do repositório e da infraestrutura.
Entrega Contínua mantém mudanças em estado potencialmente liberável, com etapas de publicação e aprovação conforme o contexto. **Deploy contínuo** automatiza a promoção para produção quando as condições definidas são atendidas. Nem todo projeto precisa de deploy contínuo: risco, regulação e janela operacional podem justificar aprovação humana.



Versão compilada em 20/9/2026 – Capítulo isolado do livro.

Acesse a versão completa em: https://alessiojr.com/disciplines/gps/conteudo/00-livro/

Acesse a versão completa em: https://alessiojr.com/disciplines/gps/conteudo/00-livro/

Figure 3: Sequência configurável de integração, testes e disponibilização. *Fonte: Elaboração do autor.*

A Figura 3 sintetiza visualmente o conceito discutido nesta seção.

No GitLab, um arquivo `.gitlab-ci.yml` pode definir pipelines, mas sua simples existência não garante que todos os pushes ou Merge Requests os executem. Regras de `workflow`, `rules`, runners e permissões influenciam o comportamento. Da mesma forma, bloquear a integração de um Merge Request com pipeline com falha requer proteção de branch, política de aprovação e configurações compatíveis. Essas são convenções técnicas que podem compor a DoD da equipe, não regras do Scrum.

6 Aplicação no projeto integrador

▶ Prática: Acordos da disciplina e da equipe

No projeto integrador, a equipe pode registrar itens do backlog e critérios de aceitação em Issues, relacionar tarefas técnicas e documentar sua DoD. Quando a disciplina exigir evidências específicas, como configuração de CI ou uso de Merge Requests, isso deve ser entendido como regra acadêmica, separada das prescrições do Scrum.

7 Perguntas reflexivas

1. Como documentação, contratos e itens de backlog podem se complementar em um projeto com integração externa?
2. Por que uma equipe poderia adotar User Stories, INVEST ou uma DoR sem apresentá-los como requisitos do Scrum?
3. Como distinguir critérios de aceitação de uma DoD ao avaliar um Incremento?
4. Se alguns itens selecionados não terminarem, que evidências sobre Sprint Goal e Incremento ajudam a inspecionar a Sprint?
5. Quais configurações do GitLab precisariam ser verificadas antes de afirmar que um pipeline bloqueia um Merge Request?

Versão compilada em 20/9/2026 – Capítulo isolado do livro.

✓ Síntese

- O Ágil valoriza software em funcionamento e colaboração sem abolir documentação, planejamento ou contratos.
- Épicos, User Stories, INVEST e DoR são técnicas opcionais para organizar e refinar necessidades.
- Critérios de aceitação contextualizam um item; a DoD define a qualidade compartilhada de um Incremento utilizável.
- A Sprint Review inspeciona o resultado e adapta o Product Backlog; não é uma fase formal de aceite.
- CI/CD e políticas de Merge Request dependem de automação e configuração verificáveis.

No próximo capítulo, compararemos abordagens ágeis que organizam o trabalho por fluxo contínuo, como Kanban.

Acesse a versão completa em: https://alessiojr.com/disciplines/gps/conteudo/00-livro/

Versão compilada em 20/9/2026 – Capítulo isolado do livro.