ROGADX 2026

ADORA: os 5 superpoderes do Vibe Coding

que vão mudar como você desenvolve software

Vinicius Mendes • RogaDX • Agosto de 2026

Vinicius Mendes

- Pai de Sarah e Gisele
- Natalense

HOJE

- **Staff Engineer** na Loadsmart
- 20+ anos desenvolvendo software
- Membro da APyB

EX

- Dataprev
- Globo.com / G1
- UFRN

O que é Vibe Coding?

O que é Vibe Coding?

Vibe Coding viralizou recentemente e descreve **um novo jeito de criar software**: desenvolver um aplicativo inteiro apenas conversando com IA, sem digitar código manualmente.

O programador deixa de ser um "digitador de código" e passa a atuar como **diretor** ou **arquiteto**. Você dita o ritmo, as ideias e o visual (a "vibe"), enquanto a IA cuida de toda a parte técnica, da sintaxe e da correção de bugs.

Sintetizado a partir do texto gerado por IA

COMO VERIFICAR?

Me dê referências com links que suportem essa definição para que eu possa verificar.

- Publicação de Andrej Karpathy
- Estudos científicos
- Livro de Vibe Coding

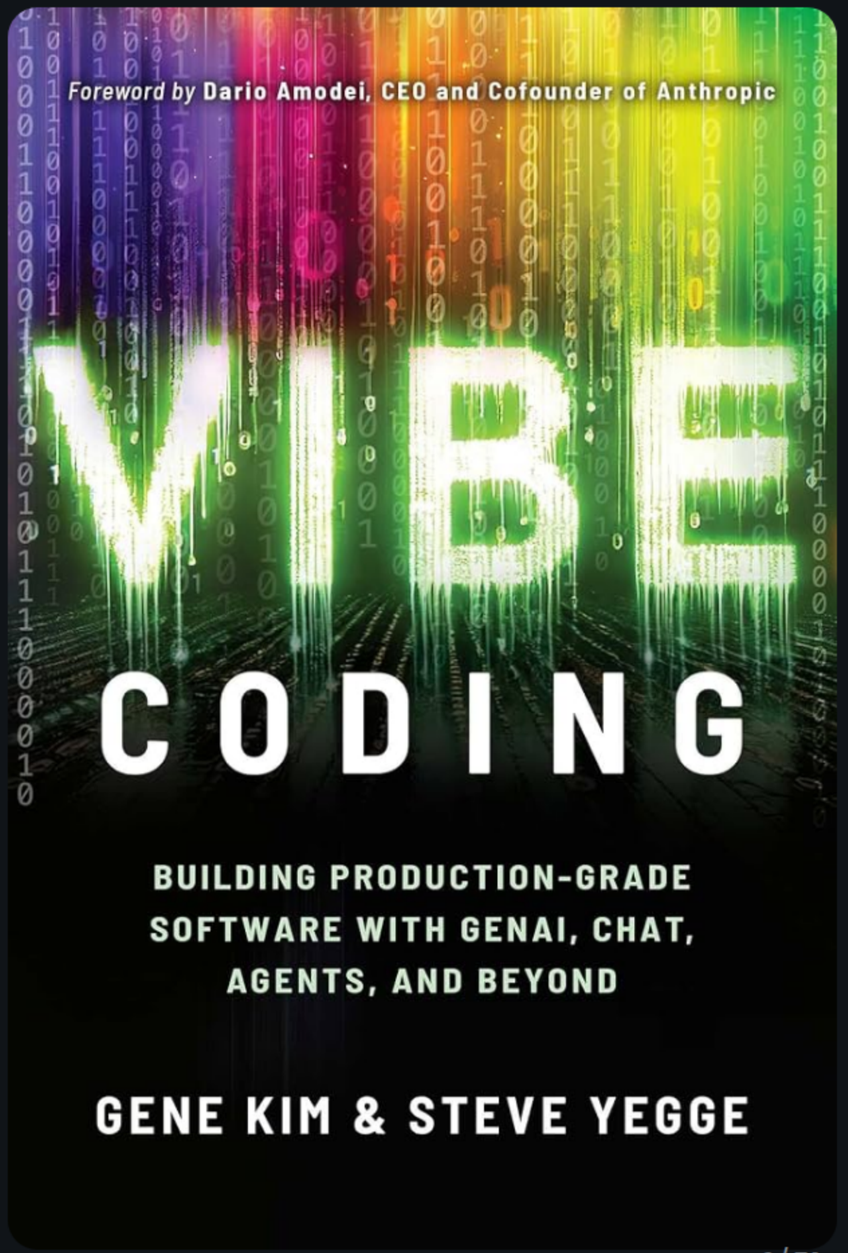
Sintetizado a partir do texto gerado por IA

"

Vibe Coding acontece sempre que você está direcionando e não digitando, permitindo que a IA cuide da implementação enquanto você foca em **visão e verificação**.

Gene Kim e Steve Yegge

Traduzido do livro "Vibe Coding: Building Production-Grade Software with GenAI, Chat, Agents, and Beyond"



Por que muitas pessoas não gostam de usar esse termo?

- Hype
- Soa superficial
- Diminui o trabalho técnico
- Falsa sensação de facilidade
- Parece irresponsável

Sintetizado a partir do texto gerado por IA

"

Existe um novo tipo de programação que eu chamo de 'Vibe Coding', onde você se entrega completamente à vibe, abraça exponenciais e esquece que o código sequer existe. [...]

Andrej Karpathy — ex-Tesla, cofundador da OpenAI

Traduzido da publicação de Fevereiro de 2025 — x.com/karpathy/status/1886192184808149383

"

Eu 'Aceito tudo' sempre, eu não leio mais diffs [...]

O código cresce além da minha compreensão [...]

[...] eu apenas contorno o bug e peço mudanças aleatórias até que ele suma.

Andrej Karpathy — ex-Tesla, cofundador da OpenAI

Traduzido da publicação de Fevereiro de 2025 — x.com/karpathy/status/1886192184808149383

Andrej Karpathy
@karpathy

There's a new kind of coding I call "vibe coding", where you fully give in to the vibes, embrace exponentials, and forget that the code even exists. It's possible because the LLMs (e.g. Cursor Composer w Sonnet) are getting too good. Also I just talk to Composer with SuperWhisper so I barely even touch the keyboard. I ask for the dumbest things like "decrease the padding on the sidebar by half" because I'm too lazy to find it. I "Accept All" always, I don't read the diffs anymore. When I get error messages I just copy paste them in with no comment, usually that fixes it. The code grows beyond my usual comprehension, I'd have to really read through it for a while. Sometimes the LLMs can't fix a bug so I just work around it or ask for random changes until it goes away. It's not too bad for throwaway weekend projects, but still quite amusing. I'm building a project or webapp, but it's not really coding - I just see stuff, say stuff, run stuff, and copy paste stuff, and it mostly works.

8:17 PM · Feb 2, 2025 · **7.3M** Views

Quais são os outros termos que essas pessoas preferem?

- Prompt Engineering
- AI-assisted Engineering / Programming / Development
- AI-driven [...]
- Agentic [...]
- Harness [...]

Sintetizado a partir do texto gerado por IA

Como estes termos se diferenciam de Vibe Coding?

A principal diferença está na percepção de **responsabilidade técnica**.

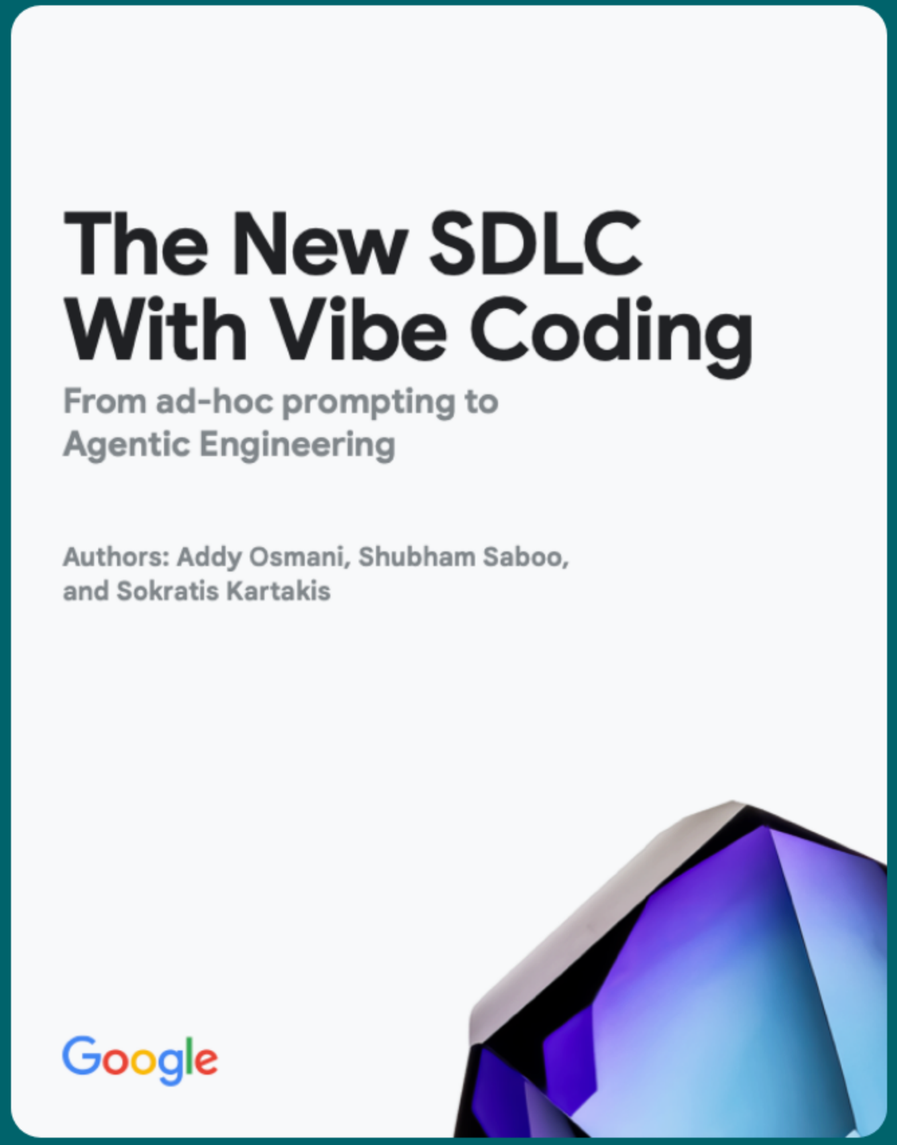
- Compreensão
- Verificação
- Disciplina de engenharia

Sintetizado a partir do texto gerado por IA

"

A diferença não é se você usa IA — é **como os resultados são verificados**.

The New SDLC With Vibe Coding — Kaggle, 2026



Vibe coding

Prompts casuais
"Será que funciona?"
Código descartável

Structured AI-Assisted

Prompts detalhados + restrições
Teste manual + checagens pontuais
Features em bases de código existentes

Agentic engineering

Specs formais + docs de arquitetura
Evals automatizados + gates de CI/CD
Sistemas de produção em escala

← menos estrutura, mais velocidade

mais estrutura, mais confiabilidade →

adaptado do paper "The New SDLC With Vibe Coding", Kaggle, 2026

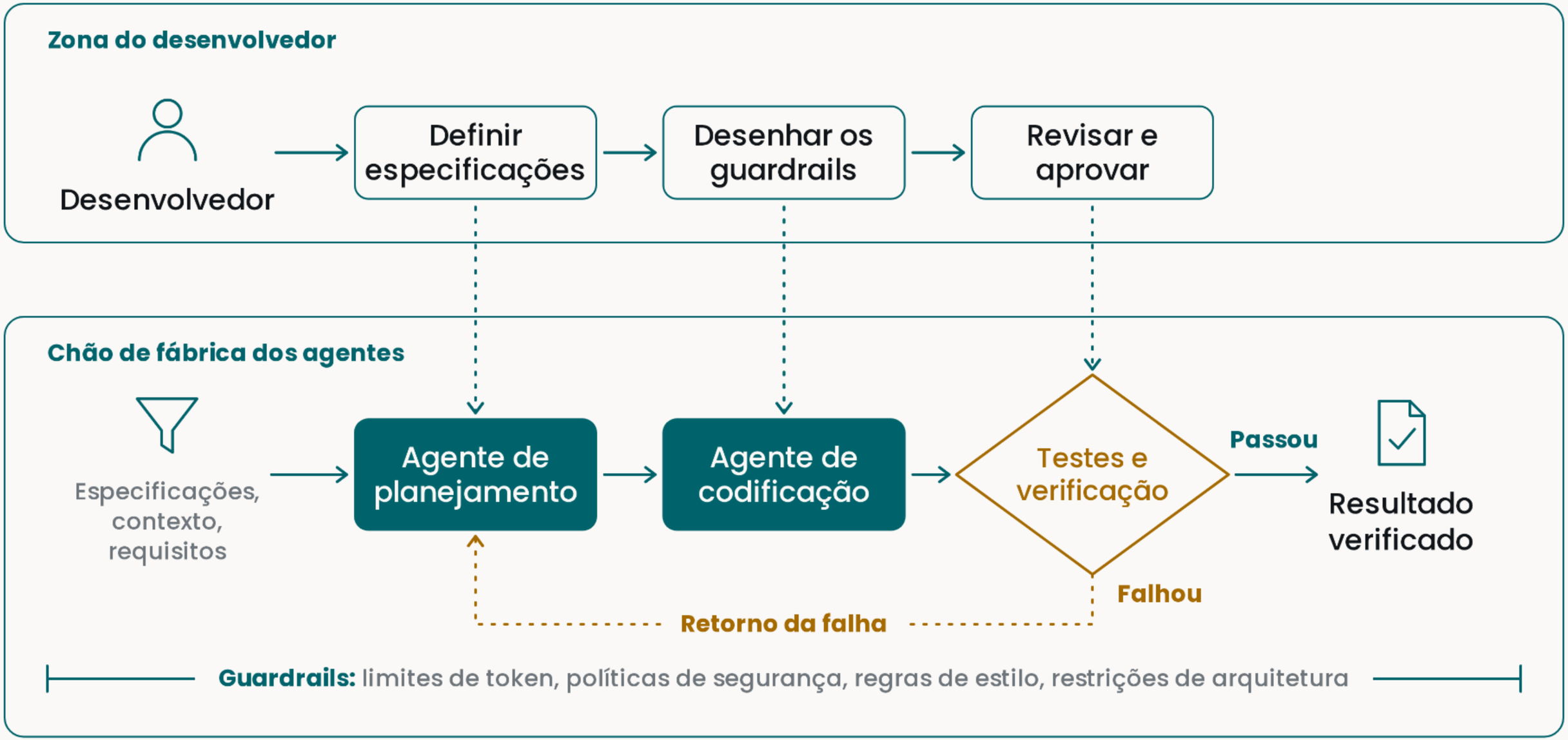
> A posição correta depende do **que está em jogo**.

SDLC iterativo tradicional

Mesmas fases, gargalos diferentes, proporções diferentes

SDLC dirigido por IA

adaptado do paper "The New SDLC With Vibe Coding", Kaggle, 2026



adaptado do paper "The New SDLC With Vibe Coding", Kaggle, 2026

Você foi promovido a chef de cozinha

Delegar implementação ≠ delegar responsabilidade.
Você é **responsável pelo resultado final**.

01 Gerenciar desenvolvimento paralelo
02 Definir padrões
03 Criar procedimentos de onboarding
04 Coordenar grandes projetos

Não tem como rejeitar a promoção

Se o seu trabalho envolve qualquer uma dessas coisas, a IA está mudando **todo trabalho relacionado a conhecimento**.

01 Pensar
02 Analisar
03 Criar
04 Comunicar
05 [...]

JÁ OUVIMOS ISSO ANTES

//

A IA não vai substituir as pessoas, mas talvez pessoas que usam IA vão substituir pessoas que não usam.

01 "Linguagens de alto nível vão matar programadores assembly"

02 "Visual Basic vai substituir programadores profissionais"

03 "Plataformas de low code vão tornar programadores obsoletos"

04 "Ferramentas no-code são o fim da engenharia de software"

Frase traduzida do Dr. Andrew Ng

Pessoas > Tecnologia

Pessoas > IA

"

Baratear ou tornar mais eficiente o uso de um recurso pode levar a uma **maior demanda** por esse mesmo recurso.

William Stanley Jevons

Paradoxo de Jevons — reddit.com/r/climatechange/comments/1kda74q

Vão existir mais empregos para desenvolvedores, não menos.

ADORA: os 5 superpoderes

A

Autonomia
fazer sozinho o que pedia um time

D

Diversão
o ciclo curto que vicia

O

Opcionalidade
adiar a decisão sem pagar por isso

R

Rapidez
e a armadilha que vem junto

A

Ambição
o que passou a caber no possível

SUPERPODER 01

Autonomia

AUTONOMIA

Desenvolvedores completam tarefas que precisariam de outras pessoas ou times.

Mais pessoas ⇒ menos tempo resolvendo o problema, mais tempo **coordenando**.

01 Fricção organizacional

02 Taxa de "ler a mente"

03 Custo de coordenação

04 Negociação e handoff

AUTONOMIA

Exemplos

01 Explicar como um repositório funciona
02 Criar protótipos executáveis
03 Não programadores tirando ideias do papel
04 Debug de erros complexos
05 Análise E2E
06 Backend implementar frontend e vice-versa

SUPERPODER 02

Diversão

DIVERSÃO

//

Vibe coding transforma o seu teclado em uma slot machine. Você puxa a alavanca e recebe uma recompensa.

01 Cada recompensa injeta dopamina e encoraja puxar de novo

02 "Vou fazer só mais uma coisinha"

03 Encurta o ciclo de feedback

Adaptado do livro Vibe Coding

SUPERPODER 03

Opcionalidade

> O desenvolvimento concorrente torna possível adiar o compromisso até o **último momento oportuno**: o momento em que a falta de decisão elimina uma alternativa importante.

Mary e Tom Poppendieck — Lean Software Development

"Decisões tomadas prematuramente são muito arriscadas" — blog.codinghorror.com/the-last-responsible-moment/

OPCIONALIDADE

Vibe Coding reduz o custo de explorar múltiplas soluções ao mesmo tempo.

01 Opção ⇒ o direito, não a obrigação, de decidir depois

02 Incerteza ⇒ menos decisões de longo prazo ⇒ mais opções

03 Decisões passam a ser mais reversíveis

04 O luxo de testar cinco ou dez soluções ao mesmo tempo

05 Modularidade aumenta as opções

OPCIONALIDADE

Exemplos

01 Propor e validar múltiplas soluções arquiteturais

02 Documentação em várias perspectivas: negócio, spec técnica, plano, código

03 Autenticação in-house vs OAuth

04 Criar POCs antes de se comprometer com a solução

SUPERPODER 04

Rapidez

RAPIDEZ

Você descreve a intenção em linguagem natural e a IA gera o código.

01 Sensação de produtividade

02 Confiança na IA

03 **Velocidade ≠ Eficiência**

04 **Eficiência ≠ Eficácia**

DORA REPORT 2025

90%

de adoção de IA entre desenvolvedores

80%

indicam ganhos de produtividade

59%

reportam influência positiva na qualidade do código

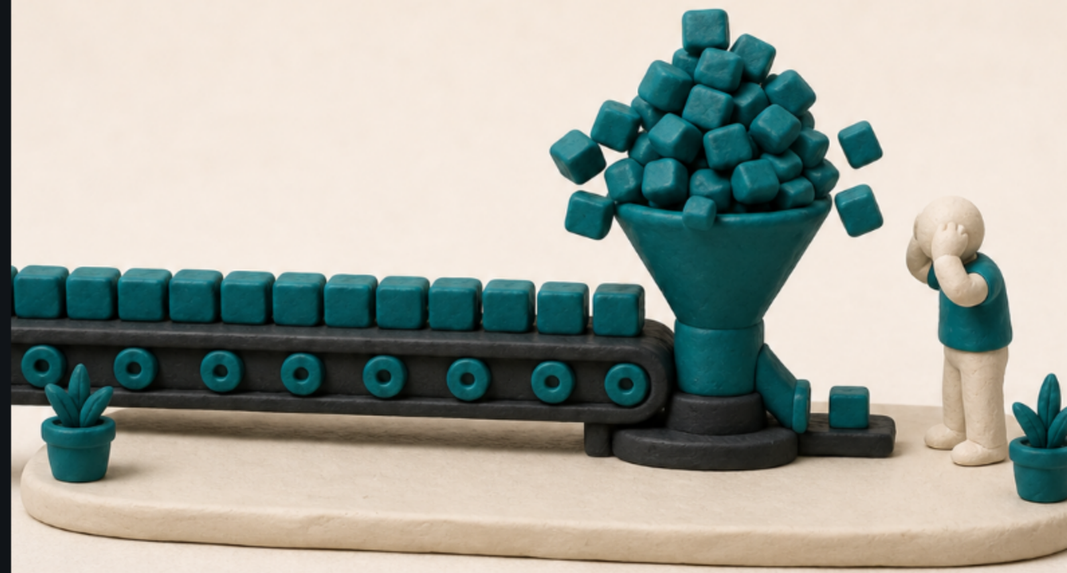
Mas em 2024 cada 25% de aumento na adoção de GenAI vinha junto com **-7% em estabilidade** e -1,5% em velocidade de entrega.

blog.google/innovation-and-ai/technology/developers-tools/dora-report-2025/

TEORIA DAS RESTRIÇÕES

//

Cada sistema tem pelo menos um gargalo



FAROS 2025 → 2026

O gargalo é a revisão

+91%
no tempo de revisão de PRs em 2025

→

+441%
no mesmo tempo, em 2026

e o resto do quadro em 2026

31%
dos PRs são mesclados sem revisão

+51%
no tamanho dos PRs

+54%
em bugs por desenvolvedor

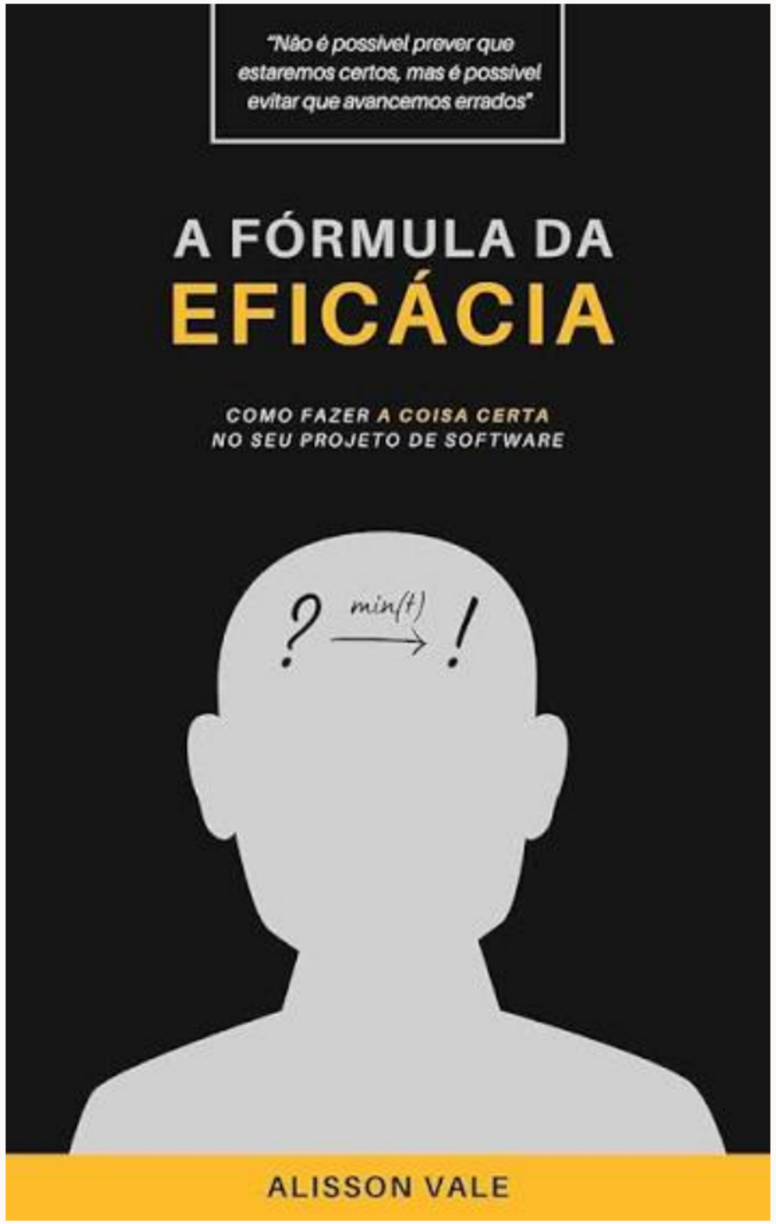
+242%
em incidentes por PR

faros.ai/blog/key-takeaways-from-the-dora-report-2025

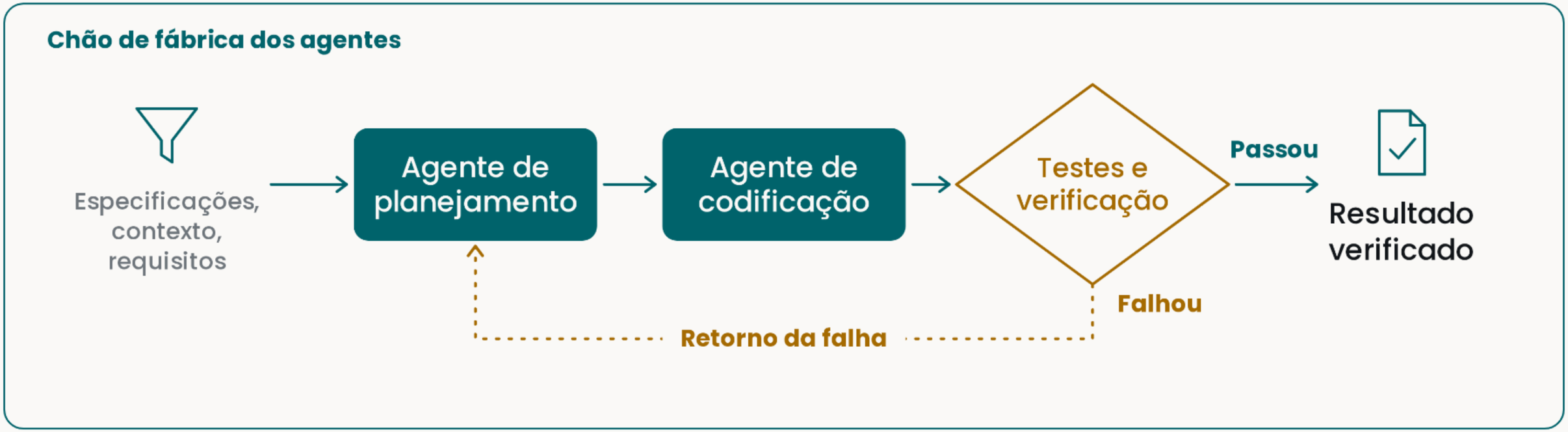
> Ser eficiente é fazer do jeito certo. Ser **eficaz** é fazer a coisa certa.

Peter F. Drucker

The Effective Executive, 1967


"Não é possível prever que estaremos certos, mas é possível evitar que avancemos errados"
A FÓRMULA DA
EFICÁCIA
COMO FAZER A COISA CERTA
NO SEU PROJETO DE SOFTWARE
? min(t) !
ALISSON VALE

? min(t) !



fórmula de Alisson Vale sobre a esteira do paper "The New SDLC With Vibe Coding", Kaggle, 2026

SUPERPODER 05

Ambição

AMBIÇÃO

//

Vibe coding altera o espectro do que é possível ser construído.

01 Custo ⬇ e retorno ⬆ ⇒ ROI ⬆

02 Tarefas são feitas, não acumulam em backlogs

03 Documentação, testes, pequenas melhorias, refatorações

Adaptado do livro Vibe Coding

AMBIÇÃO

Exemplos

01 Estruturar a documentação técnica do projeto

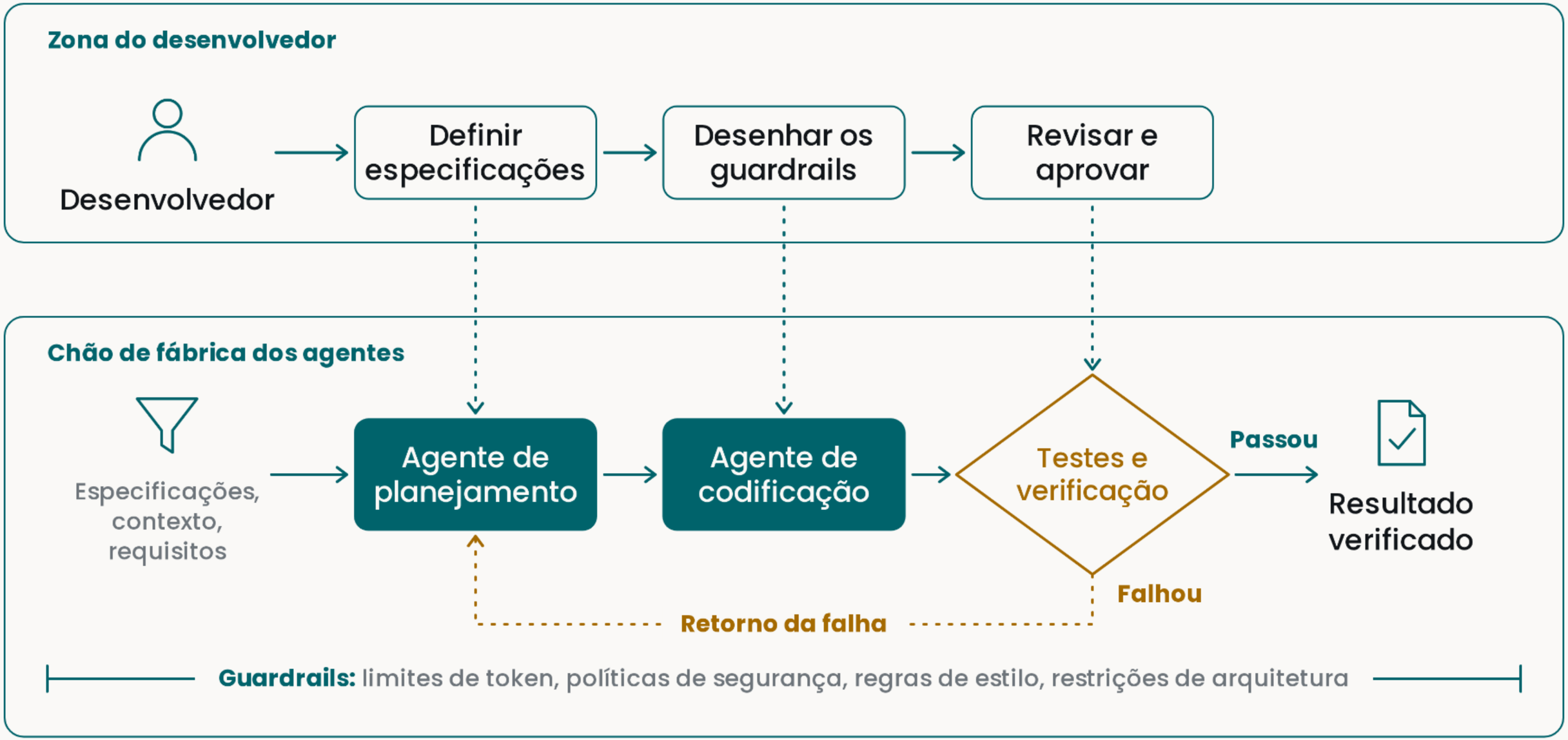
02 Definir um novo padrão e refatorar o código para segui-lo

03 Corrigir um bug numa parte do código que você não entende

04 Programadores que abandonaram o código voltando a programar

05 Programador editando vídeo com IA

Como eu faço?



adaptado do paper "The New SDLC With Vibe Coding", Kaggle, 2026

DICAS

Dicas

01 Converse como com um colega sábio

02 Copie e cole o comportamento percebido: log, mensagem, captura de tela

03 Use ferramentas difíceis sem precisar aprender cada detalhe

04 Seja claro sobre o que quer alcançar

05 Crie checkpoints para cada ponto de sucesso

DICAS

A mágica acontece quando a IA consegue observar, executar e validar o próprio trabalho

01 Defina suas regras inegociáveis

02 Tenha uma estrutura de código bem definida

03 Peça para a IA criticar o próprio trabalho

O que pode dar errado?

O QUE PODE DAR ERRADO

O que pode dar errado?

01 Tarefas muito grandes

02 Falta de critérios de aceite claros e verificáveis

03 Supervisão insuficiente em tarefas complexas e de alto impacto

04 Limites mal definidos do que a IA pode ou não fazer

05 Loop de verificação muito longo

06 Falta de definição de padrões

COMO EVITAR

Como evitar

01 Validação e controle frequentes

02 Modularidade reduz complexidade, habilita agentes paralelos e opções

03 Aprendizado constante

04 Questionar a IA pode trazer muitos aprendizados

A IA não vai substituir as pessoas, mas talvez pessoas que usam IA vão substituir pessoas que não usam.

vbmendes.dev

newsletter	blog.vbmendes.dev
site	vbmendes.dev
instagram	/vbmendes.dev
github	/vbmendes
linkedin	/vbmendes
bsky	/vbmendes
x	/vbmendes

print