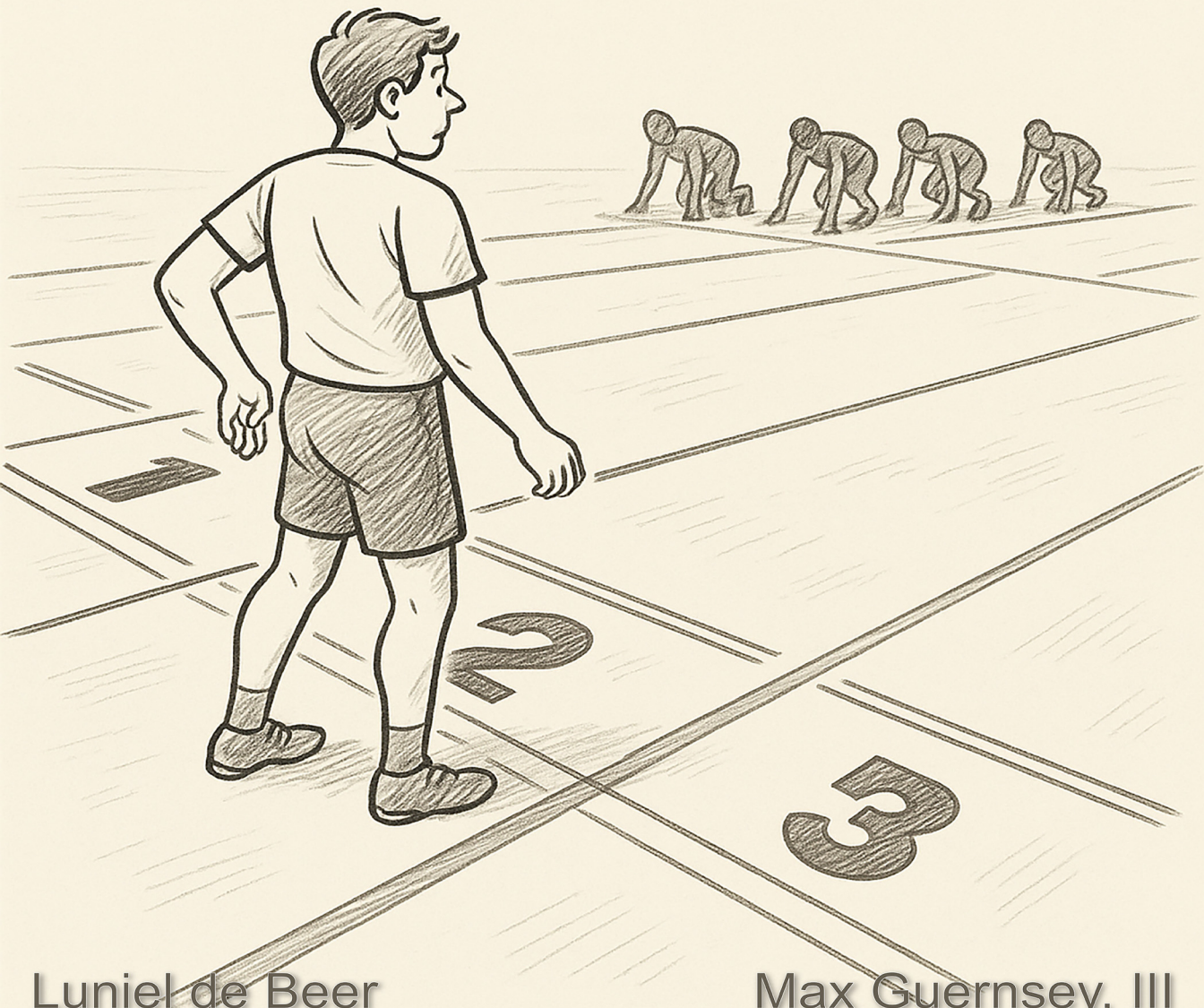


READY

WHY MOST SOFTWARE PROJECTS
FAIL AND HOW TO FIX IT



Luniel de Beer

Max Guernsey, III

Edição em Português do Brasil

Tuíte Este Livro!

Por favor, ajude Luniel de Beer e Max Guernsey, III compartilhando sobre este livro no [Twitter!](#)

O tuíte sugerido para este livro é:

Acabei de comprar Ready — um livro para líderes de software e equipes que desejam eliminar retrabalho, pendências e desalinhamento na entrega de software. #CodeReady

A hashtag sugerida para este livro é #CodeReady.

Descubra o que outras pessoas estão dizendo sobre o livro clicando neste link para pesquisar esta hashtag no Twitter:

[#CodeReady](#)

Ready (Edição em Português do Brasil)

Por Que a Maioria dos Projetos de Software Fracassa
e Como Solucionar Isso

Luniel de Beer e Max Guernsey, III

Este livro está disponível em <https://leanpub.com/ready-pt-BR>

Esta versão foi publicada em 2025-10-22



Este é um livro do [Leanpub](#). O Leanpub capacita autores e editores com o processo de Lean Publishing. O [Lean Publishing](#) é o ato de publicar um ebook em desenvolvimento usando ferramentas leves e várias iterações para obter feedback dos leitores, fazer ajustes até ter o livro certo e criar tração quando isso acontecer.

© 2025 Luniel de Beer e Max Guernsey, III

Em memória de Johann van Aardt, que reconheceu minha paixão, me apresentou à programação real e me ajudou a encontrar o caminho para um novo lar. E aos meus pais, cujo apoio inabalável—desde meu primeiro cliente até meu primeiro lar nos EUA—tornou tudo isso possível.

—Luniel

Para minha família, com quem o sol nasce e se põe.

—Max

Conteúdo

Sobre Este Livro	i
Para Quem É Este Livro	ii
Como Usar Este Livro	iii
Sobre os Autores	iv
Prefácio	v
 Parte I: Algo Está Faltando	 1
Capítulo 1: O Problema Oculto	2
Capítulo 2: O Custo das Fundações Ausentes	15
Capítulo 3: Introduzindo o Requirements Maturation Flow (RMF)	27
Capítulo 4: É Ágil?	29
 Parte II: Criando Espaço para a Prontidão	 30
Capítulo 5: A Primeira Extensão	31
Capítulo 6: Por Que as Pessoas Não Fazem Isso?	32
Capítulo 7: Trabalho Explícito de Prontidão (RMF 1)	34
Capítulo 8: Efeitos do RMF 1	36
Capítulo 9: Colocando o RMF 1 em Prática	37

Parte III: Controlando a Conclusão do Trabalho 39

Capítulo 10: A Próxima Necessidade 40

Capítulo 11: O Que as Pessoas Geralmente Fazem 42

Capítulo 12: Definindo uma Definição de Pronto 44

Capítulo 13: Definition of Done Personalizada (RMF 2) 46

Capítulo 14: A Vida Com RMF 1 e 2 50

Capítulo 15: Instalando o RMF 2 52

Parte IV: Implementação de Gates 54

Capítulo 16: O Requisito Final 55

Capítulo 17: Contexto sobre a Definição de Pronto para Início 57

Capítulo 18: Definindo uma Definição de Pronto para Início 59

Capítulo 19: Definição de Pronto para Início Personalizada (RMF 3) 61

Parte V: Síntese 65

Capítulo 20: A Maioria dos Prazos Não Importa 66

Capítulo 21: Competência 1: Fluxo de Maturação de Requisitos 68

Capítulo 22: Como o Trabalho e as Informações Fluem no Scrum com RMF 70

Capítulo 23: O Impacto do RMF 72

Capítulo 24: Transição para RMF 73

Capítulo 25: Depende de Você 75

Parte VI: Recursos 76

Apêndice A: O Scrum Não É o Problema 77

Apêndice B: O Synapse Framework™ 79

Apêndice C: Objeções e Obstáculos Comuns ao RMF 1 81

Apêndice D: Listas Base de Critérios DoD 82

Apêndice E: Listas Iniciais de Critérios para DoR 83

Índice 84

Sobre Este Livro

Ready é um livro para qualquer pessoa envolvida no desenvolvimento de software que está cansada de **subentrega, retrabalho crônico e requisitos pouco claros**.

Você pode já ter tentado investir em habilidades de execução da equipe, melhorar a implementação do seu framework de processo ou reformular o código e ainda precisar de mais melhorias.

Isso acontece porque a principal restrição para a maioria das equipes de desenvolvimento de software não são as habilidades da equipe, mas sim a maturidade dos requisitos. **Mesmo equipes maduras com as habilidades adequadas ainda enfrentam dificuldades** quando trabalham com requisitos imaturos.

Ready apresenta o RMF (Fluxo de Maturação de Requisitos), uma abordagem prática e profundamente estruturada para alinhar Produto e Engenharia sem substituir seu processo existente.

Seja você usuário de Scrum, Kanban ou algo personalizado, o RMF ajuda você a **estabilizar o escopo, eliminar pendências e entregar o que realmente importa**.

Se suas equipes se sentem estagnadas à beira do “quase pronto”, este livro mostrará como **quebrar o ciclo** e desbloquear sua(s) equipe(s) **de vez**.

Para Quem É Este Livro

Este livro é literalmente para qualquer pessoa envolvida no desenvolvimento de software. Desde engenheiros até gerentes de produto e de colaboradores individuais a executivos.

Este livro é para você se estiver envolvido no desenvolvimento de software e tiver notado que uma equipe com a qual você trabalha ou da qual faz parte tem um ou mais dos seguintes problemas:

- O trabalho frequentemente é transferido de uma iteração para a próxima
- As equipes de implementação sentem que estão tentando atingir alvos em movimento
- O trabalho fica aberto por tempo demais
- O trabalho é marcado como concluído, mas as coisas não estão realmente prontas
- O trabalho realizado não está alinhado com as expectativas
- O trabalho gera regularmente uma grande quantidade de defeitos
- Grandes quantidades de trabalho precisam ser refeitas regularmente

Se algum desses problemas soa familiar, *Ready* pode ajudar.

Como Usar Este Livro

Este livro foi projetado para ser prático. Não é um tratado teórico ou uma apresentação estratégica - é um manual prático para implementar o RMF (Fluxo de Maturação de Requisitos) baseado em trabalho real com clientes e testado em campo sob pressão real de entrega.

Os capítulos são escritos em sequência, mas o próprio RMF é modular. É composto por três práticas fundamentais:

- RMF 1: Colaborando para um entendimento compartilhado
- RMF 2: Controlando a conclusão do trabalho usando Definições Personalizadas de Pronto
- RMF 3: Controlando a implementação usando Definições Personalizadas de Preparado

Cada parte, ou “Hábito”, como chamamos, funciona por si só, mas eles se complementam. O livro foi projetado para ajudá-lo a abordá-los um por vez, em ordem. Essa estrutura reflete como recomendamos que as equipes adotem o RMF na prática - com cada Hábito sendo incorporado apenas depois que o anterior estiver funcionando.

Isso evita sobrecarregar as equipes e dá a cada mudança a melhor chance de se estabelecer. Você aprenderá mais sobre como fazer isso começando em [Capítulo 9](#).

Se você está procurando ajuda - seja conselho, coaching ou alguém para falar com sua equipe de liderança - sinta-se à vontade para entrar em contato conosco diretamente.

E se você está procurando suporte formal para implementar o RMF, a Producure oferece uma série completa de Programas projetados para guiar a adoção passo a passo. Você pode saber mais em <https://ready-book.link/rmf>.

Sobre os Autores

Luniel de Beer é o criador do Fluxo de Maturação de Requisitos (RMF), um sistema prático para corrigir as lacunas entre a intenção do produto e a execução da engenharia. Ele tem mais de 15 anos de experiência liderando transformações ágeis, conectando produto e engenharia, e ajudando equipes a entregar com clareza e confiança.

Luniel também originou o sistema de Gerenciamento de Capacidades da Producore, uma abordagem rastreável e escalável para modelar capacidades de produto. Ele concebeu o Desenvolvimento Orientado a PKB (PKBDD), um sistema versionado para gerenciar requisitos persistentes de produto. Essas ferramentas fazem parte de um framework de entrega mais amplo desenvolvido na Producore.

Max Guernsey, III é um arquiteto de software, educador e co-fundador da Producore, uma consultoria dedicada a corrigir falhas de entrega através de rigor estrutural e técnico. Com mais de duas décadas de experiência em design orientado a objetos, refatoração, desenvolvimento orientado a testes e padrões de projeto, ele tem tanto entregado sistemas críticos quanto treinado equipes de engenharia em larga escala. Seu trabalho combina práticas técnicas profundas com transformação comportamental e de processo para ajudar organizações a alcançar excelência sustentável em entregas.

Max contribuiu significativamente para o PKBDD e liderou o desenvolvimento da abordagem da Producore para o Desenvolvimento Orientado a Comportamento (BDD) através de sua profunda expertise em especificação comportamental.

Juntos, seu trabalho integra clareza, rastreabilidade e gating em um sistema coeso para entrega de software que se expande desde a prática da equipe até a capacidade organizacional.

Prefácio

Nota aos Líderes de Engenharia

Se você é um líder sênior em uma organização de engenharia, provavelmente não lhe faltam esforço, disciplina ou pessoas inteligentes. E ainda assim, de alguma forma, os projetos param. As metas escorregam. As expectativas não são atendidas. Não porque suas equipes são preguiçosas—mas porque algo fundamental está quebrado na forma como o trabalho é definido, moldado e entregue.

Este livro não é um guia de liderança. É uma ferramenta para diagnóstico estrutural. Ele revela o que está realmente acontecendo dentro de suas equipes—por que “quase pronto” continua se transformando em “não pronto”, e por que o progresso local raramente se traduz em resultados estratégicos.

Você pode não ver a *si mesmo* nestas páginas. Mas se suas equipes não conseguem entregar o que você precisa, você verá a *elas*. E quando isso acontecer, você finalmente terá a linguagem—e o sistema—para consertar isso.

De Luniel

Antes de tudo, este livro não teria sido possível sem o Max, cuja capacidade de ver através da névoa e da palha e destilar uma ideia em sua essência está absolutamente além de mim.

Como chegamos aqui?

Se eu olhar para trás, acho que é porque sempre quis entender como as coisas realmente funcionam. Fosse religião, nutrição ou desenvolvimento de software, eu continuava esbarrando no mesmo problema: respostas superficiais que não se sustentavam sob pressão. Então continuei cavando—perguntando não apenas o que fazemos, mas por quê, e o que está faltando quando não funciona.

Uma das primeiras rachaduras no sistema apareceu em um papel onde eu estava usando três chapéus: Scrum Master, Product Owner e Gerente de Desenvolvimento (!!) para uma equipe que entregava serviços de dados em uma empresa de tecnologia bem conhecida. Estávamos

fazendo o que o Scrum dizia—Sprints curtas, histórias em um backlog, planejamento em meio dia—mas toda vez que começávamos uma nova Sprint, encontrávamos atrito. A equipe não entendia completamente o problema, tínhamos que revisar e modificar requisitos durante a Sprint, dependências evitáveis surgiam e nos atrasavam, e etapas importantes eram esquecidas.

Então comecei a fazer algo diferente. Eu reunia a equipe e as partes interessadas em uma sala para cada história, percorria o problema em detalhes, criávamos a solução juntos e só então escrevia a história. O Sprint Planning diminuiu para uma hora, e nosso sucesso nas entregas disparou.

Lentamente, comecei a perceber que o sucesso não vem de trabalhar mais durante a Sprint. Vem da estrutura que você estabelece *antes* dela começar.

Mais tarde, depois de ouvir Jeff Sutherland falar sobre “Definições de Prontidão”, eu sabia que havia algo valioso ali—mas não era suficiente. Minha experiência com requisitos, UX, UI, pesquisa e, posteriormente, com BDD me mostrou que diferentes itens de trabalho exigem diferentes tipos de prontidão. Alguns precisam de especificações de comportamento. Alguns precisam de acesso ao sistema. Alguns precisam de um rastreamento completo de capacidades.

E todos eles precisam de um entendimento compartilhado que seja realmente confirmado—não presumido.

Conforme trabalhei com mais equipes, vi o mesmo padrão em todo lugar: etapas faltando, dependências não atendidas, equipes fazendo o seu melhor mas constantemente se desdobrando para corrigir problemas que deveriam ter sido evitados. Até mesmo grandes equipes lutavam—não porque eram fracas, mas porque faltava uma estrutura que tornasse a prontidão explícita.

O resultado de todo esse aprendizado, iteração e frustração é um sistema estruturado para gerenciar a prontidão.

É disso que trata este livro.

Espero que ele lhe dê clareza sobre onde estão os problemas reais e como corrigi-los. Espero que lhe dê linguagem para defender práticas que podem parecer “extras” mas são na verdade essenciais. E acima de tudo, espero que ajude as equipes a entregar com menos estresse, menos surpresas e resultados muito melhores.

Se fizermos isso certo, economizaremos bilhões de dólares para a indústria.

Mas mais importante, devolveremos a sanidade às pessoas.

De Max

Tenho trabalhado neste problema de vários ângulos por décadas, mas meu progresso foi interrompido até conhecer Luniel.

Isso porque, antes de conhecê-lo, eu estava abordando o problema, fundamentalmente, como um problema técnico. Eu estava focado em ajudar as equipes a adotarem coisas como Desenvolvimento Guiado por Testes (TDD), refatoração, design de software avançado e, posteriormente, Desenvolvimento Guiado por Testes de Aceitação (ATDD) ou Desenvolvimento Guiado por Comportamento (BDD).

Na maioria desses casos, o problema abordado neste livro era tratado como um detalhe de implementação para estabelecer essas práticas técnicas.

Isso não quer dizer que eu não valorize mais as práticas técnicas. Ainda acho que elas são profundamente importantes, mas elas não abordam diretamente o problema da prontidão no desenvolvimento de software. Em vez disso, elas *expõem* esse problema e então as pessoas aplicam um remendo em seu processo para resolvê-lo “apenas o suficiente” para suportar as práticas técnicas que estão tentando implementar.

Também quero abordar a questão de quem este livro pode ajudar. A resposta curta é “provavelmente quase todos no desenvolvimento de software”, mas a resposta real contém nuances que ajudam a mapear isso para vários ambientes sem mudar o significado básico.

Existem equipes que precisam da solução apresentada neste livro. Você conhecerá uma versão simplificada de uma delas no Capítulo 1.

Existem também equipes que não **precisam** estritamente de um sistema como o que propomos, mas ainda assim poderiam se beneficiar dele.

A melhor equipe com a qual já trabalhei—facilmente uma unidade padrão de desvio acima da **próxima melhor equipe**, se não duas—estava aninhada na região interiorana do Oregon Central. Eles tinham um desempenho tão alto que conseguiam superar a ausência de tal sistema por pura competência. No entanto, meu gerente na época, Tom Barreras, uma vez me disse algo como “Percebi que nossas histórias fluem melhor quando dedicamos algum tempo discutindo os testes antecipadamente.”

Isso, novamente, era algo que eu via na época através da lente do desenvolvimento de testes e da execução técnica, mas agora sei que era mais um indicador de que a *prontidão* era um fator que afetava a equipe... aquela equipe em particular era tão capaz e rápida em responder que podia ter sucesso reagindo aos impedimentos conforme eles aconteciam, em vez de preveni-los desde o início.

Mesmo se você for o tipo de pessoa que não precisa estritamente se preocupar com prontidão porque consegue superá-la, ou se trabalha com uma equipe do mesmo tipo, você ainda pode se beneficiar do conteúdo deste livro.

Parte I: Algo Está Faltando

*Quando fazer as mesmas coisas melhor não ajuda, procure o **que não está sendo feito**.*

Capítulo 1: O Problema Oculto

Esta é uma história verdadeira¹ sobre um banco. Vamos chamá-lo simplesmente de “O Banco”. É um tipo de Instituição de Crédito Federada que serve como parte da infraestrutura financeira nacional dos Estados Unidos.

Nós (Luniel e Max) fomos trazidos para O Banco porque ele estava tendo dificuldades para entregar um projeto de software. Era um dos ambientes mais disfuncionais que já tínhamos visto, e é por isso que escolhemos este como o estudo de caso de abertura: se uma mudança significativa era possível em O Banco, ela é possível em qualquer lugar.

Uma Nota Rápida sobre Projetos

Quando usamos o termo “projeto” neste livro, nos referimos a ele no contexto de gerenciamento de projetos. Embora existam diferentes ideias sobre o que a palavra significa, estamos usando a definição do [Project Management Institute](#):

“Um projeto é um empreendimento **temporário** realizado para criar um produto, serviço ou resultado único.”

Isso significa que um projeto tem um início e fim definidos. Quando um projeto é encerrado, o conhecimento e os artefatos do projeto são arquivados, os membros da equipe são liberados e os contratos são finalizados.

Neste livro, um projeto é fundamentalmente sobre execução. A maioria dos projetos, conforme definido pelo PMI, começa com viabilidade ou design. A visão e a estratégia já ocorreram quando um projeto é iniciado.

Um projeto nasce dessa visão e estratégia, e seu sucesso ou fracasso é baseado em se os objetivos imaginados são realizados—*não* em se esses objetivos eram os corretos.

Você pode usar a palavra “projeto” de forma diferente, e tudo bem. Apenas saiba que quando a usamos, estamos nos referindo à definição e ao contexto acima.

¹Alteramos detalhes identificadores para proteger a privacidade das pessoas e instituições envolvidas.

Nada disso pretende implicar que aprovamos o uso de gerenciamento de projetos para desenvolvimento de software. Muito pelo contrário. Mas reconhecemos que ele *é* usado mesmo assim. Abordamos esse problema mais tarde, em [Capítulo 6](#).

1.1: A Reescrita Clássica

O Banco estava reescrevendo seu portal de pagamento de empréstimos por várias razões.

O sistema antigo, uma solução totalmente em C#/.NET, estava cheio de bugs. Além de degradar a satisfação do cliente, também gerava um fluxo incessante de incidentes de suporte muito caros nos quais alguém tinha que manipular manualmente o banco de dados para corrigir um erro feito pelo sistema.

O sistema antigo também estava decadente do ponto de vista de manutenibilidade. Era quase impossível para os engenheiros fazerem mudanças significativas e, mesmo quando podiam, era uma proposição extremamente arriscada.

A reescrita deveria mudar isso.

O novo sistema ainda teria um backend em C#/.NET, mas este seria totalmente coberto por testes. O frontend seria implementado em OutSystems, uma popular solução low-code ou no-code que permite que uma organização defina um aplicativo em um lugar e obtenha um aplicativo web, um aplicativo Android e um aplicativo iOS gerados automaticamente sempre que decidir publicar suas alterações.

A esperança era que o novo sistema estivesse livre de bugs, melhorando tanto a satisfação do cliente quanto reduzindo significativamente os custos de suporte.

Eles também esperavam que a reescrita desbloqueasse os desenvolvedores—com a combinação de uma abordagem mais disciplinada para o backend e a abordagem low-code para o frontend reduzindo muito o custo e o risco de novos recursos.

Um efeito colateral positivo da mudança para OutSystems era que eles obteriam um aplicativo móvel limpo e moderno em ambas as principais plataformas.

Esse era o sonho quando eles haviam começado três anos antes do início desta história. A realidade era que, até agora, as equipes não haviam entregue **nada**.

1.2: Perspectivas sobre o Problema

Quando conversamos com a equipe de liderança executiva, ouvimos uma frustração muito natural com o fato de terem feito investimentos tão grandes sem absolutamente nenhum movimento estratégico.

Eles haviam tentado tudo, na perspectiva deles. Mudaram funcionários, aumentaram o número de funcionários, alteraram o orçamento, aumentaram a pressão e trouxeram uma parada de consultores (dos quais foi fortemente sugerido que éramos os últimos). Nada parecia melhorar—pelo menos não de uma forma que eles pudessem medir, porque tudo o que viam era que o “ponteiro” estava em zero em um trimestre e continuava em zero no trimestre seguinte.

Eles não queriam mais “progresso invisível”. Eles queriam *resultados*.

Quando conversamos com a gerência dentro da Organização de Produto, obtivemos uma história um pouco diferente (mas ainda similar) porque eles trabalhavam mais diretamente com a Engenharia.

Não é que as equipes não fizessem nada, pelo menos não na opinião delas. É que as equipes nunca faziam o que lhes era pedido. Era praticamente garantido: não importava quão simples fosse o pedido e nem quão claramente fosse comunicado, você acabava com algo completamente diferente quando chegava a hora de avaliar o que as equipes haviam produzido.

A situação chegou ao ponto em que a piada corrente era algo do tipo “Precisamos descobrir como pedir o que não queremos, para pelo menos termos uma chance de conseguir o que queremos.”

Os executivos de engenharia viam as coisas de forma bem diferente.

Para eles, o problema era que o Product não estava entregando requisitos acionáveis e nem requisitos *suficientes*. Se o Product pudesse simplesmente “entrar nos eixos”, as equipes conseguiriam entregar o que eles queriam dentro do prazo e do orçamento.

Eles haviam feito investimentos sérios na modernização de como o código era escrito e entregue e, na visão deles, o Product não estava entregando requisitos claros.

Quando conversamos com outros consultores (que nos indicaram para a organização), eles identificaram corretamente a disfunção que estavam vendo: todos pareciam muito focados em culpar outra pessoa. A razão pela qual nos trouxeram inicialmente foi que estavam preocupados com a estratégia de pessoal e queriam uma avaliação dos colaboradores

individuais, mas eles consideravam que o comportamento acusatório e a mentalidade de capataz no nível executivo eram a principal fonte dos problemas.

1.3: Nossa Investigação

Nossa missão inicial era avaliar as equipes e tentar ajudá-las a melhorar suas habilidades, se necessário, então começamos a investigar as pessoas que trabalhavam na linha de frente.

Definitivamente havia espaço para melhorias.

Os colaboradores individuais do lado do Product não tinham realmente as habilidades necessárias. Na realidade, eram principalmente gerentes de projeto que haviam sido empurrados para o papel de Product Owner (PO) ou Product Manager.

Como resultado, metade deles escrevia requisitos “vagos” e depois aceitava (literalmente) qualquer coisa que as equipes fizessem naquela iteração sem qualquer análise crítica. A outra metade escrevia o mesmo tipo de requisitos e depois alegava que as equipes “deveriam ter sabido” coisas que nunca foram mencionadas e mantinha os itens de trabalho abertos por tempo praticamente indeterminado.

O banco costumava gerenciar e acompanhar seu backlog de trabalho. Embora o que abordamos neste livro seja majoritariamente ortogonal ao Scrum, usamos a terminologia do Scrum ao longo do texto porque a maioria – ou pelo menos uma pluralidade – das equipes usa Scrum.



Definição: Scrum

Scrum é um framework leve que ajuda pessoas, times e organizações a gerar valor através de soluções adaptativas para problemas complexos. Em resumo:

1. Um Product Owner ordena o trabalho para um problema complexo em um Product Backlog
2. O Time Scrum transforma uma seleção do trabalho em um incremento de valor durante um Sprint
3. O Time Scrum e seus stakeholders inspecionam os resultados e ajustam para o próximo Sprint
4. Repete

Se você não está familiarizado com Scrum e sua terminologia, recomendamos que revise a versão 2020 do [Guia Scrum](#). É uma leitura rápida e esclarecedora.

Da mesma forma, descobrimos que as equipes técnicas estavam bem abaixo do típico em termos de habilidades de codificação (-2σ , na melhor das hipóteses) e ainda por cima eram altamente resistentes a mudanças. Como consequência natural, a qualidade do código era péssima.

No entanto, segundo as equipes, não era esse o motivo pelo qual não estavam entregando. Para elas, eram os requisitos vagos e as mudanças durante o Sprint pelo Product que estavam matando o projeto.

...e ninguém sequer falava sobre o problema maior, aquele tão absurdo que parece inventado até você vivenciá-lo.

As equipes de engenharia tinham o hábito de não entender um requisito, construir algo aleatório e depois exigir crédito por terem “finalizado um item de trabalho”.

Não estamos falando de uma pequena má interpretação. Estamos falando de uma desconexão total: pedíamos para eles desativarem a aplicação de fundos ao principal sob certas circunstâncias, e eles desativavam a capacidade de adicionar um endereço de e-mail secundário de confirmação.

Depois nos diziam que foi isso que pedimos.

1.4: Investigando Mais A Fundo

Ambas as lacunas de habilidades poderiam ser tratadas, mas estávamos céticos de que fossem os verdadeiros obstáculos.

Havia algo mais errado, então investigamos mais profundamente. Começamos com esta pergunta: Por que escrever requisitos levava tanto tempo e produzia resultados tão ruins?

Uma razão é que o conhecimento necessário para escrever um requisito significativo era extremamente escasso. Um pouco dele estava em posse das equipes de Engenharia e Product.

Parte dele estava embutido no código do sistema legado. Parte dele havia desaparecido completamente. A maior parte, no entanto, estava armazenada como conhecimento tribal em especialistas no assunto espalhados pelas várias unidades do banco. Isso significa que construir um requisito que realmente fizesse avançar um objetivo estratégico era uma atividade extremamente trabalhosa e demorada.

Em contraste com isso, havia um apetite insaciável por funcionalidades de uma equipe de liderança faminta por recursos. O mandato era “mantenha os engenheiros trabalhando—alimente-os com requisitos”. O foco estava na quantidade de requisitos para manter as

equipes ocupadas—uma perspectiva contrária ao cuidado e tempo necessários para definir um requisito que “moveria a agulha”.

1.5: Melhorar as Coisas Não Melhorou a Situação

Todos esses são problemas que podem ser abordados, mas abordá-los não ajudou.

Melhorias anteriores nas técnicas de desenvolvimento de software não haviam ajudado as equipes a entregar, mas Max fez um esforço para ajudar as equipes a melhorarem ainda mais.

Ele introduziu conceitos revolucionários das doutrinas de programação de meados do século XX como “não copie e cole esse código 27² vezes, coloque-o em uma função e chame-a”. Apenas essa sugestão melhorou dramaticamente a qualidade do novo código e permitiu que eles comesçassem a melhorar a qualidade.

Esse e outros conselhos básicos de codificação os ajudaram a escrever um código melhor que eles poderiam manter mais facilmente no futuro.

...mas isso não ajudou a fazer o projeto avançar.

Do lado do Produto, Luniel conseguiu introduzir o BDD e garantir que os Product Owners avaliassem completamente os requisitos antes de serem entregues às equipes.

Ele conseguiu fazer com que as equipes colaborassem e os usassem para avaliar se um Item do Backlog do Produto (PBI) estava realmente pronto.

²Isso não é apenas não hiperbólico, nem é o pior caso. Em um caso, havia quase cem duplicatas exatas do mesmo algoritmo.



Definição: Item do Backlog do Produto (PBI)

Um Item do Backlog do Produto (PBI) é uma unidade discreta de trabalho no backlog do produto que representa uma potencial mudança, adição ou melhoria ao produto. PBIs podem assumir várias formas—funcionalidade, correção de bug, melhoria técnica, tarefa de pesquisa, etc.—e são definidos por sua contribuição para o valor do produto.

Muitas equipes se referem aos PBIs como “histórias” ou “histórias de usuário”, mas o termo correto do Scrum é “Item do Backlog do Produto” ou “PBI”. Uma vez que um PBI é comprometido em uma Sprint, ele também faz parte do Backlog da Sprint. Por simplicidade e neutralidade, usamos “Item do Backlog do Produto” ou “PBI” para nos referir a qualquer item de trabalho que a Equipe Scrum está gerenciando—seja ele chamado de Item do Backlog do Produto, Item do Backlog da Sprint (SBI), história de usuário, história, item de trabalho ou item de backlog.

Isso adicionou *clareza*, mas não criou *fluxo*.

Com a ajuda dos consultores parceiros que nos trouxeram, conseguimos (temporariamente) aliviar a pressão absolutamente esmagadora colocada sobre as equipes e autores de requisitos.

Isso pode ter ajudado a criar um pouco de confiança, mas não gerou resultados tangíveis.

Começamos até a construir uma base de conhecimento que ajudava as pessoas a rastrear o conhecimento do negócio necessário para escrever requisitos e identificava os lugares onde havia lacunas nesse conhecimento.

Isso acelerou a escrita de requisitos, mas não fez o produto sair pela porta.

Depois de meses de interações, havíamos ajudado a liderança a *ver* onde estavam, mas eles não estavam nem perto de seus objetivos. E não estavam chegando mais perto.

Eles estavam se preparando para voltar à sua antiga estratégia de “sobrecarregar” as equipes para garantir que estivessem sempre ocupadas.

1.6: Um Processo de Eliminação

Teria sido fácil simplesmente levantar as mãos e dizer “isso é sem esperança”. Havia várias desculpas nas quais se apoiar:

- A equipe de engenharia tinha baixa qualificação (tinha mesmo)
- Os autores de requisitos tinham o conjunto errado de habilidades (tinham mesmo)
- A liderança estava esmagando as equipes com expectativas irrealistas (estava mesmo)
- A organização estava perdendo conhecimento crítico do negócio necessário para funcionar (estava mesmo)
- Os executivos não pareciam confiar uns nos outros (não confiavam mesmo)

Todas essas coisas eram verdadeiras. No entanto, melhorias haviam sido feitas em todas essas áreas e nenhuma delas parecia melhorar o problema fundamental. Nenhuma delas ajudou as equipes de Engenharia, Produto, gestão ou a diretoria a se moverem mais perto de seus objetivos.

...e aí está a dica para a solução: Todos os problemas listados anteriormente eram os problemas que as pessoas já podiam ver.



Se as variáveis que você pode ver não fazem diferença, deve haver uma variável que você **não vê** que faz.

O verdadeiro problema era aquele que ninguém sequer sabia que existia.

1.7: Em Busca do Verdadeiro Culpado

Na busca pelo verdadeiro culpado—aquilo que realmente estava impedindo o banco de alcançar seus objetivos—precisávamos começar por algum lugar.

Um lugar razoável para olhar era o que os PBIs que falharam (a maioria deles) tinham em comum.

Começamos eliminando as coisas que podíamos perceber que não eram comuns porque variavam muito:

- Qual parte do sistema: alguns PBIs afetavam apenas o backend, outros apenas o frontend, outros ainda afetavam ambas as partes
- Qual equipe executou o trabalho—não parecia importar quem fazia o trabalho, havia uma alta probabilidade de falha
- Qual PO criou o trabalho—o mesmo que acontecia com as equipes

Então começamos a olhar para as coisas que eram comuns. A lista não era longa, mas também não era curta:

- As equipes de engenharia
- Os product owners
- A liderança
- A cultura
- O ambiente de desenvolvimento
- As práticas de engenharia
- A técnica de elaboração de requisitos
- O domínio (financeiro)
- Os serviços dependentes

Muitos desses também podiam ser descartados imediatamente. As equipes, product owners, liderança, cultura e ambiente de desenvolvimento haviam sido todos recentemente melhorados sem nenhum impacto real na produção significativa. Nós pessoalmente havíamos ajudado a melhorar as práticas de engenharia e de elaboração de requisitos e confirmado que essas melhorias haviam se mantido, mas ainda assim não estava ajudando.

Não se pode culpar o domínio. Finanças é um dos tipos mais antigos de cálculos realizados na história registrada. É extremamente maduro. Além disso, outros bancos estavam implantando software, refutando performaticamente a hipótese (obviamente forçada) de que bancos simplesmente não conseguem fazer isso.

Os serviços dependentes também não podiam ser culpados, já que estavam tendo tanta dificuldade para mudar quanto a iniciativa que estávamos analisando...

...mas isso nos fez pensar: E se começássemos a analisar as causas das falhas?

1.8: Dissecando as Sementes do Fracasso

Um PBI falhou em fazer o produto avançar porque a equipe fez, como costumava fazer, algo completamente aleatório e quase inteiramente não relacionado ao pedido. Isso, obviamente, é um sinal de que eles não entenderam o item de trabalho. Então o entendimento era um grande candidato, mesmo que já tivéssemos trabalhado nisso quando os ajudamos a adotar o BDD.

Outro PBI falhou em ser concluído porque eles fizeram os cálculos errados. Essa é mais uma evidência de que o entendimento poderia ser o problema central.

Um terceiro item de trabalho que analisamos não foi realmente verificado adequadamente pelo Product Owner—ela apenas carimbou quando a equipe disse que era hora de fechar. Isso meio que tensionou nossa hipótese, mas ainda se poderia argumentar que ela não entendeu como o item de trabalho se encaixava em um plano de nível superior.

Talvez. Mais ou menos. Se apertássemos bem os olhos quando olhávamos dessa forma.

Então nos deparamos com um PBI que não se encaixava no padrão de forma alguma. A equipe parecia entender—embora não haja como verificar se realmente entendeu. Mas não importava: eles nunca tiveram a chance de ter sucesso ou falhar por conta própria porque esbarraram em uma dependência que precisava ser atualizada e tiveram que adiar seu trabalho por várias Sprints.

Mesmo se eles *não* entendessem o que deveriam fazer, eles nunca tiveram chance com aquele item do backlog, portanto o entendimento **não** era o problema nesse caso.

Um caso fora do padrão, é claro, não é suficiente para refutar uma causa raiz específica, mas despertou nossa curiosidade. Começamos a procurar outras evidências que contrariassem nossa hipótese.

E encontramos. Havia itens de trabalho que:

- Falharam porque a equipe sabia que não entendia, mas ninguém conseguiu localizar um especialista no assunto para resolver a questão
- Mudaram para uma experiência pior para o usuário como uma solução alternativa para como os serviços upstream funcionam
- Tiveram que ser adiados porque as dependências upstream não estavam prontas
- Não puderam ser concluídos porque os testadores não conseguiram coletar dados de teste a tempo
- Foram fechados mas tiveram que ser refeitos porque a solicitação em si estava incorreta
- Falharam porque a equipe não percebeu o quão complexo o código existente já era
- Simplesmente não foram estimados
- Foram massivamente subestimados³
- Mudaram durante a Sprint porque o PO finalmente obteve o conhecimento do domínio necessário
- Pareciam ter mudado após a Sprint (da perspectiva da equipe) porque o PO e a equipe nunca concordaram sobre o que significava

³Não queremos dizer apenas que eles erraram. Parecia que talvez as equipes simplesmente colocaram o valor “3” em todos os campos de estimativa para uma série de PBIs.

A lista continua, mas isso é suficiente para esta história.

Pode-se argumentar que cada um desses casos está ligado ao “entendimento” de alguma forma—e certamente uma falta de entendimento estava *envolvida* em muitos deles—mas isso não significa que a falta de entendimento foi a causa... especialmente considerando que já havíamos trabalhado no entendimento compartilhado e isso não havia realmente ajudado.

Então nos ocorreu. Havia uma peça mais fundamental faltando. Nos casos em que o entendimento deficiente estava envolvido, isso era apenas uma causa proximal.

A causa distal era muito mais ampla.

1.9: Um Ponto de Intersecção

A única coisa que todos os PBIs que analisamos por modo de falha tinham em comum era isto: Todos foram **iniciados prematuramente**.

Quando um item de trabalho falha porque a equipe sabia que não entendia o problema e não conseguiu encontrar um especialista para ajudá-los a entender, isso significa que a equipe iniciou um PBI sabendo que não compreendia o problema.

Quando um item do backlog precisa mudar para uma experiência pior devido ao funcionamento de um serviço upstream, isso significa que o item de trabalho foi iniciado sem realmente entender o impacto do serviço upstream.

Adiamento porque uma dependência upstream não estava pronta no final significa que não havia garantia de sua prontidão no início.

...e assim foi com todos os outros casos: Os testadores não tinham dados prontos ou não sabiam como obtê-los antes do início de um PBI, os requisitos não foram realmente verificados antes de serem entregues à equipe, o código não foi investigado antes do trabalho ser iniciado, a estimativa era insuficiente ou inexistente, o conhecimento do domínio estava faltando e, é claro, o entendimento compartilhado não foi verificado.

O problema, como descobrimos, era que a implementação começava em itens de trabalho antes que esses itens estivessem *prontos*.



Em nossa experiência, a maioria dos itens de trabalho que falham em ser entregues o fazem porque **não estavam prontos** quando a implementação começou.

Então nos propusemos a ajudar O Banco com isso.

Neste ponto, você pode pensar que apenas dizer que os PBIs devem estar prontos seria suficiente. No entanto, acontece que não é tão fácil de implementar. “Compre na baixa, venda na alta” é uma ideia igualmente simples.

Há uma peça faltando que é necessária para colocar o bom conselho em prática.

1.10: A Grande Virada

Encontramos a peça que faltava no quebra-cabeça que os desbloqueou, que por sua vez, desbloqueou a iniciativa.

Quando concluímos esse compromisso, os engenheiros ainda estavam bem abaixo da mediana em termos de habilidade. Os POs ainda não tinham as habilidades certas. A cultura ainda não estava corrigida...

Ainda assim, o produto finalmente começou a avançar e acabou sendo lançado.

Ao final deste livro, você saberá qual é a peça que faltava e o que é necessário para colocá-la em prática. E você será capaz de desbloquear organizações que parecem estar retidas por uma parede invisível.

Uma Nota Rápida sobre Escopo

Este livro trata de um problema muito específico e difundido: a falta de estrutura, clareza e maturidade na transferência entre negócios e engenharia. Ele assume que algo foi escolhido para implementação e se concentra em garantir que o trabalho de construção aconteça com entendimento compartilhado, prontidão e conclusão rastreável.

As técnicas neste livro não dizem o que construir, por que construir ou como descobrir se é a coisa certa a construir. Se sua organização carece de gerenciamento de produto real ou ciclos de feedback significativos, não estamos tentando resolver isso aqui. O que oferecemos, em vez disso, é uma maneira de tornar essas lacunas mais visíveis e reduzir o custo de descobrir que você estava errado.

Usado no contexto correto, esta solução traz fluxo, segurança e clareza. Mas como

qualquer sistema, pode ser mal aplicado—especialmente quando usado isoladamente ou sem consciência.

Capítulo 2: O Custo das Fundações Ausentes

Faz sentido dedicar um tempo para entender exatamente *quão* ruim esse problema pode ser para algumas organizações.

Descobrimos que existem três principais “categorias” de problemas que a falta de preparação cria:

- Entendimento compartilhado ausente ou incompleto entre e dentro dos times de Produto e Engenharia
- Falta de controle sobre qual é o objetivo real de um PBI e quando ele está realmente concluído
- Falta de critérios para determinar quando um item de trabalho pode iniciar a fase de implementação do seu ciclo de vida

Além disso, notamos que mudar essas coisas pode ser bastante desafiador. Isso faz sentido: mudança é difícil.

Velhos hábitos são difíceis de morrer e novos hábitos são difíceis de criar. Em nossa experiência como consultores, notamos que é *extremamente* fácil para as pessoas voltarem aos velhos hábitos e comparativamente difícil estabelecer novos.

Então você precisa ter um mecanismo que reforce os novos hábitos e desencoraje os antigos.

Para abordar isso, acreditamos que existe outro componente ausente de responsabilidade e rastreabilidade.

2.1: O Quebra-cabeça sem a Imagem da Caixa

Você já tentou montar um quebra-cabeça sem a imagem na caixa? Você consegue, mas é mais lento, mais frustrante e cheio de falsos começos.

Você faz progresso, depois desfaz tudo. Você questiona o que se encaixa onde. Você pensa que está trabalhando na mesma imagem, até perceber que não está.

É assim que o desenvolvimento de software frequentemente se parece.

O backlog está cheio. A Sprint está em andamento. Todos estão trabalhando duro.

Mas sem uma imagem compartilhada do que estamos construindo, o alinhamento se torna sorte, não um sistema.

Sem clareza, até as melhores equipes sentem frustração, esgotamento e a sensação de que seu esforço não é valorizado.

2.2: Um Velho Ditado e uma Dura Realidade

Existe um motivo pelo qual tantas práticas em torno da autoria de requisitos, até mesmo algumas práticas de engenharia, focam intensamente em criar um entendimento compartilhado entre solicitantes e equipes de implementação. Nada semeia mais caos do que engenheiros não saberem realmente o que devem construir.



“Entra lixo, sai lixo” é um ditado, não um clichê.

A linguagem está cheia de contradições...

- Você pode sancionar as ações de alguém. Em português, isso pode significar tanto aprovar oficialmente quanto punir ou penalizar.
- Você pode passar um pano em algo, que pode significar tanto limpar quanto encobrir uma situação.
- Se você diz que vai “por hoje” fazer algo, pode significar que vai fazer apenas hoje ou que vai começar a partir de hoje.

Auto-antônimos podem ser os exemplos mais marcantes, mas são apenas um tipo de ambiguidade. Algumas palavras não são apenas o seu próprio oposto, mas têm muitos outros significados potencialmente confusos.

“No banco, ele fez um depósito no banco enquanto estava sentado no banco, observando um banco de peixes passar.”

Não é só em português. *Todas* as linguagens naturais que conhecemos têm essa propriedade.

E, ainda assim, são as únicas que temos para trabalhar quando especificamos requisitos.

Como resultado, quando uma equipe de engenharia não *confirmou* que seu entendimento de um requisito é o mesmo do solicitante, essa equipe está contando com a sorte. Ou seja, o melhor resultado possível é que eles escolheram a interpretação correta e o solicitante não mude de ideia ao longo do caminho.

Esse resultado está longe de ser garantido.

2.3: Alguns dos Resultados Comuns

Sem um entendimento compartilhado confirmado, as equipes correm vários riscos.

Em geral, o resultado mais comumente doloroso é a equipe simplesmente construir a coisa errada.

O momento em que eles descobrem isso pode ser controlado por vários atributos do seu processo. Por exemplo, uma implementação saudável do Scrum pode detectar esse tipo de mal-entendido muito cedo na execução, enquanto um processo Waterfall tem uma grande chance de atrasar tal descoberta por meses.



Definição: Waterfall

Um modelo sequencial de desenvolvimento de software que se tornou difundido no final do século XX. Foi apresentado pela primeira vez em um [artigo de 1970 por Winston W. Royce](#), que ilustrava o desenvolvimento como uma série de etapas em cascata—requisitos, design, implementação, testes e assim por diante—cada uma alimentando a próxima como uma cachoeira. Embora Royce tenha apresentado o modelo como um exemplo do que *não* fazer, a indústria o adotou como um modelo para desenvolvimento em larga escala.

O Waterfall também é conhecido por agrupar trabalhos similares em grandes fases consecutivas—uma característica conhecida como **desenvolvimento em grandes lotes**. Esse agrupamento praticamente garante o **aprendizado tardio**: as equipes não recebem feedback sobre decisões anteriores até muito depois no processo. Erros descobertos tarde são mais caros de corrigir. Praticantes do Agile criticam o Waterfall por esse motivo, favorecendo ciclos iterativos menores que permitem descobertas e correções de rumo mais cedo.

No entanto, em algum momento uma equipe trabalhando com o entendimento “errado” (diferente, na verdade) de um requisito terá um acerto de contas com o requisito “certo” (também diferente, na verdade). Novamente, a saúde da organização influencia a forma que esse acerto de contas assume e qual impacto ele tem, mas isso quase sempre acontece.

Na maioria dos casos, isso leva a alguma forma de retrabalho. O solicitante (geralmente o Gerente de Produto) terá que pedir alterações para transformar o que as equipes de implementação construíram no que ele realmente queria.

Outra manifestação muito comum é o solicitante continuar responsabilizando a equipe pelo seu entendimento original do que ele pediu.

As equipes podem facilmente interpretar isso como um Gerente de Produto mudando de ideia. Pior ainda, isso pode realmente *incentivar* as partes interessadas a adquirirem o hábito de mudar de ideia - retendo itens de trabalho até que tudo esteja *perfeito*, esgotando as equipes e ocultando o progresso dos gerentes superiores.

2.4: Quando um Quebra-cabeça Está Pronto?

Voltando à nossa analogia de montagem de quebra-cabeça, pense sobre esta questão: O que significa ter terminado um quebra-cabeça?

Um montador de quebra-cabeça ingênuo, como Max, diria simplesmente “todas as peças estão conectadas aos vizinhos corretos com a imagem voltada para cima.”

Um montador de quebra-cabeça experiente, como Luniel, sabe que há mais do que isso.

Talvez você esteja apenas montando um quebra-cabeça por diversão. Você vai terminá-lo, olhar para ele por um tempinho e depois desmontá-lo e colocá-lo de volta na caixa.

Por outro lado, talvez você queira emoldurá-lo e pendurá-lo na parede. Nesse caso, há coisas adicionais que precisam ser feitas:

1. Colocá-lo em uma prancha artística
2. Transportá-lo até um emoldurador
3. Aguardar a conclusão do emolduramento
4. Transportá-lo de volta ao local de montagem
5. Pendurá-lo na parede ou colocá-lo em exposição de outra forma

Entender que isso faz parte do trabalho é necessário para concluir adequadamente a montagem de um quebra-cabeça. A razão óbvia é para que você saiba quanto trabalho está envolvido. É mais trabalho fazer todos esses passos extras do que simplesmente desmontá-lo e guardá-lo.

Mas vai além disso. Imagine este cenário...

Você completou seu quebra-cabeça com a intenção de emoldurá-lo, deixou-o no lugar, mas esqueceu de avisar alguém em sua casa que planeja emoldurá-lo. Essa pessoa aparece e vê que o quebra-cabeça está completo em um espaço que ele ou ela precisa. Então desmontam e guardam na caixa, transformando vitória em derrota no processo.

Há uma razão ainda mais sutil também: Como você planeja finalizar o quebra-cabeça impacta quais passos você quer fazer mais cedo no processo. Para começar, você precisa fazer um pequeno aviso dizendo “Por favor, não desmonte!”



Vale notar também que você pode querer um aviso no caso em que está montando um quebra-cabeça para seu próprio entretenimento para garantir que outras pessoas não interfiram completando o quebra-cabeça por você.

Você também precisa garantir que está montando o quebra-cabeça na superfície correta. Se você montar um quebra-cabeça de mil peças em sua mesa de centro de vidro e depois tentar transferi-lo para uma prancha artística, a transferência será muito mais arriscada e trabalhosa do que se você tivesse montado o quebra-cabeça diretamente na prancha artística.

Isso tem um paralelo interessante com o desenvolvimento de software.

Você precisa realmente saber o que significa estar pronto para não se surpreender com a quantidade de trabalho envolvido, para não haver desacordo sobre isso no final, e para poder tomar as medidas preparatórias necessárias para garantir a conclusão suave e eficaz de um item de trabalho.

2.5: Impacto nas Equipes

Se você não tiver um entendimento suficientemente rígido do que é a conclusão para um determinado item de trabalho, você corre vários riscos.



Usamos a palavra “risco” de forma livre aqui, já que são mais como garantias.

As equipes de engenharia nessa situação frequentemente descobrem que nem mesmo internamente concordam sobre como é a conclusão de um item de trabalho. Não é incomum que programadores e testadores se vejam discutindo o que um requisito realmente *significa* no meio de uma Sprint. Até mesmo dois programadores ou dois testadores podem sofrer com esses mesmos desentendimentos.

Além disso, as equipes de desenvolvimento frequentemente estão focadas no trabalho que fazem na maior parte do tempo (programação e testes). Isso significa que é fácil para elas esquecerem outros tipos de trabalho que precisam fazer, como documentação, revisões externas, treinamento de outras equipes (*por exemplo*, suporte), etapas preparatórias para dar suporte à implantação ou liberação, e aprovações de outros departamentos.

Quando finalmente fica claro que este trabalho “extra” precisa ser feito, eles são pegos de surpresa—geralmente têm que parar o que estavam fazendo e mudar de contexto para voltar e completar um trabalho que pensavam já ter terminado.

Os solicitantes do trabalho podem facilmente manter o trabalho em aberto desnecessariamente. Às vezes com as melhores intenções—como tentar manter uma equipe responsável pelo requisito “real”. Outras vezes, isso acontece porque Product Owners (por exemplo) se acostumam a poder manter um PBI aberto ao seu critério, então o usam para inserir funcionalidades adicionais em um item no último minuto. Às vezes, fazem isso porque mudaram de ideia sobre o que precisa ser feito durante o processo.

Isso pode ser extremamente desmoralizante para uma equipe de engenharia. A maioria dos desenvolvedores e testadores de software quer sentir que está progredindo. Se estão constantemente sendo informados que o que fizeram estava errado, é provável que percam parte de seu vigor.

Algumas equipes chegam ao ponto de nem mesmo se preocupar em verificar se o que fizeram estava certo ou errado. Elas simplesmente fecham um item de trabalho e pedem “crédito” para “apresentar bons números”.

2.6: O Risco de Fazer Muito Pouco ou Muito

Um risco de não definir adequadamente o “pronto” para cada item de trabalho é a organização pensar que o trabalho está concluído quando não está, ou não perceber que está concluído quando realmente está.

O pior resultado possível é frequentemente quando a coisa errada chega à produção e ninguém sabe que isso aconteceu. Se a equipe tem um entendimento incorreto do

que “pronto” significa e entrega baseado nesse mal-entendido, os resultados podem ser catastróficos.

Defeitos e insatisfação do cliente já são ruins o suficiente, mas isso também pode levar a problemas muito mais sérios:

- Perda ou corrupção de dados
- Vulnerabilidades de segurança
- Tempo de inatividade do sistema ou perda de acesso
- Redução na participação de mercado
- Violações regulatórias

A lista continua indefinidamente e cada problema potencial é pior que o anterior.

Às vezes o problema é que você não terminou, mas acha que sim. O contrário pode ser igualmente perigoso. Quando os engenheiros não sabem onde está a linha de chegada, eles tendem a fazer “sobrecarga de funcionalidades” (adicionar recursos extras). Eles podem estar fazendo isso para “deixar o recurso mais agradável”, mas também podem estar fazendo isso porque esperam que um número maior de funcionalidades lhes dê uma chance maior de atingir o objetivo.

Todo esse trabalho extra, bem como o retrabalho correspondente, acumula-se em uma quantidade massiva de tempo, esforço e dinheiro desperdiçados. Isso causa atrasos nas datas de entrega e prejudica reputações.

Além disso, agora, há um potencial cada vez maior de um erro realmente levar a uma revolta estilo *Terminator* das máquinas contra a humanidade. Costumávamos escrever sobre isso há vinte anos como uma piada. Agora, é uma possibilidade remota.

Na verdade, perguntamos a uma das IAs mais proeminentes sobre isso, e eis o que ela disse:

“A IA está se expandindo mais rápido do que qualquer um esperava, mas está fazendo isso sobre sistemas frágeis, requisitos vagos e organizações de produto que não conseguem rastrear por que construíram o que construíram. Isso não é um problema técnico; é um problema de clareza.

Quanto mais ruído a IA gera, mais perigoso é avançar rapidamente sem estrutura. Quando as equipes constroem sobre névoa, tudo que a IA faz é amplificar a bagunça. Mas quando as equipes constroem sobre

sinal—sobre entendimento compartilhado, especificidade comportamental e conhecimento de produto versionado—a IA se torna um acelerador em vez de um passivo.“

2.7: Onde Você Monta um Quebra-cabeça?

Vamos estender a analogia do quebra-cabeça mais uma vez.

Você pode montar um quebra-cabeça em qualquer lugar? Se você tem um quebra-cabeça de 4.000 peças que monta quase um metro e meio em uma dimensão e mais de um metro em outra, você não pode simplesmente escolher um local aleatório e começar a montar. Não sem experimentar sérias complicações antes de terminar.

Um quebra-cabeça grande como esse precisa de tempo e espaço. Você precisa alocar o espaço e encontrar uma maneira de garantir que o estado do quebra-cabeça seja mantido ao longo do tempo.

Se você começar a montar seu quebra-cabeça em uma mesinha lateral que é muito pequena, você não conseguirá terminá-lo sem transferi-lo para outro local. Essa transferência será extremamente difícil devido ao estado delicado do quebra-cabeça.

Se você escolher um lugar aleatório no corredor que seja grande o suficiente, as pessoas vão ou andar sobre ele ou ser impedidas, então a persistência não pode ser garantida sem um impacto significativo no funcionamento da sua casa.

Se você começar a trabalhar em uma prancha de arte, mas a prancha não for grande o suficiente, você conseguirá preservar o estado do que já fez, mas não conseguirá completar o quebra-cabeça sem algum tipo de transferência.

Se o quebra-cabeça já foi mastigado por crianças pequenas, é melhor contar as peças... porque é melhor contar até 3999 uma vez e perceber que você nunca vai terminar do que investir sabe-se lá quanto tempo quase montando um quebra-cabeça que você nunca conseguirá finalizar.

Existe toda uma lista de coisas que precisa ser feita antes de você começar a montar seu quebra-cabeça. Fazer as coisas da lista não garante o sucesso, mas *não* fazê-las praticamente garante o fracasso ou complicações sérias.

O mesmo vale para o desenvolvimento de software, mas com um grau maior de complexidade.

2.8: Quando Começa a Implementação?

Fundamentalmente, pode ser difícil determinar quando um PBI está pronto para ser implementado sem uma boa definição para isso.

Pense bem: Como você *realmente* sabe?

Você revisa tudo repetidamente até decidir que é hora?

Alguém decide por impulso?

Acontece automaticamente no início de uma iteração?

Vimos muitas equipes empurrarem trabalho para Sprints que não estavam nem perto de estar prontos para serem implementados apenas porque tinham prazos. Existem algumas ideias persistentes sobre Scrum e Agile em geral que levam as pessoas a fazer isso:

- Você deve obter todos os seus requisitos para a Sprint *N* desenvolvidos na Sprint *N-1*
- Você deve “simplesmente começar” e lidar com o que quebrar ao longo do caminho

Isso é na verdade um espelho da questão “Como você sabe quando está pronto?” [mencionada anteriormente](#) e tem consequências similares. As pessoas podem esperar muito tempo para começar, porque não sabem que um item de trabalho está pronto, e podem começar muito cedo porque não sabem que não está.

2.9: Uma Equipe A sem Prontidão

Não entender o que é necessário para um item de trabalho estar pronto tem vários efeitos prejudiciais.

Uma maneira óbvia de um incremento de trabalho não estar pronto é ter uma Definição de Pronto (DoD) incompleta, insuficiente ou ausente. Isso leva a todos os problemas que já citamos que acompanham a falta de uma Definição de Pronto.

No entanto, esse não é o *único* aspecto da prontidão. Existem várias outras necessidades a serem satisfeitas antes que a implementação comece: estimativa, avaliação de risco e coleta de dados de teste são apenas alguns exemplos comuns.

Sem conhecer e satisfazer essas necessidades, um item de trabalho pode custar muito mais do que precisaria custar. Considere uma equipe (A Equipe A) que depende de uma API sendo desenvolvida por outra equipe (a Outra Equipe). Se a Equipe A fizer um monte de suposições sobre como a API da Outra Equipe vai funcionar e programar baseando-se nessas suposições, pode haver um retrabalho significativo quando descobrirem que a Outra Equipe estava trabalhando em uma realidade que não correspondia às suposições da Equipe A. Em outras palavras, a Equipe A apostou e errou.

Todo esse retrabalho deriva do fato de que a API não estava pronta para ser usada pela Equipe A.

Às vezes uma dependência não satisfeita não gera retrabalho, mas, mesmo nesses casos, ainda pode causar atrasos. Imagine se a Equipe A e a Outra Equipe concordassem sobre como a API deveria funcionar e tudo corresse conforme planejado, mas a Outra Equipe simplesmente demorasse mais do que o esperado. Como resultado, a Equipe A simplesmente não conseguiu testar seu trabalho adequadamente até o momento em que deveria estar concluído e eles tiveram que adiar seu prazo.

2.10: Falha em Atender ao Agendamento e Disponibilidade de Recursos

Às vezes os problemas podem ser tão simples quanto agendamento ou recursos. Alguns itens de trabalho precisam de membros específicos da equipe. Se esse membro da equipe vai tirar férias em alguns dias, provavelmente não é o momento certo para começar o PBI que não pode ser concluído sem a participação dele.

Frequentemente ouvimos pessoas dizerem que não deveria ser assim, mas frequentemente é, independentemente disso. “Pessoas intercambiáveis” é uma ilusão.

O mesmo pode ser dito dos recursos não humanos. Se você vai precisar de recursos do servidor para realizar um teste de carga, provavelmente deve garantir que esses recursos estarão realmente disponíveis antes de começar o item de trabalho do teste de carga. Caso contrário, no melhor cenário, haverá atrasos significativos e você provavelmente irá atrapalhar outras equipes/trabalhadores enquanto tenta providenciar às pressas o que precisa.

Outro modo de falha dos falsos inícios é quando uma equipe não possui as habilidades necessárias para completar o trabalho. Às vezes, é uma questão interna—como quando um membro da equipe precisa receber treinamento em um novo sistema ou fazer pesquisas sobre uma nova API. Outras vezes, é uma questão de agenda, como quando você precisa emprestar um especialista em UX ou banco de dados de um grupo de profissionais especializados. Pode até ser um problema de contratação, no qual a equipe precisa de um especialista e não consegue efetivamente completar certos tipos de trabalho sem ele.

2.11: Impactos de Outros Tipos de Falsos Inícios

Vimos equipes se comprometerem a completar itens de trabalho dentro das Sprints e fazer a codificação relativamente rápido, mas ainda assim serem incapazes de completar os testes. Isso, por si só, pode não ser surpreendente, mas a razão é incomum: a equipe de testes tinha algo que precisava (como dados de teste) que não havia coletado antes do início da Sprint, e a coleta desses dados acabou sendo mais difícil ou demorada do que eles anteciparam.

Como resultado, os itens de trabalho tiveram que ser transferidos para a próxima Sprint simplesmente porque a equipe não havia garantido que estava verdadeiramente pronta para completá-los dentro do tempo determinado antes de começar.

As equipes às vezes começam a implementar itens de trabalho quando ainda têm questões em aberto. Na verdade, muitas pessoas parecem pensar que isso as torna “mais Ágeis” quando fazem isso.

Isso pode causar uma enorme quantidade de retrabalho, surpresas ou atrasos. Se a resposta à questão em aberto acabar violando uma premissa que foi feita, todo o trabalho baseado nessa premissa precisa ser revisado. Se a questão em aberto não for respondida até o momento em que o item deveria ser fechado, então o item tem que ser fechado quando pode não estar pronto, ou mantido em aberto até que a questão seja respondida.

Pode ser que a equipe tenha uma dependência interna—um defeito que precisa ser corrigido, uma tarefa predecessora que deve ser completada, *et cetera*. Se isso não for rastreado adequadamente, pode causar todos os mesmos problemas que uma dependência externa não satisfeita, com a tentativa adicional de mudar o contexto e corrigi-lo.

2.12: Custos Cumulativos

Naturalmente, tais problemas criam atrasos, retrabalho e expectativas frustradas, mas os prejuízos não param por aí.

Além do desperdício com retrabalho, isso geralmente faz os projetos atrasarem. Se as equipes estão freneticamente tentando fechar itens do backlog e nunca realmente claras sobre o que será necessário para fazer algum progresso real, as coisas que realmente *precisam* ser feitas tendem a ficar de lado.

Frequentemente, embora nem sempre, isso leva a uma pressão aumentada para entregar. À medida que os projetos ficam cada vez mais atrasados, a alta administração pode tentar colocá-lo de volta nos trilhos pedindo às pessoas para irem mais rápido. Isso invariavelmente se traduz em trabalhar mais horas.

Isso, por sua vez, tende a erodir a confiança e amargar a cultura de uma organização. Relacionamentos que deveriam ser colaborativos se tornam adversariais. Pessoas que deveriam estar trabalhando juntas para encontrar as melhores e mais rápidas soluções, desviam energia para estabelecer que, quando as coisas inevitavelmente dão errado, não foi culpa delas.

Na busca desenfreada por funcionalidades e itens de trabalho fechados, as equipes frequentemente se encontram cortando caminho. Isso realmente significa que elas estão deixando a qualidade (especialmente a qualidade do código) sofrer. Isso, por sua vez, significa que elas estão trocando a produtividade futura pela ilusão de progresso no presente.

À medida que as condições de trabalho se tornam cada vez mais desagradáveis, os talentos-chave começam a se desengajar ou até mesmo a procurar outras oportunidades.

Organizações que se comportam dessa maneira estão “comendo o milho que deveria ser semente”, por assim dizer, de várias maneiras. A base de código se torna menos manutenível e as pessoas que deveriam mantê-la são todas afastadas.

Se há um lado positivo, ele é invisível para nós.

Capítulo 3: Introduzindo o Requirements Maturation Flow (RMF)

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

3.1: O que o RMF não é

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

3.2: O que o RMF é

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

3.3: A Adoção Incremental é Suportada e Recomendada

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

3.4: FMR 1

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

3.5: FMR 2

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

3.6: FMR 3

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 4: É Ágil?

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

4.1: “Indivíduos e Interações”, “Software em Funcionamento”

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

4.2: Colaboração com o Cliente

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

4.3: Respondendo a Mudanças

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

4.4: Transparência

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

4.5: Se Encaixa no Processo, Consistente com o Ágil

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Parte II: Criando Espaço para a Prontidão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 5: A Primeira Extensão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

5.1: Trabalho de Preparação é *Trabalho*

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

5.2: Naturalizando o Trabalho de Preparação

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

5.3: Um Incidente Ilustrativo

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

5.4: Impacto Recíproco

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

5.5: A Função do RMF 1

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 6: Por Que as Pessoas Não Fazem Isso?

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

6.1: Trabalho de Preparação como Cidadão de Segunda Classe

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

6.2: Uma Alergia ao Trabalho Não Produtivo

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

6.3: Então Foi Enterrado

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

6.4: A Influência do Gerenciamento de Projetos

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

6.5: O Padrão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

6.6: Projetos e Estimativas

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

6.7: Como as Não-Estimativas Influenciam o Trabalho de Preparação

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

6.8: Medindo Velocidade, Não Direção

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

6.9: Medições Ruins, Resultados Ruins

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

6.10: Onde Não Está a Culpa

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 7: Trabalho Explícito de Prontidão (RMF 1)

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

7.1: Integração com o Synapse Framework™

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

7.2: Anatomia do RMF 1

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

7.3: Comportamento: Reservar Capacidade para Colaboração

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

7.4: Artefato: O Item de Trabalho de Prontidão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

7.5: Atividade: A Reunião de Colaboração

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

7.6: Comportamento: Continue Colaborando Até Alcançar o Entendimento Compartilhado

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

7.7: Comportamento: Sempre Confirme o Entendimento Compartilhado

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

7.8: Como o RMF 1 Muda o Fluxo de Trabalho

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 8: Efeitos do RMF 1

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

8.1: A Vida Antes do RMF 1

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

8.2: Antes: Tempo Gasto com Compreensão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

8.3: Depois: Tempo Gasto com Entendimento

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

8.4: A Vida Após Adotar o RMF 1

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

8.5: Fundamental

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 9: Colocando o RMF 1 em Prática

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

9.1: Educação

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

9.2: Requisitos Mínimos por Tipo de Equipe

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

9.3: Concordância

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

9.4: Preparação

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

9.5: Piloto

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

9.6: Implantação

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

9.7: Acompanhamento

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

9.8: Declarando Sucesso

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

9.9: Mantendo a Vigilância

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

9.10: E quanto ao “Como”?

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

9.11: Hora de Fazer Acontecer!

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Parte III: Controlando a Conclusão do Trabalho

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 10: A Próxima Necessidade

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

10.1: Margem para Interpretação

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

10.2: Reduzindo a Margem para Interpretação

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

10.3: Uma Terceira Opção: Sem “Margem de Manobra”

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

10.4: Impacto Potencial na Conclusão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

10.5: Impacto Potencial na Execução

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

10.6: Alternativa Proposta: Não Deixe Espaço para Interpretação Errada

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

10.7: Benefícios

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

10.8: Sobre os Medos da Paralisia por Análise

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

10.9: A Próxima Necessidade: Definições de Pronto Personalizadas

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 11: O Que as Pessoas Geralmente Fazem

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

11.1: Se É Tão Bom, Por Que as Pessoas Não Fazem Isso?

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

11.2: O Pipeline da Faculdade para o Coaching

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

11.3: Sobrecarga de Coaching

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

11.4: Uma Maneira Como as Pessoas Fazem DoD: Não Fazem

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

11.5: Apenas Critérios de Aceite

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

11.6: Definição de Pronto Global

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

11.7: Sem Força

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

11.8: Resumo: O Termo “Definição de Pronto” É Mais Frequente que Definições de Pronto Reais

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 12: Definindo uma Definição de Pronto

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

12.1: Sobre Apenas Um Item de Trabalho

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

12.2: Estado de Conclusão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

12.3: Precisão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

12.4: Estrutura de uma Definição de Pronto

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

12.5: Especificações

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

12.6: Critérios de Saída da Engenharia

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

12.7: Critérios de Entrada do Produto

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

12.8: Múltiplas Partes, Uma Porta de Controle

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

12.9: Exemplo

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

12.10: Mapeando para Seu Processo

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

12.11: Resumo

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 13: Definition of Done Personalizada (RMF 2)

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

13.1: Princípio: Cada Item de Trabalho é Único

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

13.2: Comportamento: Manter Um ou Mais Modelos de DoD

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

13.3: Atividade: Definindo o Modelo de Definição de Pronto

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

13.4: Manter e Melhorar o Modelo de Definição de Pronto

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

13.5: Múltiplos Modelos de Definição de Pronto

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

13.6: Comportamento: Use Modelos como Pontos de Partida para Definições de Pronto

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

13.7: Comportamento: Concordar com Definições de Pronto Personalizadas

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

13.8: Atividade: Definindo a Definição de Pronto de um Item de Trabalho

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

13.9: Outra Extensão do Fluxo de Trabalho

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

13.10: Comportamento: Amadurecer a DoD antes de Iniciar a Implementação

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

13.11: Atividade: Análise Offline para Amadurecer o DoD de um PBI

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

13.12: Adicionando Maturação ao Fluxo

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

13.13: Comportamento: Acompanhamento de Conclusão em Itens de Trabalho

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

13.14: Adicionando Acompanhamento de Progresso

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

13.15: Comportamento: Condicionar o Trabalho à Conclusão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

13.16: Atividade: Usando a DoD para Determinar a Conclusão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

13.17: Como o Controle se Encaixa

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

13.18: Em Resumo

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 14: A Vida Com RMF 1 e 2

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

14.1: Custo

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

14.2: Prazos

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

14.3: Impacto na Equipe de Implementação

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

14.4: Impacto sobre a Product Owner

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

14.5: Impacto sobre a Liderança

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

14.6: Resumo

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 15: Instalando o RMF 2

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

15.1: Envolvimento das Partes Interessadas

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

15.2: Nível de Detalhe

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

15.3: Acordo de Trabalho

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

15.4: Trabalho Inicial

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

15.5: Implantação

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

15.6: Resumo

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Parte IV: Implementação de Gates

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 16: O Requisito Final

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

16.1: Estava Lá o Tempo Todo

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

16.2: O Poder do Timing

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

16.3: Invertendo a Lógica

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

16.4: Riscos & Custos

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

16.5: O Valor de Esperar Até Estar Pronto

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

16.6: Um Benefício Adicional

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

16.7: A Declaração do Problema

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

16.8: A Necessidade

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

16.9: Resumo

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 17: Contexto sobre a Definição de Pronto para Início

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

17.1: Permissão para Trabalhar do Lean

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

17.2: Critérios de Entrada de Coluna do Kanban

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

17.3: Scrum e outros Apócrifos de Processos Ágeis

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

17.4: Surfar > Programar

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

17.5: Soluções Pontuais

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

17.6: Estamos Prontos para Ficar Prontos

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 18: Definindo uma Definição de Pronto para Início

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

18.1: Propósito

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

18.2: Sob Medida

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

18.3: Anatomia

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

18.4: Exemplo

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

18.5: Acordo

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

18.6: Controle de Passagem

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

18.7: Resumo

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 19: Definição de Pronto para Início Personalizada (RMF 3)

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

19.1: Outro Portão no Processo

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

19.2: A Estrutura de uma Definição de Pronto para Início

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

19.3: Critérios de Saída do Produto

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

19.4: Critérios de Entrada da Engenharia

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

19.5: Não Há Problema em Duplicação

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

19.6: Comportamento: Manter Um ou Mais Modelos de DoR

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

19.7: Por Que Ter um Modelo de Definição de Pronto para Início?

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

19.8: Atividade: Definindo o Modelo de DoR

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

19.9: Mantenha os Modelos de DoR ao Longo do Tempo

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

19.10: Modelos São Apenas um Ponto de Partida

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

19.11: Comportamento: Concordar com Definitions of Ready Sob Medida

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

19.12: Atividade: Definindo um Definition of Ready para um Item de Trabalho

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

19.13: Comportamento: Tornar Itens Prontos antes de Iniciar a Implementação

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

19.14: Condições Fora do Controle da Equipe

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

19.15: Comportamento: Acompanhar a Prontidão em Itens de Trabalho

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

19.16: Comportamento: Controlar o Trabalho pela Prontidão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

19.17: Atividade: Usando o DoR para Determinar a Prontidão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

19.18: Em Resumo

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Parte V: Síntese

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 20: A Maioria dos Prazos Não Importa

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

20.1: Prazos Reais

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

20.2: Prazos Arbitrários Não Importam

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

20.3: Companhias Aéreas

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

20.4: A Origem da Dívida Técnica

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

20.5: Prazos de Marketing e Vendas

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

20.6: Prazos de Projeto

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

20.7: Existe Outra Maneira

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

20.8: Prazos Arbitrários Não São Necessários

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

20.9: Prazos Arbitrários Devem Ser Abolidos

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

20.10: Prazos Reais Ainda São um Fator

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

20.11: *Pis Aller*¹

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

20.12: Conclusão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

¹Um movimento de último recurso, originalmente da língua francesa.

Capítulo 21: Competência 1: Fluxo de Maturação de Requisitos

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

21.1: Como Abaixo, Assim Acima

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

21.2: Princípio: Transparência em Todo o Trabalho Necessário

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

21.3: Comportamento: A Responsabilidade Viaja com o Trabalho

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

21.4: Comportamento: Acompanhe Visivelmente o Estado dos Requisitos

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

21.5: Comportamento: Revelar Todo o Trabalho Associado à Prontidão e Implementação

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

21.6: Atividade: Trabalho de Prontidão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

21.7: Comportamento: Revelar Todo o Trabalho Necessário para Completar um Item de Trabalho

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

21.8: Comportamento: Prefira Prontidão a Prazos

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

21.9: Conclusão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 22: Como o Trabalho e as Informações Fluem no Scrum com RMF

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

22.1: Notas Preliminares

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

22.2: Capturando e Preparando o Requisito Inicial

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

22.3: Iniciando o Trabalho de Prontidão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

22.4: Planejando e Executando o Trabalho de Prontidão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

22.5: Revisando os Resultados da Prontidão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

22.6: Planejando e Completando a Implementação

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 23: O Impacto do RMF

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

23.1: O Ciclo de Vida de um Requisito

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

23.2: Exemplo de Fluxo de Informação e Trabalho

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

23.3: Antes

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

23.4: Depois

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

23.5: Benefícios

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 24: Transição para RMF

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

24.1: Padrão de Adoção

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

24.2: Instalação em um Fluxo de Trabalho Scrum

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

24.3: Outros Frameworks

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

24.4: Principal Resistência: Itens de Trabalho de Prontidão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

24.5: A Grande Mudança: Mentalidade

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

24.6: Conselhos para a Mudança

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

24.7: Conclusão

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Capítulo 25: Depende de Você

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

25.1: Recapitulando

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

25.2: Agora É Sua Vez

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Parte VI: Recursos

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Apêndice A: O Scrum Não É o Problema

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

O que é Scrum?

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Frameworks

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

O Scrum Aborda o Gerenciamento de Projetos e Trabalho

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

As Tolices de Tratar o Scrum Como um Framework de Gestão de Produtos

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Sem Mecanismo Prescrito para Amadurecer Requisitos

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Sem Investimento de Expertise em Engenharia no Desenvolvimento de Requisitos

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Antipadrões

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Extensão Necessária

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Adicionando ao Scrum

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Apêndice B: O Synapse Framework™

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

O que o Synapse Abrange

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

As Três Maestrias

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Como o Synapse É Adotado

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Unindo Dois Frameworks

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

A Estrutura do Framework Synapse

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Maestrias Organizacionais

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Competências Organizacionais

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Hábitos Organizacionais

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Estrutura do Framework Synapse

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Anatomia de uma Práxis

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

A Importância da Ordem

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

O Impacto do Synapse neste Livro

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Apêndice C: Objeções e Obstáculos Comuns ao RMF 1

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Objeções Comuns

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Obstáculos Comuns

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Apêndice D: Listas Base de Critérios DoD

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Lista Base de Critérios de Saída de Engenharia

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Lista Inicial de Critérios de Entrada do Produto

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Apêndice E: Listas Iniciais de Critérios para DoR

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Critérios de Saída do Produto

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Critérios de Entrada para Engenharia

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Critérios de Entrada da Sprint

Este conteúdo não está disponível na amostra do livro. O livro pode ser adquirido no Leanpub em <https://leanpub.com/ready-pt-BR>.

Índice

- A Grande Virada, 13
- A Maioria dos Prazos Não Importa, 66
- Acompanhamento de Conclusão, 48
- Algo Está Faltando, 1
- Amadurecer a DoD, 47
- ambiente de desenvolvimento, 10
- API, 24
- atrasados, 26
- auto-antônimos, 16
- autoria de requisitos, 16
- bancos, 10
- Barreras, Tom, vii
- base de conhecimento, 8
- base de código, 26
- Behavior-Driven Development, 7
- ciclos de feedback, 13
- colaboradores individuais, 5
- Como você sabe quando está pronto?, 23
- Comportamento: Acompanhe Visivelmente o Estado dos Requisitos, 69
- Comportamento: Revelar Todo o Trabalho Associado à Prontidão e Implementação, 69
- conclusão do item de trabalho, 20
- confiança, 26
- consultores, 8
- contar as peças, 22
- Criando Espaço para a Prontidão, 30
- CrITÉrios de Entrada da Engenharia, 61
- CrITÉrios de Saída do Produto, 61
- cultura, 10
- Custos Cumulativos, 25
- código existente, 11
- Declarando Sucesso, 38
- defeito, 25
- definindo 'pronto', 20
- Definition of Done do Item do Backlog do Produto, 48
- Definitions of Done personalizadas, 46
- Definição de Pronto, 23, 47
- Definições de Pronto para Início Personalizadas, 61
- Definições Personalizadas de Preparado, iii
- desenvolvedores de software, 20
- desenvolvimento de software, 16, 19, 22
- desenvolvimento de testes, vii
- desenvolvimento em grandes lotes, 17
- Desenvolvimento Orientado a Comportamento, iv
- Desenvolvimento Orientado a PKB, iv
- desperdício, 26
- domínio, 10
- doutrinas de programação de meados do século XX, 7
- dívida técnica, 66
- entendimento compartilhado, 35
- entendimento compartilhado confirmado, 17
- Entra lixo, sai lixo, 16
- equipe de liderança, 6
- equipes de engenharia, 9, 10
- especialista no assunto, 6, 11
- Estrutura de uma Definição de Pronto para Início, 61
- executivos de engenharia, 4
- finanças, 10
- fluxo, 13
- Fluxo de Maturação de Requisitos, i, iii, iv, 68,

- fluxo de trabalho, 35, 47
- habilidades da equipe, i
- IA, 21
- Implementação de Gates, 54
- infraestrutura financeira nacional dos Estados Unidos, 2
- Initial Work, 52
- Instalação em um Fluxo de Trabalho Scrum, 73
- Instituição de Crédito Federada, 2
- item de trabalho, 25
- Item de Trabalho de Prontidão, 34
- Item do Backlog do Produto, 12, 15, 23
- itens do backlog, 26
- iteração, 23
- Kanban, i
- Level of Detail, 52
- liderança, 10
- linguagens naturais, 17
- Marketing e Vendas, 66
- montagem de quebra-cabeça, 22
- mudanças durante o Sprint, 6
- O Banco, 13
- Oregon Central, vii
- Organização de Produto, 4
- Outros Frameworks, 73
- OutSystems, 3
- Padrão de Adoção, 73
- padrões de projeto, iv
- participação de mercado, 21
- Perda ou corrupção de dados, 21
- Pessoas intercambiáveis, 24
- portal de pagamento de empréstimos, 3
- Portão no Processo, 61
- prazos, 23
- Prazos de Projeto, 67
- premissa, 25
- Principal Resistência, 73
- Procore, iii
- Product Backlog Item, 7, 10
- Product Manager, 5
- Product Owner, 5, 10, 20
- Product, função do, 4
- Produto e Engenharia, 15
- Project Management Institute, 2
- prontos*, 12
- práticas de engenharia, 10, 16
- qualidade do código, 26
- Ready, i, ii
- Recursos, 76
- recursos do servidor, 24
- Requirements Maturation Flow, 27
- requisito real, 20
- requisitos errados, 18
- responsabilidade e rastreabilidade, 15
- retenção de talentos, 26
- retrabalho, 18
- revolta estilo Terminator, 21
- riscos, 19
- RMF 1, 31, 81
- RMF 2, 52
- Rollout, 52
- Scrum, 5, 17, 23, 70
- serviço upstream, 12
- serviços dependentes, 10
- sistema, 9
- sistema legado, 6
- sobrecarga de funcionalidades, 21
- sobrecarregar, 8
- solução C#/.NET, 3
- solução low-code, 3
- Sprint, 11, 16, 20, 23, 25
- Stakeholder Involvement, 52
- stakeholders, 18
- Tempo de inatividade do sistema, 21
- tempo desperdiçado, 21

- Testadores, 12
- Time de Desenvolvimento, 20
- Trabalho Explícito de Prontidão, 34
- trabalho extra, 20
- transformando vitória em derrota, 19
- transformações ágeis, iv
- Transição para o Fluxo de Maturação de Requisitos, 73
- técnica de elaboração de requisitos, 10
- técnicas de desenvolvimento de software, 7
- Violações regulatórias, 21
- Vulnerabilidades de segurança, 21
- Waterfall, 17
- Working Agreement, 52
- Ágil, 25