

The background is a vibrant yellow. It is decorated with a complex network of black and white dots of various sizes. These dots are interconnected by thin black lines, some of which are solid and others dashed. The lines and dots form a series of overlapping circles and arcs, creating a sense of movement and connectivity, reminiscent of a network diagram or a stylized molecular structure.

**LAURENT DÉSÉCHALLIERS**

# **CONTEXTO**

Como construir um contexto que permita  
à IA entender o que importa,  
conectar seus conhecimentos e capitalizar  
o que você aprende a serviço dos seus objetivos.

# Le Contexte

*Como construir um contexto que permita à IA entender o que importa, conectar seus conhecimentos e capitalizar o que você aprende a serviço dos seus objetivos.*

---

Laurent Deséchalliers

---

## Nota sobre a tradução

---

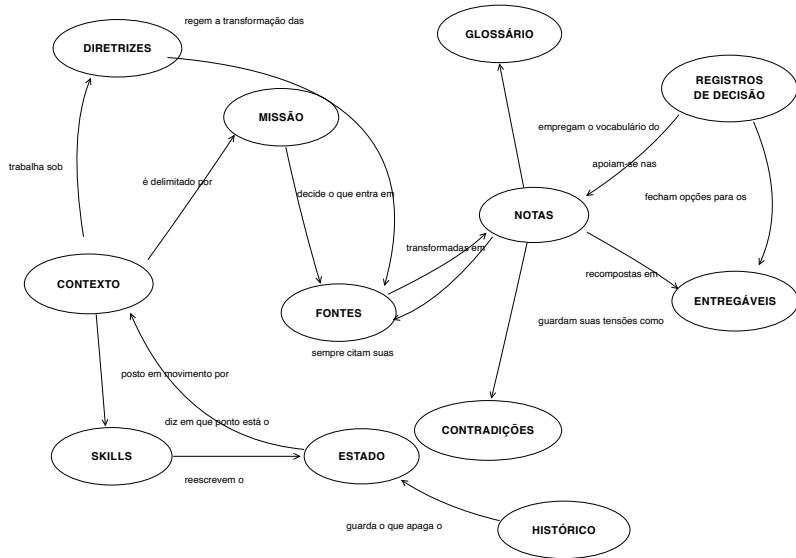
Esta edição é uma tradução automática, produzida por uma inteligência artificial a partir do texto francês.

Ela não foi revisada por um tradutor humano. As ideias, a estrutura e os exemplos são fiéis ao original; algumas construções, por outro lado, podem soar desajeitadas, e certas passagens guardarão marcas da língua de partida.

É o preço a pagar para que esta obra exista em vários idiomas, e não apenas em francês. Em caso de dúvida sobre uma formulação, a edição francesa é a referência.

## Definição do Contexto

*Um contexto é um sistema delimitado por uma missão humana, dentro do qual uma IA organiza e mobiliza o que se sabe, de onde se sabe, como os elementos se ligam entre si e em que ponto está o trabalho, para ajudar o humano a alcançar seu objetivo sem recomeçar do zero. Ao fazer isso, o sistema constitui e faz render um capital informacional duradouro: o conhecimento produzido continua disponível, se enriquece com o tempo e pode ser reutilizado para outros objetivos.*



---

## Introdução

---

Nunca tivemos tanta informação à disposição. Mesmo assim, ainda gastamos boa parte da nossa energia para reencontrá-la, recolocá-la em ordem e reconstruir o que já tínhamos entendido alguns dias antes.

Uma ideia útil raramente chega numa forma limpa. Ela nasce de uma conversa, ganha contorno num documento, encontra uma objeção numa reunião, produz uma decisão e depois vira um artigo, uma especificação ou um projeto. Nossas ferramentas guardam cada um desses rastros. O que lhes custa é guardar o fio que os liga.

A inteligência artificial muda uma parte do problema. Ela sabe ler, relacionar informações, reformular e transformar materiais muito diferentes. Mas uma conversa com uma IA continua sendo um momento frágil: o assunto avança, a discussão se alonga, e depois é preciso relembrar o objetivo, as decisões tomadas, as fontes usadas e o que ainda falta fazer.

Foi para responder a essa dificuldade que construí o `Contexto`. Um `Contexto` é um espaço de trabalho dedicado a um assunto. Um humano e uma IA guardam nele o objetivo perseguido, as fontes, as ideias, as regras e o estado do trabalho, para produzir resultados sem precisar reconstruir tudo a cada sessão.

A IA pode então trabalhar a partir de uma história real. Ela reencontra a origem de uma ideia, lembra uma escolha, sinaliza uma contradição, aponta uma prova que falta, retoma um rascunho onde ele tinha parado. **Ela ajuda a organizar o pensamento e a colocá-lo à prova. O humano continua responsável pelo sentido e pelas decisões.**

Esse sistema nasceu no meu trabalho diário, de um único arquivo e de um pedido muito simples feito a uma IA: organize o que eu vou te contar. Ele se desenvolveu por tentativas sucessivas, e cada dificuldade tornava visível uma falta. Ele continua evoluindo: algumas das soluções apresentadas aqui se sustentaram em vários usos, outras continuam sendo respostas provisórias para problemas que ainda encontro.

Este livro não parte, portanto, de uma teoria acabada. Parte de problemas que enfrentei: quinhentos e-mails que ficaram impossíveis de acompanhar, ideias espalhadas por várias ferramentas, documentos de clientes aos quais era preciso responder sem perder o trabalho já feito, textos retomados depois de uma interrupção.

Ele se dirige a quem transforma informação em compreensão, em decisões e em entregáveis: product managers, consultores, redatores, pesquisadores, analistas e, de modo mais amplo, trabalhadores do conhecimento. Ele propõe um jeito de fazer da IA uma parceira de trabalho ao longo do tempo, mantendo o controle sobre o próprio material.

---

O livro está organizado em dois livros complementares. **Livro I — Compreender o Contexto** conta como o sistema nasceu e o que ele muda na relação com a IA. **Livro II — Construir e manter vivo o Contexto** explica como construí-lo e usá-lo ao longo do tempo.

Eles podem ser lidos em sequência ou separadamente. O capítulo **Em que ordem ler este livro** apresenta os diferentes percursos possíveis.

---

## Nota sobre os exemplos

---

As reflexões e os exemplos deste livro se baseiam em mais de trinta anos de experiência profissional na área de tecnologia da informação. Eles vêm de situações encontradas em projetos e ambientes de trabalho variados, além de muitas conversas com product managers, CTOs, engineering managers e outros profissionais de diferentes empresas. Portanto, abrangem várias organizações e diferentes períodos da minha trajetória.

Para adequar essas situações ao propósito do livro, eu as modifiquei, generalizei ou recombinei. Os casos de uso, os dados numéricos e as métricas ilustram problemas recorrentes e o que aprendi com eles. São construções: nenhuma descreve um cliente, um produto ou uma organização reais, nem reflete os dados, as práticas ou as posições do meu empregador.

Os exemplos tirados da minha vida pessoal — o condomínio, meus artigos e meus próprios textos — se apoiam em situações que realmente vivi. Alguns detalhes capazes de identificar pessoas foram, porém, alterados.

As análises, os métodos e as opiniões expressos neste livro são pessoais e representam apenas o meu ponto de vista.

---

## Sobre o autor

---

Trabalho na indústria de software há várias décadas e hoje atuo como product manager. Meu percurso me levou do desenvolvimento e da arquitetura à organização das equipes, e depois ao produto, às suas operações e ao seu marketing. Essas mudanças de ofício mudaram a minha relação com a informação. Já não bastava conhecer uma solução técnica ou reencontrar um documento. Era preciso reunir sinais espalhados, entender o que eles diziam em conjunto, decidir entre várias direções e produzir textos adequados a leitores muito diferentes.

Meu interesse por inteligência artificial e por sistemas distribuídos é antigo. Hoje ele encontra uma pergunta bem concreta: **como trabalhar com uma IA sem entregar a ela o controle sobre as próprias informações, as próprias decisões e o próprio pensamento?**

Colaborei com a imprensa especializada e trabalhei como revisor técnico para a M2 Édition e a Eyrolles. Hoje escrevo em [malorean.net](http://malorean.net), sobre product management, IA, métodos de trabalho e sistemas de informação.

Também colaboro com a *Proposition Marketing*, uma revista independente dedicada ao marketing estratégico e operacional. Escrever para leitores vindos do marketing e da direção de empresas me obriga a explicar as questões de

produto sem supor um vocabulário comum. Essa exigência também é a deste livro.

Eu teria muito prazer em trocar ideias com outras pessoas que constroem sistemas próximos, diferentes ou complementares: o assunto merece ser discutido, confrontado e enriquecido por outras práticas. E também com quem pesquisa questões vizinhas na universidade — passei alguns anos em dois centros de pesquisa do CNRS, e reencontrar esse meio seria um prazer.

Laurent Deséchalliers

- [laurent@malorean.net](mailto:laurent@malorean.net)
- <https://malorean.net>

---

## Agradecimentos

---

Meus primeiros agradecimentos vão para você.

Você que teve a ideia um pouco louca de abrir este livro: um livro um pouco mais grosso do que o previsto, que certamente contém algumas repetições e trata de um assunto abstruso.

Até aqui trabalhei no Contexto de um jeito bastante solitário.

Espero, no entanto, que essa solidão não dure: que o assunto desperte em outras pessoas a vontade de explorá-lo, de discuti-lo, de criticá-lo e de fazê-lo evoluir. Talvez os próximos artigos publicados no [malorean.net](http://malorean.net) sejam então a ocasião de agradecer a muitos colaboradores e colaboradoras.

Agradeço sobretudo à minha mulher, Christine.

Ela viu as noites e os fins de semana se prolongarem diante deste livro. Ela me apoiou durante toda a escrita dele, sabendo ao mesmo tempo me lembrar de que era tarde e de que já era mesmo hora de fechar esse maldito computador.

Prometido, foi só um parêntese.

E, muito além deste livro, obrigado por me acompanhar todos esses anos nesta bela existência que nós dois compartilhamos.

---

## Em que ordem ler este livro

---

O livro segue a ordem em que as coisas me aconteceram: um problema vivido, uma solução improvisada e, por fim, um sistema. Por isso a descrição completa do Contexto só chega no Livro II.

**O percurso “compreender”** é essa ordem. Você lê de ponta a ponta, e cada peça do sistema aparece no momento em que a falta que a produziu acabou de ser mostrada.

**O percurso “construir”** é para quem já viveu o problema e quer a mecânica, não a narrativa.

1. **Definição do Contexto**, algumas linhas para saber do que se trata.
2. **A parte 5 — Por dentro de um Contexto**, na ordem dos seis capítulos dela: o mapa, abrir, alimentar, transformar, produzir e decidir, salvar.
3. **A parte 6 — O Contexto no seu ecossistema**, para ligar o que você acabou de construir ao resto dos seus sistemas de informação.
4. **O Livro I**, quando uma escolha de projeto lhe parecer arbitrária: é lá que você encontra o problema que a provocou.

Esse atalho custa alguma coisa. O Livro II foi escrito para um leitor que já tem em mente as situações do Livro I, e faz referência a elas. Você vai cruzar, portanto, com exemplos em

---

que lhe faltará um pouco de contexto — é o preço a pagar, e ele continua valendo se você já sabe por que está aqui.

---

## Experimentar antes de ler

---

Este livro descreve um método, e um método se conta mal. Você pode ver um funcionando antes de ler a teoria, ou durante — muitas vezes é o melhor momento para entender para que serve cada peça.

Três repositórios estão abertos em [github.com/laurent-malorean](https://github.com/laurent-malorean).

**Contexte Nano Notion** é o mais rápido de instalar. Não exige ambiente de desenvolvimento nem competência técnica: bastam um ChatGPT e um Notion. O ChatGPT carrega a inteligência do Contexto, o Notion lhe serve de memória — é ali que as fontes, as notas, as decisões e o estado são escritos e relidos de uma conversa para outra. Em poucos minutos você tem um Contexto que se sustenta.

**Contexte V2** é a implementação completa, a que uso todos os dias e com a qual este livro foi escrito. Instala-se com o Claude Code ou com o Codex, um ou outro à escolha, e pede um pouco mais de preparação que a anterior.

**LLM Wiki** não é um Contexto. É uma biblioteca de conhecimento construída a partir de documentos — livros, normas, artigos, cursos — dos quais a IA extrai fichas curtas e ligadas entre si, que depois se consultam em linguagem corrente. O Contexto mantém o fio de uma missão; o LLM Wiki guarda em massa o que já foi lido. A parte 6 explica como os dois trabalham juntos.

---

Nada obriga a começar por aí. Mas se em algum momento a leitura lhe parecer abstrata, esse é o caminho mais curto para que volte a ser concreta.

## Do mesmo autor

---



**Malorean Corp** — Até pouco tempo atrás, transformar uma ideia em protótipo às vezes levava semanas. Hoje, uma pessoa com a ajuda da inteligência artificial pode conseguir isso em poucos dias. Essa mudança é impressionante. Mas a verdadeira transformação talvez aconteça em outro lugar: na maneira como trabalhamos juntos.

*Lançamento previsto: final de novembro de 2026.*

Mais informações em <https://malorean.net>.

# Sumário

## LIVRO I

### Compreender o Contexto

|                                                                                 |           |
|---------------------------------------------------------------------------------|-----------|
| <b>Parte 1 — O nascimento de uma necessidade .....</b>                          | <b>21</b> |
| 1.1 Quinhentos e-mails e uma história que desaparece .....                      | 22        |
| 1.2 Uma primeira definição .....                                                | 31        |
| <b>Parte 2 — O que o Contexto muda .....</b>                                    | <b>39</b> |
| 2.1 Da conversa a um trabalho que se pode retomar .....                         | 40        |
| 2.2 Dois clientes, um white paper e uma contradição a conservar .               | 47        |
| <b>Parte 3 — Três usos concretos .....</b>                                      | <b>57</b> |
| 3.1 Responder a um questionário do cliente sem perder o trabalho<br>feito ..... | 58        |
| 3.2 Escrever um artigo a partir de um pensamento ainda informe .                | 64        |
| 3.3 Amadurecer um tema de produto no meio do caos .....                         | 72        |
| <b>Parte 4 — O que os entregáveis deixam para trás .....</b>                    | <b>83</b> |
| 4.1 A resposta cabe numa célula, o raciocínio não .....                         | 84        |
| 4.2 Seis meses depois, a resposta já não basta .....                            | 88        |

LIVRO II

Construir e manter vivo o Contexto

Parte 5 — Por dentro de um Contexto ..... 95

5.1 O mapa completo ..... 96

5.2 Abrir: reencontrar a missão e o presente ..... 107

5.3 Alimentar: fazer entrar o material ..... 118

5.4 Transformar: passar das fontes às ideias ..... 125

5.5 Produzir e decidir ..... 135

5.6 Salvar e retomar ..... 145

Parte 6 — O Contexto no seu ecossistema ..... 153

6.1 Consultar conhecimentos estruturados ..... 154

6.2 Consultar o produto real ..... 161

6.3 O dia em que tive três Contextos ..... 170

6.4 Ler na casa do vizinho sem se mudar para lá ..... 176

6.5 O que a circulação custa ..... 183

Antes de fechar

Encerramento ..... 191

7.1 Epílogo ..... 192

7.2 Parentescos e vizinhanças ..... 195

## LIVRO I

# Compreender o Contexto

---

- 1 O nascimento de uma necessidade
- 2 O que o Contexto muda
- 3 Três usos concretos
- 4 O que os entregáveis deixam para trás



# Parte 1

## O nascimento de uma necessidade

---

O Contexto não nasceu de uma ideia sobre inteligência artificial. Nasceu de uma caixa de e-mails que tinha se tornado impossível de administrar e de algo que demorei a perceber: o verdadeiro problema não era o número de mensagens.

Esta parte conta essa origem e tira dela uma primeira definição: simples o bastante para ser lembrada e completa o bastante para orientar o restante do livro.

## Quinhentos e-mails e uma história que desaparece

---

Na primeira vez que criei um `Contexto`, eu não estava tentando inventar um método de trabalho com a inteligência artificial. Só estava tentando parar de perder o fio do meu condomínio. Faço parte do conselho do condomínio. Por isso recebo as mensagens do síndico, dos proprietários, dos inquilinos e dos prestadores:

- um e-mail anuncia um problema novo,
- o seguinte fecha um orçamento,
- um terceiro adia um serviço,
- uma semana depois alguém responde no mesmo tópico para falar de outro bloco.

Nesse meio-tempo, outros três assuntos avançaram.

O volume não era o único problema. Uns quinhentos e-mails se espalhavam por uns quinze assuntos:

- obras,
- finanças,
- seguros,
- assembleias,
- inadimplentes,
- concorrências,
- ocorrências e pedidos diversos

- ...

Cada um andava no seu ritmo, às vezes por meses.

Quem acompanha uma obra todo dia guarda a história dela na cabeça sem esforço. Sabe qual prestador esteve lá, por que o cronograma mudou e qual decisão ainda espera resposta. Quando eu abria uma mensagem nova, nem sempre tinha essa continuidade. Precisava voltar pelo histórico da conversa, encontrar as mensagens anteriores e reconstruir a situação antes mesmo de conseguir entender a nova informação.

Eu queria poder fazer perguntas simples:

- como está o telhado?
- qual prestador mandou o último orçamento das calhas?
- o que decidimos sobre aquela infiltração?
- o que ainda esperamos do síndico?

Minha caixa de entrada tinha as respostas. O que ela não me dava era a visão de conjunto.

### **Um arquivo e uma frase**

Comecei com o arranjo mais simples possível: um arquivo e uma instrução dada a uma IA.

| *Junte as informações e divida por assuntos.*

Eu passava os e-mails. Ela os associava a assuntos e mantinha um resumo de cada um.

- Tudo sobre as obras ficava junto.
- As questões de dinheiro formavam outro bloco.

- Os prestadores, os seguros e as assembleias foram achando seu lugar.

1

Essa primeira tentativa já trazia algo. Eu podia perguntar como estava um assunto sem reler todas as mensagens que tratavam dele. A IA não procurava mais apenas uma ocorrência na caixa: ela me mostrava o que tinha entendido da situação.

O arquivo se reorganizava conforme chegavam novas informações.

- Aparecia uma seção quando um assunto novo ganhava peso.
- Duas seções se separavam quando começavam a misturar realidades diferentes.

A informação já não se limitava a se acumular. Ela tomava uma forma com a qual dava para trabalhar.

Depois esse primeiro sistema deu uma resposta exata e ainda assim insuficiente.

### **A obra estava pronta, mas a história dela tinha sumido**

Quando uma obra chegava ao fim, o resumo dizia: “obra concluída, recebimento sem ressalvas”.

Esse era mesmo o estado presente dela. Também era uma perda.

- Quando a obra tinha começado?
- Em que momento tinha atrasado?
- Por que o cronograma tinha mudado?
- Quando o aditivo tinha sido assinado?

Todas essas etapas tinham contado. A última atualização as apagara em favor da situação final.

Eu tinha pedido um resumo. A IA o mantinha atualizado substituindo o estado antigo pelo novo. Ela tinha seguido a instrução, mas a instrução não protegia a história.

Eu precisava, então, de duas vistas do mesmo assunto.

- A primeira tinha de responder rápido à pergunta “como estamos?”.
- A segunda tinha de guardar os fatos na ordem: início da obra, atraso, novo acordo, serviço, recebimento.
- O estado presente permitia agir.
- A história permitia entender como tínhamos chegado a esse estado e conferir um ponto em caso de dúvida.

Essa distinção nasceu de uma falha bem concreta. Eu a tomava por uma melhoria de organização. Ela virou um dos princípios mais importantes do meu trabalho com a IA: **o que é verdade agora e o que aconteceu não cumprem a mesma função.**

### **A IA não podia adivinhar minhas regras**

Outro problema apareceu com os nomes:

- dois prestadores tinham o mesmo sobrenome.
  - Para distingui-los era preciso entender o assunto do e-mail.
- uma proprietária aparecia ora com o sobrenome de solteira, ora com o de casada.

Sem nenhuma indicação, a IA podia criar duas pessoas onde havia só uma.

Outras diferenças vinham do jeito de trabalhar.

- Os inadimplentes abaixo de cem euros não eram acompanhados como os outros.
- Esse limite não vinha de nenhuma regra universal que a IA pudesse conhecer.
- Vinha da nossa maneira de fazer.

Comecei a formular esses conhecimentos como regras duradouras, escritas por extenso no arquivo:

```
# Regras de tratamento

- Martin citado a respeito do telhado designa o telhadista;
  Martin citado a respeito
  das áreas verdes designa o jardineiro.
- Dupont-Lemoine e Lemoine designam a mesma proprietária,
  unidade 14.
- Apresentar os inadimplentes abaixo de cem euros numa
  lista separada.
```

Nada mais sofisticado do que isso. Três frases que eu teria de repetir a cada conversa e que não precisei dizer nem mais uma vez.

Cada regra vinha de um erro observado ou de uma ambiguidade real. Ela fixava um conhecimento que eu carregava na cabeça e que a IA não podia reconstruir com certeza.

Assim o sistema começava a guardar dois tipos de informação:

- as informações sobre o condomínio,
- a maneira como eu queria que fossem tratadas.

### **O arquivo que ajudava começou a atrapalhar**

Com as semanas, o arquivo cresceu. Passou de quinhentas linhas, depois de oitocentas, depois de mil.

Esse crescimento mostrava sua utilidade:

- os assuntos eram mais bem acompanhados,
- os prestadores identificados,
- as decisões reencontradas.

Mas o que tinha tornado o sistema possível começava a limitar seu uso.

- Informação demais, de tipos diferentes, dividia o mesmo espaço.
- As regras se misturavam aos fatos.
- As finanças conviviam com as obras, as pessoas, os seguros e as concorrências.

Encontrar a informação certa voltava a dar trabalho.

Primeiro separei as regras de tratamento. Elas não descreviam o condomínio; diziam como trabalhar com as informações sobre ele. Isolá-las permitia reencontrá-las e aplicá-las com mais facilidade.

Depois repartí as informações entre vários temas. As obras ganharam seu próprio espaço, assim como as finanças, os prestadores, as decisões de assembleia ou os seguros. Uma visão geral dizia o que cada um continha. A IA podia então

abrir o tema pertinente à pergunta, em vez de percorrer todo o conteúdo a cada vez.

Por fim, reuni o vocabulário específico: termos jurídicos, siglas e expressões próprias do condomínio. O objetivo não era construir uma enciclopédia. Era dar um sentido estável às palavras das quais dependia entender as trocas.

Eu continuava sem nenhuma grande arquitetura na cabeça. Acrescentava uma peça quando um problema tornava essa peça necessária.

### **Confiar sem perder a fonte**

A IA às vezes errava. Podia aproximar duas informações que não tratavam exatamente do mesmo assunto ou escolher o prestador errado entre dois homônimos. O sistema tinha, então, de acelerar a compreensão sem virar a única versão disponível dos fatos.

Guardei todos os e-mails originais. Quando eu precisava conferir uma resposta ou quando havia algo importante em jogo, podia voltar às informações na forma em que tinham chegado. O resumo dava acesso rápido. A fonte permitia conferir.

Essa escolha é o que tornava o uso viável. Eu podia aceitar algum erro pontual na organização dos e-mails, porque as mensagens originais continuavam acessíveis. O mesmo grau de confiança não serviria para um documento bancário, uma obrigação regulatória ou uma decisão irreversível. O risco

depende sempre do que se confia ao sistema e da possibilidade de conferir.

O Contexto não substituía, portanto, minha caixa de entrada. Ele transformava o conteúdo dela numa memória de trabalho, deixando-me uma porta de volta para as mensagens originais.

### **O que tinha aparecido**

Em poucas semanas, o ponto de partida — um arquivo e uma frase — tinha virado um conjunto capaz de atender a várias necessidades: mostrar o estado presente, guardar a história, aplicar regras próprias do condomínio, distinguir os assuntos, firmar o vocabulário e voltar às fontes.

Quando chegava um e-mail novo, eu podia me ocupar do que ele mudava em vez de reconstruir tudo o que viera antes.

Quando surgia uma pergunta, eu podia fazê-la nos termos do problema em vez de caçar a mensagem certa no tópico certo.

Quando a IA errava, eu podia corrigir uma regra ou voltar ao texto original.

Eu não tinha concebido o Contexto de uma vez. Tinha-o visto aparecer através de uma sucessão de insuficiências: o resumo apagava a história, os nomes se confundiam, as regras se perdiam, o arquivo ficava grande demais. Cada correção tornava o trabalho mais duradouro e mais fácil de retomar.

Essa origem conta. O Contexto não nasceu da vontade de classificar mais. Nasceu da necessidade de entender informações que mudam, guardar as evidências necessárias

para conferi-las e dar continuidade ao trabalho sem recomeçar do zero.

---

1

**PARA LEMBRAR — DUAS VISTAS DO MESMO ASSUNTO**

O estado responde a “como estamos?”. A história responde a “como chegamos aqui?”. Um resumo mantido em dia esmagando o anterior guarda a primeira e perde a segunda. As duas são necessárias: uma para agir, a outra para entender e conferir.

---

## Uma primeira definição

---

O sistema nascido daqueles quinhentos e-mails já pode receber uma primeira definição.

*Um Contexto é um espaço de trabalho dedicado a um assunto, onde uma pessoa e uma IA guardam o que precisam para entender, tomar decisões e produzir resultados ao longo do tempo.*

Esta definição é propositalmente simples. O Contexto descrito neste livro tem uma organização precisa, que a segunda metade vai detalhar. Para entender o que ele traz, no começo bastam seis referências:

- um objetivo,
- informações,
- ideias,
- um estado do trabalho,
- regras próprias,
- resultados a produzir.

### **Um assunto e um objetivo**

O Contexto do condomínio não tentava conter tudo o que eu sabia. Ele perseguia um objetivo: entender os assuntos em curso, guardar a história deles e reencontrar as informações necessárias para agir no conselho do condomínio.

Esse objetivo formava uma fronteira. Um e-mail sobre o telhado tinha seu lugar. Uma leitura interessante sem relação com o condomínio, não. Esse limite evitava que o sistema virasse um depósito onde se acumulasse toda informação julgada interessante.

Um Contexto começa, então, por uma intenção. Ele pode servir para:

- escrever um livro,
- explorar um problema de produto,
- preparar uma decisão,
- responder a questionários do cliente,
- acompanhar uma entidade de negócio,
- ...

### **Informações cuja origem é preservada**

As informações entram num Contexto de duas maneiras.

A primeira recebe o que já existe antes da sessão de trabalho:

- um e-mail,
- um artigo,
- um livro,
- uma página web,
- a transcrição de uma entrevista
- ...

São as **fontes documentais**. Chegam como objetos já formados e permanecem estáveis. No caso do condomínio, o e-mail que anuncia o adiamento de um serviço pertence a essa primeira categoria.

A segunda recebe o que se formula durante o trabalho com a IA. Quando explico uma situação, corrijo uma interpretação, afinio uma nuance ou exprimo uma convicção vinda da minha experiência, o que eu digo vira uma **fonte de sessão**. Ela se constrói aos poucos ao longo da troca. Quando explico que dois nomes designam a mesma proprietária, o que eu digo entra primeiro por essa segunda porta.

A diferença está, portanto, no momento e na maneira como as informações aparecem.

- O documento existe antes da sessão e é trazido para ela.
- A fonte de sessão nasce durante a sessão e cresce com ela.

Depois, as duas cumprem a mesma função: guardar o que foi recebido ou dito antes de interpretá-lo.

- Elas não garantem que tudo o que contém seja verdade.
- Elas permitem reencontrar o que foi recebido ou dito e, depois, avaliar sua confiabilidade.

A partir dessas fontes, o Contexto pode extrair uma ideia, uma regra ou um estado do trabalho. O conhecimento obtido ao interpretar e organizar as informações fica registrado em outro lugar. Se uma data parece falsa, se duas pessoas foram confundidas ou se uma interpretação fica discutível, dá para voltar à origem dela.

### **Ideias tornadas utilizáveis**

Acumular fontes não basta. Quinhentos e-mails bem guardados continuam sendo quinhentos e-mails para ler.

Por isso o Contexto vai transformando essas fontes em conhecimento útil. Ele faz aparecer:

- assuntos,
- fatos importantes,
- regras,
- distinções,
- perguntas abertas,
- relações.

Uma troca isolada entra na história da obra a que pertence. Várias menções de um mesmo prestador enriquecem um conhecimento comum. Uma correção impede que uma confusão se repita.

Essa etapa intermediária conta porque torna a informação utilizável.

- As fontes preservam o que foi recebido ou dito.
- O conhecimento organizado permite entender e agir.

Em trabalhos mais complexos, certas ideias poderão servir várias vezes: um mesmo conhecimento de produto alimenta uma decisão, uma especificação e uma documentação; um mesmo fato verificado ajuda a responder a vários questionários do cliente. O resultado muda, enquanto a ideia e a procedência dela continuam disponíveis.

### **Um estado que permite continuar**

Um trabalho em andamento sempre tem um estado atual: o que se sabe, o que continua incerto, o que foi decidido e o que deve vir depois.

No condomínio, o estado respondia a uma pergunta imediata: como está essa obra hoje? A história dela respondia a outra: como chegamos até aqui? O sistema precisava das duas para que eu realmente pudesse retomar o trabalho.

Essa distinção vai além do acompanhamento de uma obra. Quando interrompo a escrita de um texto ou a exploração de um problema de produto, quero reencontrar mais do que um monte de documentos. Quero saber:

- o que está firmado,
- que ponto continua frágil,
- que ação pode colocar o trabalho de novo em movimento.

O Contexto guarda essa continuidade fora da conversa.

- A sessão com a IA pode parar.
- O trabalho conserva um estado ao qual a sessão seguinte poderá voltar.

### **Regras próprias do trabalho**

A IA dispõe de conhecimentos gerais. Em compensação, ignora boa parte do que torna uma situação particular:

- os dois prestadores que têm o mesmo sobrenome,
- a maneira como uma equipe classifica um sinal vindo do cliente,
- o nível de prova exigido antes de publicar uma afirmação
- o tom escolhido para um livro,
- ...

O Contexto guarda essas regras junto do trabalho que elas regem. Elas podem ser corrigidas quando a experiência

mostra seu limite. Depois se aplicam sem que o humano precise reformulá-las a cada pedido.

Essas regras não dão à IA a responsabilidade do trabalho. Elas tornam explícito o que o humano espera dela. A pessoa define o objetivo, traz ou valida as fontes, corrige as interpretações e toma as decisões. A IA incorpora as informações, estabelece relações, organiza, faz perguntas, reformula e prepara resultados dentro desses limites.

### **Resultados adaptados a uma necessidade**

O `Contexto` continua focado no que deve tornar possível.

No condomínio, o resultado podia ser:

- uma resposta a uma pergunta,
- um panorama das obras,
- a preparação de uma conversa.

Em outros usos será:

- um artigo,
- um questionário do cliente,
- uma documentação.

O mesmo espaço pode produzir vários resultados porque preserva as fontes antes que elas recebam uma forma final. As fontes não ficam trancadas no primeiro documento que as usa. As ideias continuam disponíveis para outra pergunta ou outro público.

Essa capacidade distingue o `Contexto` de um simples arquivo morto.

- Um arquivo morto permite conservar.

- O Contexto organiza como o que foi conservado é retomado e transformado.

### **Uma forma que pode começar de modo bem simples**

Na minha prática, um Contexto toma a forma de arquivos que eu posso ler, mover e guardar independentemente da ferramenta que me ajuda a trabalhá-los. Essa forma atende à minha necessidade de controle e continuidade. Outras realizações são possíveis, se preservarem as mesmas funções. O primeiro Contexto, no entanto, não tinha nada de sofisticado. Cabia num arquivo. Cresceu quando seus limites ficaram visíveis. Essa progressão continua importante: não é preciso construir todo o sistema descrito neste livro para começar a testar a ideia.

Bastam um assunto, um objetivo e algumas fontes iniciais.

- A IA ajuda a organizá-los.
- O humano observa o que falta, acrescenta uma regra, guarda uma fonte ou separa duas informações que não têm o mesmo papel.
- A arquitetura aparece no ritmo dos problemas encontrados.

O Contexto, portanto, não designa tanto uma estrutura específica de pastas, mas uma relação de trabalho duradoura entre uma pessoa, uma IA e um conjunto de informações organizado em torno de um objetivo.

A parte seguinte vai examinar o que essa relação muda. Uma conversa com uma IA já pode ser fecunda. A passagem ao Contexto começa quando essa conversa acontece num

ambiente capaz de preservar seu objetivo, sua história e o estado do trabalho para além do momento presente.

---

1

**PARA LEMBRAR — AS SEIS REFERÊNCIAS DE UM CONTEXTO**

Um objetivo que traça uma fronteira, informações cuja origem é preservada, ideias tornadas utilizáveis, um estado que permite continuar, regras próprias e resultados a produzir. Essas seis referências bastam para começar. A arquitetura vem depois, quando um problema encontrado a torna necessária.

---

# Parte 2

## O que o Contexto muda

---

Uma conversa com uma IA já pode oferecer muita coisa. Ela explica, reformula, propõe e muitas vezes dá uma resposta melhor do que a que você teria achado sozinho.

Esta parte pergunta, então, o que um Contexto acrescenta. A resposta não está na qualidade das respostas obtidas, mas no que permanece de uma sessão para outra: um trabalho que recomeça onde tinha parado e documentos que preservam seus desacordos em vez de aplainá-los.

## Da conversa a um trabalho que se pode retomar

---

Uma conversa com uma IA pode começar de um jeito bem simples.

Dou a ela uma ideia ainda confusa. Ela reformula. Peço um roteiro, depois outro. Ela levanta uma objeção em que eu não tinha pensado. Deixo minha intenção mais clara, descarto um caminho, aprofundo outro. Em poucos minutos a conversa faz avançar algo que talvez tivesse ficado muito tempo diante de uma página em branco.

Essa é uma das grandes forças do chat: basta uma pergunta para começar. Para uma tarefa curta, essa forma funciona muito bem. Peço uma síntese, uma explicação, algumas possibilidades. A resposta chega, eu uso, e a conversa pode terminar.

A relação muda quando o trabalho dura.

### Cinco frases antes de conseguir escrever

Reabro uma conversa de seis dias atrás para retomar um texto começado na semana anterior. Antes de conseguir escrever uma única linha, digito isto:

*Retome o texto sobre a qualidade dos relatórios de serviço. Ele é dirigido a responsáveis de operações, não a desenvolvedores. Tínhamos descartado encurtar o*

*formulário, é cedo para decidir. O número sobre as auditorias vem de uma única entrevista, trate como hipótese. Estávamos na terceira seção.*

Cinco frases para recolocar a IA dentro de um trabalho que ela já tinha diante de si seis dias antes.

Tudo o que acabei de digitar existia antes de eu digitar: o ângulo, o público, a decisão, o ponto frágil e o lugar no texto estavam na conversa anterior. Eu apenas os recuperei da minha própria memória, porque a conversa os tinha guardado na ordem cronológica em que apareceram.

O trabalho segue outra lógica.

- Uma instrução dada no começo ainda precisa valer no fim.
- Uma fonte acrescentada ontem precisa poder sustentar um parágrafo escrito amanhã.
- Uma decisão antiga precisa impedir que a mesma discussão se reabra.
- Um ponto pendente precisa reaparecer na hora certa.

Quanto mais a conversa cresce, mais o humano precisa segurar essa ordem na cabeça. Quando começa uma sessão nova, ele reconstrói a situação para a IA: este é o objetivo, isto foi o que mudou, este é o tom, estas são as decisões, paramos aqui.

O chat continua útil. O peso da continuidade, porém, recai sobre quem o usa.

## **Dar um lugar ao que deve durar**

Um Contexto muda a relação ao dar ao trabalho um espaço distinto da conversa.

A discussão continua sendo o lugar da troca. É onde se questiona, se explora, se esclarece e se corrige. O que deve durar passa para o espaço de trabalho:

- o objetivo,
- as fontes,
- as regras,
- as ideias firmadas,
- as decisões,
- os textos,
- o ponto em que o trabalho está.

A diferença pode parecer pequena. É como a que separa uma reunião do projeto de que ela trata. Uma boa reunião pode fazer o projeto avançar, mas o projeto não cabe inteiro na memória da reunião. Os documentos, as decisões e as ações continuam existindo depois que ela acaba.

Guardar a transcrição completa do chat não produziria o mesmo efeito. Ela é preciosa quando se quer reencontrar exatamente o que foi dito, mas segue a forma da conversa: uma pergunta, uma resposta, uma correção, uma divagação, uma volta ao assunto. Para retomar o trabalho, ainda seria preciso reler essa sequência e decidir o que conta agora. O Contexto, por sua vez, transforma o que foi dito.

- Uma correção vai para o lugar onde poderá impedir o erro de voltar.
- Uma decisão continua visível como decisão.
- Uma fonte continua ligada ao que ela sustenta.
- Um texto existe na sua última versão completa.

A memória passa, assim, a ser organizada para facilitar a ação. Ela não busca apenas guardar o passado. Ela prepara a intervenção seguinte.

### **Uma página para reencontrar trinta e dois textos**

Medi o que isso muda ao voltar ao trabalho neste livro.

O manuscrito tinha então trinta e dois textos. Algumas partes estavam escritas, outras ainda vazias. Uma seção tinha sido dividida porque ocupava um espaço desproporcional. Dois trechos esperavam minha releitura. Uma afirmação não se apoiava em nenhum exemplo concreto e precisava ser refeita.

Um a um, nenhum desses elementos é difícil de entender.

Juntos, porém, formam uma situação que é preciso reconstruir antes de conseguir escrever uma única linha útil.

Eu poderia abrir os trinta e dois textos, reler o roteiro, percorrer minhas conversas antigas e procurar as observações deixadas em suspenso. Esse trabalho de arqueologia levaria tempo e deixaria uma dúvida: será que esqueci justamente o ponto que devia impedir um erro?

Em vez disso, abro o Contexto do livro. Ele me entrega um arquivo de retomada, que naquela manhã dizia mais ou menos isto:

```
# Retomada
```

```
## Onde estamos
```

```
Existem trinta e dois textos, do roteiro à conclusão.
```

```
As partes técnicas estão escritas; as duas primeiras  
    continuam teóricas demais.
```

```
## O que saber antes de continuar
```

```
O leitor que temos em mente está descobrindo o assunto: ele  
    perde o fio assim que um conceito aparece antes de um  
    exemplo concreto.
```

```
A afirmação sobre o fracasso dos sistemas documentais ainda  
    não se apoia em nenhuma experiência pessoal.
```

```
## A próxima ação
```

```
Reescrever a abertura em torno do caso dos quinhentos  
    e-mails do condomínio.
```

```
## O que ficou aberto
```

```
Dois trechos esperam minha releitura.
```

```
Várias fontes entraram sem terem sido transformadas em  
    notas.
```

Uma página, lida em menos de um minuto. Sei o que fazer, e não abri nenhum dos trinta e dois textos.

### **Continuar não é retomar**

**Continuar é conhecer a ação seguinte. Retomar é reencontrar o sentido do trabalho antes de realizar essa ação.**

Uma lista de tarefas pode me dizer que uma fonte precisa ser tratada. Ela não me explica por que essa fonte vem primeiro, que parte do livro ela deve iluminar nem que risco ela deve permitir corrigir.

Para retomar, preciso de um conjunto mais rico:

- o objetivo que o trabalho persegue;
- o ponto a que ele realmente chegou;
- as decisões que ainda orientam o que vem;
- os pontos frágeis que não podem ser esquecidos;
- uma próxima ação precisa o bastante para recomeçar.

O arquivo de retomada, aliás, se lê nessa ordem. “Trinta e dois textos” dá uma medida. “Ele perde o fio assim que um conceito aparece antes de um exemplo concreto” indica uma direção. A próxima ação decorre das duas e nomeia o caso preciso que deve abrir o livro.

O que foi recuperado é um raciocínio, não apenas uma posição numa lista.

Ainda assim, a retomada não tenta recriar na minha cabeça cada detalhe da sessão anterior. Um resumo curto demais apaga a razão de uma escolha. Um relatório longo demais adia o momento em que o trabalho recomeça.

### **Uma interrupção não escolhe a hora**

No começo eu imaginava a retomada como consequência de um encerramento limpo. Uma sessão terminaria, seu estado seria cuidadosamente resumido e a próxima ação anotada.

A prática raramente segue esse roteiro.

- Começa uma reunião.
- A bateria acaba.
- O dia termina mais cedo do que o previsto.
- Outra urgência toma o lugar do assunto.

O momento em que essa memória é mais útil é justamente aquele em que não se preparou o fim.

O Contexto mantém, portanto, sua capacidade de retomada durante o trabalho. As decisões acham seu lugar quando são tomadas, as tarefas quando são formuladas, as fontes quando entram.

Retomar vira uma propriedade do sistema em vez de uma disciplina heroica do usuário.

O humano mantém, ainda assim, um papel decisivo. Ele pode julgar que a próxima ação proposta já não vem primeiro, que uma decisão antiga precisa ser reaberta ou que um fato novo muda o objetivo. A memória não congela a trajetória. Ela torna visível a trajetória a partir da qual se decide mudá-la.

As cinco frases que eu redigitava ao reabrir minha conversa, não redigito mais. Elas estão escritas em algum lugar, e alguém que não sou eu as relê no meu lugar no começo de cada sessão.

---

#### **PARA LEMBRAR — O CHAT, O AMBIENTE E A RETOMADA**

O chat é o lugar da troca; o Contexto é o lugar do que deve durar. Uma conversa guarda suas informações na ordem em que foram ditas; um trabalho exige outra organização. Continuar é conhecer a ação seguinte; retomar é reencontrar o sentido do trabalho antes de realizá-la. Uma interrupção não avisa: a capacidade de retomar se mantém durante o trabalho, não na hora de encerrá-lo.

---

## Dois clientes, um white paper e uma contradição a conservar

---

Até aqui descrevi o Contexto e vários efeitos que ele produz. Mas com o que ele se parece de verdade quando a gente abre? O que se vê nos arquivos? Como várias fontes fazem aparecer ideias que podem estar certas separadamente e ainda assim ser difíceis de conciliar?

O assunto desse Contexto inventado é simples: entender por que alguns técnicos entregam relatórios de serviço incompletos, antes de decidir se o produto deve mudar.

A missão dele cabe em poucas linhas:

*Entender o que degrada a qualidade dos relatórios de serviço. Distinguir as dificuldades de preenchimento, as restrições de organização e as necessidades que vêm do uso dos dados. Não escolher uma solução antes de ter confrontado várias fontes.*

Essa missão dá uma direção ao trabalho. Ela pede para entender um problema, ainda não para desenhar uma funcionalidade.

### **Um primeiro depoimento revela uma ideia**

O primeiro documento colocado no Contexto é a transcrição de uma entrevista com uma responsável de operações.

Ela explica:

*“Os técnicos preenchem os relatórios muitas vezes na sexta-feira. Em campo, o formulário lhes parece longo demais. No fim da semana já esqueceram parte dos detalhes.”*

2

A IA guarda essa entrevista como **fonte**. Depois extrai uma ideia e a registra numa ficha separada, o que eu chamo de **nota atômica**:

***Um preenchimento exigente demais pode levar alguns técnicos a adiar o relatório e a perder detalhes.***

Essa formulação não diz que todos os técnicos agem assim nem que o tamanho do formulário seja a única causa. Ela guarda o centro do depoimento: na organização descrita, o preenchimento é adiado e parte da informação desaparece entre o serviço e a redação.

A fonte continua disponível. A nota não a substitui; torna a ideia principal dela mais fácil de reencontrar e de confrontar com o que vier depois.

### **O segundo depoimento enriquece a mesma nota**

Algumas semanas depois, uma segunda entrevista entra no Contexto.

Dessa vez um responsável de operações conta:

*“Tornamos doze campos obrigatórios para melhorar a qualidade dos dados. Agora alguns técnicos escrevem ‘nada a relatar’ ou põem um traço só para conseguir validar.”*

Esse depoimento não descreve exatamente o mesmo comportamento. Os técnicos da primeira entrevista adiam o preenchimento. Os da segunda preenchem o formulário, mas contornam o que ele exige.

As duas fontes tratam, mesmo assim, da mesma ideia central: uma restrição pensada para obter mais informação pode acabar reduzindo a qualidade real do que é digitado.

A segunda entrevista se junta, então, à primeira **nota atômica**. Ela não substitui a primeira fonte nem cria uma ideia concorrente. Ela enriquece a nota ao fazer aparecer outra forma do mesmo problema: depois do preenchimento adiado, a completude de fachada.

A nota já pode afirmar:

***Um preenchimento exigente demais pode levar alguns técnicos a adiar ou a contornar o relatório, em prejuízo da informação recolhida.***

São duas fontes referenciadas. Cada uma guarda seu papel e seus limites. A nota fica mais sólida sem pretender que dois depoimentos bastem para medir o alcance do fenômeno.

### **O white paper revela outra ideia**

Acrescento depois um white paper dedicado ao uso dos dados das intervenções.

O documento defende outra exigência. Para entender as falhas recorrentes, comparar equipamentos e preparar certas auditorias, uma empresa precisa de relatórios suficientemente estruturados. Se cada técnico escreve o que quer num campo livre — ou não preenche nada —, o preenchimento talvez fique mais fácil, mas os dados perdem boa parte da utilidade coletiva.

Essa **fonte** não torna menos verdadeiros os depoimentos anteriores. Os técnicos descritos continuam adiando ou contornando. O PDF revela outra realidade: eliminar toda restrição também teria um custo.

A IA poderia se sentir tentada a fundir esses elementos na primeira nota e a reescrevê-la como um meio-termo geral. Seria um erro. O white paper não matiza o fato observado nas entrevistas; traz outra proposição.

Ele produz, então, uma segunda **nota atômica**:

***Relatórios estruturados são necessários para comparar as intervenções e explorar os dados produzidos por elas.***

O Contexto contém agora duas ideias distintas:

- a restrição de preenchimento pode degradar a informação realmente recolhida;
- a ausência de estrutura pode degradar o uso posterior dessa informação.

As duas notas podem ser verdadeiras ao mesmo tempo. Ainda assim puxam a decisão de produto em direções opostas.

### **A oposição deve continuar visível entre as notas**

O papel do Contexto não é alterar a primeira nota até a contradição sumir. Ele precisa guardar as duas proposições e tornar explícita a relação entre elas.

Uma **nota temática** pode então expressar o que se compreende ao ler as duas em conjunto:

***A qualidade de um relatório depende de uma tensão entre o esforço pedido no momento do preenchimento e a estrutura necessária ao seu uso.***

Essa terceira nota não é uma média diplomática. Ela não substitui nenhuma das duas notas atômicas. Ela mostra o problema novo que aparece quando as lemos juntas.

A contradição também passa a fazer parte do estado do trabalho: como obter informações estruturadas sem levar as pessoas a adiar o preenchimento ou burlar as exigências? Nenhuma das três fontes permite responder ainda.

### **É isto que se vê na tela**

O Contexto usado neste exemplo existe como uma pequena pasta de arquivos. A estrutura dele não é o conceito em si: é a minha maneira atual de lhe dar uma forma concreta.

```
contexte-qualite-comptes-rendus/  
├── 00-frame/  
│   ├── mission.md  
│   └── directives.md  
├── 01-state/  
│   ├── state.md  
│   └── contradictions.md  
├── 03-sources/  
│   ├── SRC00001-entretien-exploitation.md  
│   ├── SRC00002-entretien-exploitation.md  
│   └── SRC00003-livre-blanc-donnees-intervention.md  
├── 04-notes/  
│   ├── atomic/  
│   │   ├── AT000001-la-contrainte-de-saisie.md  
│   │   └── AT000002-des-comptes-rendus-structures.md  
│   └── thematic/  
│       └── THM00001-la-qualite-depend-de-la-saisie.md  
└── 05-outputs/  
    └── questions-prochain-entretien.md
```

Agora você pode ligar as palavras usadas desde o começo do livro a objetos visíveis.

- A **missão** explica por que a pasta existe.
- As **fontes** guardam o que chegou antes da interpretação.
- As **notas atômicas** firmam ideias distintas.
- A **nota temática** torna inteligível a tensão entre elas.
- O **estado** diz o que essa tensão deixa em aberto.
- O **entregável** prepara a ação seguinte.

Nada disso exige um software misterioso. O VS Code (mas poderia ser o aplicativo Codex ou o Claude Cowork) é aqui a janela pela qual acesso o Contexto. O Contexto mesmo está nos arquivos, na função de cada um e nas relações guardadas entre eles.

## A resposta se apoia no trabalho realizado

Alguns dias depois retomo o trabalho e proponho à IA:

*“Talvez devêssemos tornar todos os campos opcionais.”*

Numa conversa isolada, a IA poderia defender essa ideia, propor uma interface mais leve ou listar as vantagens dela. No Contexto, ela encontra as duas notas atômicas e a tensão que as liga.

Ela pode então responder:

*Essa proposta provavelmente reduziria parte do atrito descrito nas entrevistas. Ela se choca, porém, com a necessidade de dados estruturados apontada no white paper. As fontes mostram duas restrições reais; ainda não permitem escolher entre simplificação, preenchimento por etapas, ditado por voz ou adaptação dos campos ao tipo de serviço.*

Essa resposta não vem de um conhecimento geral de product management. Ela se apoia no trabalho em curso. A IA sabe que pergunta estamos tentando entender, que ideias são sustentadas por que fontes e que contradição continua aberta. Ela não decide por mim. Ela apenas impede que minha nova intuição apague uma das realidades já estabelecidas.

## O primeiro entregável prepara a fonte seguinte

**O estado** do Contexto não diz: “desenhar um formulário novo”.

Ele separa o que parece firmado do que continua incerto.

**O que parece firmado** Restrições de preenchimento podem levar ao adiamento ou a formas de burlar as exigências. Ainda é preciso manter estrutura suficiente para usar os relatórios.

**O que continua aberto** Ainda não sabemos que atrito pesa mais nem que informações precisam permanecer estruturadas de todo jeito.

**Próxima ação** Preparar uma entrevista que distinga as causas do problema em vez de apresentar logo uma solução.

O primeiro entregável é, então, modesto: um guia de perguntas para as novas entrevistas. Ele deve ajudar a entender quando o relatório é preenchido, quais são as restrições em campo, a utilidade real dos diferentes campos e como as exigências são contornadas.

O entregável não resolve a contradição. Ele prepara as informações que talvez permitam resolvê-la mais adiante.

### **Um objeto de trabalho, não uma máquina de fundir tudo**

Este exemplo mostra também um erro que a IA pode cometer: querer encaixar uma fonte nova numa nota existente porque os assuntos se parecem.

Uma **nota atômica** pode crescer quando várias fontes tratam da mesma proposição. É o que acontece com as duas entrevistas. Mas o white paper não é uma prova nova dessa

proposição. Ele revela uma ideia diferente, que merece uma nota própria.

A distinção conta. Se cada contradição reescreve a nota anterior, *o conjunto de ideias* acaba aplainando as tensões. O conhecimento parece coerente, mas já não corresponde ao que as fontes trouxeram de fato.

O Contexto não garante que a IA escolha sempre a fronteira certa. O humano precisa poder recusar uma fusão, pedir uma nota nova e conferir a relação criada entre elas. A estrutura torna essa correção possível e duradoura; ela não a decide sozinha.

Agora você viu o objeto:

- **fontes** que ficam intactas,
- **notas atômicas** que guardam cada uma seu centro,
- uma **nota temática** que aproxima sem fundir,
- uma **contradição** que continua aberta e um **entregável** que prepara o que vem.

Os capítulos seguintes poderão voltar aos usos do Contexto e depois abrir com mais detalhe a arquitetura dele a partir desses elementos agora visíveis.

---

**PARA LEMBRAR — UMA CONTRADIÇÃO SE CONSERVA**

Duas fontes podem ser verdadeiras juntas e puxar a decisão em direções opostas. Uma nota atômica cresce quando uma fonte reforça a mesma proposição; uma proposição diferente merece a própria nota. Fundir para fazer a tensão sumir produz um conhecimento coerente que já não corresponde às fontes.

---

# Parte 3

## Três usos concretos

---

As duas partes anteriores disseram o que o Contexto muda. Esta parte o mostra em funcionamento, em três usos que pratico de verdade.

Responder a um questionário do cliente, escrever um artigo a partir de um pensamento ainda informe, amadurecer um tema de produto no meio de um dia fragmentado. Três durações, três tipos de material, três destinatários, e o mesmo movimento a cada vez: reencontrar o que já foi entendido, transformá-lo e depois produzir sem esgotar o que serviu.

## Responder a um questionário do cliente sem perder o trabalho feito

---

O primeiro questionário do cliente que tratei com este sistema tinha quarenta e três perguntas.

Era preciso responder sobre a arquitetura do produto, sua segurança, os backups, a continuidade do serviço e a proteção dos dados. Parte das informações estava na nossa documentação. Outras estavam enterradas em arquivos de configuração. Para algumas era preciso recorrer ao time que operava o serviço.

Tivemos de reunir sete fontes técnicas e fazer uma ida e volta com o time. Esse trabalho se somava à redação propriamente dita: entender o que o cliente potencial pedia, reencontrar a informação, conferir se ainda era verdade, decidir o que podíamos comunicar e depois devolver a resposta ao formato esperado.

Não era uma situação excepcional. Cada novo cliente potencial mandava o próprio documento. Uma vez era uma planilha; outra, um arquivo Word ou um PDF. Os repositórios de referência mudavam, as frases também, mas voltavam os mesmos assuntos.

E, mesmo assim, toda vez tínhamos a impressão de recomeçar. Responder levava de três a seis horas e ocupava

várias pessoas, sem nenhuma garantia de que o cliente potencial virasse cliente. Depois o documento saía e o trabalho ficava trancado no arquivo enviado.

Meu primeiro objetivo era, então, simples: quando chegasse o questionário seguinte, eu não queria partir do zero de novo.

### **Uma resposta plausível, mas falsa**

A solução mais evidente era guardar os questionários antigos e reaproveitar as respostas deles.

Descobrimos logo por que isso não bastava.

Alguns documentos recebidos já traziam respostas. Elas podiam vir de um modelo, de outro fornecedor ou de uma troca anterior. Num deles, uma resposta dizia que a certificação ISO 9001 era de 2017. A frase parecia perfeitamente aproveitável. Era precisa, já escrita e colocada diante da pergunta certa.

A data correta era 2018.

O erro se resumia a um ano, mas revelava um problema bem maior. Uma resposta antiga se parece com um conhecimento. Ela tem uma forma afirmativa e tranquilizadora. Quando uma IA a encontra num documento, pode retomá-la sem esforço. E, no entanto, a presença dela não diz quem a escreveu, nem em que se apoia, nem se descreve realmente o nosso produto. Entendi então que eu não devia construir uma biblioteca de respostas para copiar. Era preciso construir uma memória do que sabíamos responder de verdade.

O questionário recebido guardava um valor: mostrava o que o cliente queria saber. Mas sozinho não podia provar a resposta. Essa distinção virou uma regra do Contexto: o documento do cliente guarda a pergunta; a resposta precisa continuar ligada a uma informação que tenhamos conferido.

### **Guardar os fatos em vez dos formulários**

Começamos a extrair dos questionários o que pudesse servir ao seguinte.

3

As perguntas já não eram arrumadas só por cliente ou por número. Nós as agrupávamos por assunto: backups, autenticação, criptografia, subcontratados, hospedagem, continuidade do serviço. Uma pergunta chamada “14.6.l” num cliente potencial podia assim se juntar a uma pergunta chamada “CL-072” em outro, se as duas tratassem do mesmo assunto.

Para cada um desses assuntos, o Contexto guardava os fatos validados e a origem deles. A resposta podia se apoiar numa documentação oficial, numa configuração conferida, num teste ou na confirmação do time competente. Guardávamos também a data da validação e os limites do que podíamos declarar.

Esse trabalho nem sempre produzia uma resposta completa. Ele também fazia aparecer os buracos.

Quando faltava uma informação, o sistema tinha de dizer. Ele preparava uma pergunta precisa para o time envolvido e indicava que linhas do questionário essa resposta permitiria

completar. Uma informação ausente deixava assim de se esconder atrás de uma frase plausível. Ela virava uma ação identificável.

Por fim, separávamos o que servia para conferir uma resposta do que podia ser enviado ao cliente. O detalhe de uma configuração podia ser indispensável a quem revisava por dentro e não ter lugar no documento público. O Contexto preparava, então, a formulação destinada ao cliente potencial e guardava ao lado as provas e as ressalvas necessárias para aprová-la.

Aos poucos o objetivo mudou. Já não se tratava apenas de terminar o questionário presente. **Cada resposta conferida devia reduzir o trabalho necessário para os seguintes.**

### O quarto questionário

O quarto questionário tinha cento e quinze perguntas, bem mais que o primeiro.

Dessa vez não partíamos mais de um conjunto de arquivos espalhados. O Contexto já conhecia os assuntos recorrentes e sabia onde reencontrar os fatos correspondentes. Noventa e seis respostas completas puderam ser produzidas direto a partir da memória acumulada e das fontes do produto, sem nova coleta.

O sistema também classificou o resto do documento. Algumas respostas eram sólidas, mas pediam uma última conferência. Outras continuavam incertas. Algumas não podiam ser dadas com as informações disponíveis.

Sobretudo, os buracos tinham mudado de natureza. Já não passávamos horas procurando a esmo quem sabia o quê. Os pontos ainda abertos correspondiam a decisões comerciais — por exemplo a duração de um compromisso — ou a confirmações técnicas pontuais. O trabalho restante era visível, atribuível e limitado.

Esse resultado me permitiu ver o que significa de verdade responder “automaticamente” a um questionário. Não se trata de pedir a uma IA que preencha todas as casas com segurança. Trata-se de preparar na hora o que der para preparar, de mostrar quanta confiança cada resposta merece e de isolar com clareza o que ainda exige um julgamento humano.

### **Dez minutos não quer dizer cem por cento de certeza**

No meu uso, a preparação de um questionário passou de três a seis horas para uns dez minutos para produzir 80 a 90 % do documento.

Esse número poderia dar uma impressão enganosa. Ele não significa que 80 a 90 % das respostas fiquem na hora certas e prontas para comprometer a empresa. Significa que o documento está em boa parte preparado: as respostas conhecidas estão postas, as fontes reencontradas, os pontos frágeis sinalizados e as informações que faltam transformadas em pedidos precisos.

A releitura humana continua indispensável. Ela cai, isso sim, nos lugares certos. Em vez de gastar tempo vasculhando

---

arquivos antigos e incomodando outra vez as mesmas pessoas, quem relê pode conferir os compromissos, decidir o que será divulgado e se concentrar nas respostas novas ou incertas.

O ganho mais importante não é, portanto, só a passagem de várias horas para alguns minutos. É a mudança de natureza do trabalho.

O primeiro questionário tinha exigido uma investigação. Os seguintes se beneficiaram de tudo o que essa investigação permitiu aprender. Um erro de data virou uma regra de prudência. Uma confirmação obtida de um time virou um fato reaproveitável. Uma pergunta sem resposta virou uma falta conhecida em vez de um esquecimento.

O cliente potencial continua recebendo o documento dele. A empresa, por sua vez, já não perde o que aprendeu ao prepará-lo.

---

#### **PARA LEMBRAR — UMA RESPOSTA VALE PELA PROVA DELA**

Uma resposta antiga se parece com um conhecimento: é afirmativa, bem formada, e cala sobre a origem e a data dela. Guardar os fatos conferidos e a procedência deles, e não os formulários já preenchidos. Uma falta explícita vale mais que uma frase plausível: ela vira uma pergunta dirigida a alguém.

---

## Escrever um artigo a partir de um pensamento ainda informe

---

Os artigos que publico no meu blog raramente começam diante do computador, numa página em branco cuidadosamente preparada.

Uma ideia aparece depois de uma dificuldade, de uma ferramenta testada, de uma conversa ou de uma mudança de opinião. Às vezes chega andando, no transporte público ou no meu sofá. Sinto que ela merece um texto, mas ainda não conheço nem o ângulo exato nem a ordem em que terá de ser contada.

Nessa hora costumo pegar o celular.

Não o uso só para gravar uma nota de voz ou ditar um texto que vou copiar depois. **Abro uma conversa com a IA.** Ela me escuta, me responde e me provoca enquanto eu penso. Essa conversa vai para o mesmo ambiente de trabalho que vou reencontrar no computador. Posso, então, começar longe da escrivadinha sem criar uma nota solta que depois teria de ser transferida e reinterpretada.

### Descarregar um pensamento enquanto ele se forma

Crio um Contexto com um objetivo simples: escrever o artigo.

Depois dou a ele tudo o que tenho na cabeça.

Às vezes durante duas horas falo de um primeiro ponto, depois de outro, e volto ao primeiro. Conto uma experiência, salto para uma conclusão, acrescento um detalhe, abro um caminho novo e largo o anterior por um tempo. Posso repetir a mesma ideia quatro vezes porque ela me volta em formas diferentes. Posso também afirmar uma coisa e dizer o contrário uma hora depois, porque meu pensamento se deslocou durante a conversa.

Essa desordem não é um acidente que se devesse remover antes de começar. Costuma ser a forma real de uma ideia enquanto ela ainda está nascendo.

Conversar com a IA me permite seguir sem impor cedo demais uma estrutura. Uma resposta me faz detalhar uma lembrança. Uma pergunta revela que acabei de pular uma etapa. Uma reformulação me mostra que o que eu achava óbvio ainda não é. Corrijo, matizo e sigo em outra direção. O Contexto guarda essas palavras como fonte de sessão. Ele não retém só as frases mais limpas. Guarda também as voltas, as hesitações e as contradições que vão permitir ver como o pensamento se deslocou.

A IA aproxima depois as quatro formulações de uma mesma ideia em vez de tratá-las como quatro ideias diferentes.

Aponta duas afirmações incompatíveis, reencontra a nuance que chegou uma hora mais tarde e junta um exemplo ao princípio que ele ilumina. **Ela não faz a desordem sumir: torna-a trabalhável.**

Essa primeira etapa me livra de uma escolha prematura. Ainda não preciso decidir que frase vai abrir o artigo, se tal ideia merece uma seção ou em que ordem contar os fatos. O que busco primeiro é esvaziar o que tenho na cabeça, sem deixar escapar as ideias que só a atravessam.

O Contexto, isso sim, sabe por que aquele material está ali. Ele não recebe uma conversa sem destino. Recebe o material ainda desordenado de um artigo a construir.

3

### **Usar a IA como parceira de amadurecimento**

Depositado esse primeiro material, pergunto à IA o que falta. Ela detecta uma afirmação rápida demais, um pressuposto nunca explicado, uma passagem que só se sustenta se o leitor já conhece minha experiência. Pede um exemplo. Aproxima duas ideias formuladas em momentos diferentes. Às vezes propõe um caminho que parece plausível e que eu recuso porque não corresponde ao que vivi.

Respondo, preciso, corrijo. Essa conversa ainda não produz o artigo. Ela adensa o material dele.

O papel da IA é importante, mas o lugar dela continua claro. Ela dispõe de tudo o que despejei e procura ali tensões mais rápido do que eu. Em compensação, ignora o que eu quero defender de verdade enquanto eu não tiver formulado. Ela abre a pergunta; eu decido se essa pergunta merece deslocar o texto.

Esse diálogo encurta o tempo necessário para esbarrar numa fraqueza. Sozinho, eu teria descoberto dois dias depois que

faltava um exemplo ou que minha conclusão se apoiava numa etapa nunca explicada. A IA provoca essa descoberta enquanto a ideia ainda é fácil de refazer.

### **Acrescentar fontes quando a experiência já não basta**

Meus artigos partem muitas vezes do que vivi. O material externo guarda, porém, o seu papel:

- uma referência dá nome ao que observei;
- uma documentação confere um funcionamento;
- outro autor defende uma posição vizinha, ou mostra um limite que eu não tinha encontrado;
- uma busca adicional revela que minha experiência cobre apenas um caso particular.

Esses documentos entram no `Contexto` como **fontes documentais**. O papel deles difere do das minhas primeiras palavras. A **fonte de sessão** guarda o que afirmo e o que entendo da minha experiência. As **fontes documentais** trazem um material que posso confrontar com essa experiência.

O artigo pode então distinguir o que observei, o que uma fonte estabelece e o que continua sendo uma interpretação. Essa distinção protege o texto de uma tentação frequente: apresentar uma experiência pessoal como regra geral.

## Transformar as fontes em ideias reaproveitáveis

Nessa altura o Contexto não passa direto das fontes para o roteiro do artigo. Ele transforma primeiro o material bruto em **notas atômicas**.

Cada nota firma uma ideia autônoma vinda da conversa ou de um documento:

- uma afirmação que quero defender;
- um exemplo que lhe dá forma concreta;
- uma nuance que limita o alcance dela;
- uma objeção a que o artigo terá de responder;
- um fato externo que confirma ou contradiz minha experiência.

A nota guarda a origem dela. Posso assim entender a ideia sem reler toda a conversa e depois voltar à fonte se precisar conferir uma formulação ou reencontrar o contexto dela.

Essa transformação muda a maneira de preparar o artigo.

- A conversa seguia o movimento do meu pensamento; as notas agrupam agora o que fala da mesma ideia, mesmo quando esses elementos apareceram com uma hora de diferença ou em duas fontes distintas.
- Uma nota central pode fazer aparecer o ângulo do artigo.
- Outras lhe trazem exemplos, provas, objeções ou consequências.

Essas notas ainda não são nem parágrafos nem um roteiro pronto. **Elas continuam neutras diante do entregável: a**

**mesma ideia poderá alimentar este artigo, uma apresentação ou outro trabalho.**

### **Passar pelo roteiro antes de redigir**

Quando o material fica suficiente, peço um roteiro.

Essa etapa me impede de passar direto de uma conversa rica para um texto que apenas seguisse a ordem dela. Uma discussão avança por voltas, correções e associações. Um artigo precisa conduzir um leitor que não conhece nem a discussão nem a história dela.

O roteiro fixa essa progressão. Diz o que precisa ser entendido primeiro, onde pôr o exemplo principal, que objeção tratar e o que a conclusão pode de fato afirmar.

A IA propõe uma estrutura a partir do material disponível. Eu a desloco, aperto, recuso as partes que dão peso demais a um caminho secundário. Duas ou três trocas bastam muitas vezes para obter uma direção clara.

A redação vem depois.

Nessa altura a IA não cria o texto a partir de uma instrução vaga. Ela trabalha com um corpus de ideias e de temas extraídos da minha experiência contada, com respostas a perguntas, com as fontes e com um roteiro validado. Agrupa as repetições, recoloca os exemplos e transforma uma fala sinuosa em prosa legível.

Releio o resultado procurando três desvios precisos:

- uma nuance que sumiu;
- uma passagem que ficou afirmativa demais;

- uma formulação que trai a ideia.

Corrijo e decido depois a versão publicada.

### **O que o Contexto traz a um trabalho curto**

Um artigo pode ser escrito em um ou dois dias. Ele não parece precisar da mesma memória que um produto acompanhado por anos ou que um livro de dezenas de capítulos.

É justamente esse caso leve que mostra que a duração não é a única coisa em jogo.

O Contexto mantém juntas as primeiras palavras, as fontes acrescentadas, as perguntas feitas, o roteiro escolhido e o texto completo. Se o trabalho para, reencontro o estado dele. Se uma frase fica discutível, posso voltar à origem dela. Se uma ideia que apareceu durante a escrita ultrapassa o artigo, ela pode ser firmada à parte e alimentar um trabalho futuro. Nem todo esse material merece necessariamente ser guardado para sempre. Um Contexto pequeno pode alcançar seu objetivo, produzir seu artigo e depois ser arquivado. O valor dele está na continuidade que deu ao trabalho enquanto ele se formava e no rastro que deixa se mais adiante for preciso entender como o texto tomou aquela direção.

A publicação em si pode disparar outras tarefas: preparar um resumo, adaptar o texto a vários idiomas ou produzir o que é necessário para colocá-lo no ar. Essas operações prolongam o mesmo resultado, sob regras que o Contexto já conhece.

---

O movimento completo continua simples: uma experiência vira fala, essa fala encontra perguntas e fontes, o material vira um roteiro, e depois o roteiro vira um texto.

A IA acelera cada transformação, e o Contexto segura o fio que leva de uma à outra.

Nada desse movimento pertence só à escrita longa. Ele começa assim que uma ideia merece mais do que uma nota de voz esquecida num celular.

---

#### **PARA LEMBRAR — A DESORDEM É MATERIAL**

Um pensamento que nasce se repete, se contradiz e volta atrás: essa é a forma normal dele, não um defeito a corrigir antes de começar. O Contexto guarda essas palavras como fonte de sessão e depois as torna trabalháveis. O roteiro se intercala entre a conversa e o texto, porque um leitor não segue a ordem de uma discussão.

---

## Amadurecer um tema de produto no meio do caos

---

Os dois capítulos anteriores seguiram um único assunto de cada vez, do começo ao fim. Este capítulo descreve como é de verdade um dia comum de um product manager: vários assuntos, pedidos e ritmos avançam ao mesmo tempo.

Sou o único product manager da minha empresa. Isso quer dizer que boa parte das informações ligadas ao produto acaba passando na minha frente.

Num mesmo dia, um cliente diz uma frase que me parece importante.

- Ela cabe em poucas palavras.
- Ainda não sei se revela um problema compartilhado por outros usuários ou só uma dificuldade específica, mas não quero perdê-la.

Um pouco depois pego um white paper sobre a evolução de uma questão do ofício.

- Dessa vez o material já não cabe numa frase, e sim num PDF de várias dezenas de páginas.
- Ainda não sei o que vai me ser útil nem se esse documento vai confirmar o que observo em campo.

Depois os desenvolvedores me falam de uma funcionalidade em construção.

- É preciso responder a uma pergunta, esclarecer um comportamento ou voltar a uma hipótese assumida algumas semanas antes.

Enquanto isso, o CPO quer fazer um balanço de duas apostas ligadas a outra parte do produto.

Uma conversa com o suporte revela enfim que falta uma documentação funcional.

- Parte dessa documentação ainda só existe na minha cabeça.

No fim do dia passei por três ou quatro assuntos diferentes (ou mais). Recebi uma citação de poucas palavras, um documento inteiro, uma pergunta sobre um desenvolvimento em curso, um pedido de decisão e uma necessidade de documentação.

**Não é um dia especialmente ruim: é um dia comum.**

### **Entradas que não têm nem o mesmo tamanho nem o mesmo destino**

O caos do product management não vem só da quantidade de informação. Vem da diferença de natureza dela.

Cada uma dessas cinco entradas olha para um momento diferente da vida do produto:

- uma frase de cliente é um sinal fraco, do qual ainda não sei se anuncia outros;
- um PDF dá uma visão geral de um ofício;

- uma discussão com os desenvolvedores trata do que estamos construindo agora;
- um pedido do suporte trata do que já existe e precisa ser explicado;
- uma pergunta do CPO obriga a comparar vários futuros possíveis.

Esses elementos não foram feitos para entrar num documento único.

3

- Eles não falam todos do mesmo assunto e não se situam no mesmo momento da vida do produto.

Se eu os reúno simplesmente numa lista de notas, guardo as informações, mas adio o trabalho de compreensão para o dia em que precisar dele.

Os Contextos me permitem manter vários fios em paralelo.

- A citação se junta ao assunto a que parece pertencer, mesmo que o significado dela continue incerto.
- O PDF é guardado com a origem dele e pode ser analisado para extrair as poucas ideias (**notas atômicas**) realmente úteis.
- A conversa com os desenvolvedores se junta ao trabalho sobre a funcionalidade em curso.
- A falta apontada pelo suporte vira um trabalho a fazer em vez de continuar uma observação ouvida entre duas reuniões.

Nem tudo é transformado logo em conhecimento:

- a frase do cliente talvez continue ambígua,

- o white paper pouco pertinente,
- a intuição desmentida na semana seguinte.

O Contexto não dá inteligência ao que entra. Ele lhe dá um lugar, uma origem e uma chance de ser reencontrado na hora certa.

**O Contexto contém o caos sem pretender aboli-lo.**

### **Enquanto as informações entram, os pedidos de saída continuam**

O dia não se resume a receber e arquivar material: também é preciso produzir.

O suporte me pede, por exemplo, uma documentação funcional sobre uma funcionalidade em que ainda estamos trabalhando.

- É preciso explicar como ela funciona, detalhar os casos importantes e dar a quem acompanha os clientes o necessário para responder direito.

No mesmo dia posso querer terminar um artigo para o blog da empresa.

- Esse artigo não trata necessariamente da mesma funcionalidade.
- Pode, porém, tratar do mesmo tema de ofício: o trabalho a que o produto responde, uma dificuldade recorrente dos usuários ou uma mudança nas práticas deles.

Os dois documentos recorrem então a parte do mesmo conhecimento sem contar a mesma coisa.

- A documentação destinada ao suporte explica o produto como ele funciona.
  - Ela tem de ser precisa, operacional e ajustada às perguntas dos usuários.
- O artigo desenvolve uma experiência, um problema de ofício ou uma reflexão.
  - Ele pode usar algumas das mesmas ideias, mas as organiza para outro leitor e com outro fim.

## 3

Não preciso reconstruir duas vezes tudo o que sei do assunto.

- O Contexto guarda um material intermediário (as **notas atômicas** e as **notas temáticas**) que ainda não é nem a documentação nem o artigo.
- A partir desse material, a IA pode preparar várias montagens: uma centrada no uso do produto, a outra na questão de ofício mais ampla.

Releio, claro, os dois resultados.

- Uma documentação para o suporte compromete o modo como o produto é entendido.
- Um artigo compromete meu ponto de vista.

Mas gasto meu tempo conferindo a substância e a orientação, em vez de reencontrar os mesmos elementos e escrevê-los de novo do zero.

Esse movimento pode acontecer no mesmo dia. Enquanto chegam informações novas, um conhecimento já organizado permite produzir os documentos de que a empresa ou eu precisamos.

### **Uma frase pequena pode virar um tema de produto**

A citação ouvida de manhã não dispara necessariamente uma decisão.

- Tomada sozinha, pode não provar nada.
- Outro cliente talvez expresse uma dificuldade parecida algumas semanas depois.
- Uma entrevista trará uma nuance.
- O white paper pego naquele dia talvez dê nome à mudança observada.
- Um dado do suporte mostrará que o problema volta mais vezes do que pensávamos.

Esses elementos chegam em momentos diferentes e sob formas diferentes.

- O Contexto permite aproximá-los sem apagar a origem deles.
- Ele pode apontar
  - que uma observação nova se parece com um sinal já guardado,
  - fazer aparecer uma contradição,
  - perguntar o que permitiria decidir entre duas interpretações.

Aos poucos, uma frase solta pode virar um tema que vale a pena trabalhar. Nessa hora eu já disponho de parte da história dela: os primeiros sinais, os documentos que a iluminam, o que os confirma, o que os contradiz e as perguntas ainda abertas.

O Contexto ajuda esse material a amadurecer.

- Ele não decide que o problema é importante nem escolhe a solução por mim.

A aposta de produto continua sendo responsabilidade minha.

- Em compensação, posso tomar essa decisão com algo mais que uma lembrança vaga ou uma série de documentos espalhados.

O mesmo sistema acompanha depois o trabalho:

- acrescentar entrevistas,
- confrontar uma hipótese com os fatos,
- preparar cenários,
- produzir uma argumentação para o CPO,
- escrever especificações,
- ...

Um sinal pequeno captado no meio de um dia pode assim continuar vivo tempo suficiente para virar uma decisão real.

### **Uma ideia abandonada pode continuar amadurecendo**

Nem toda exploração chega à produção.

Já trabalhei em usos da inteligência artificial que pareciam interessantes mas que a tecnologia disponível ainda não permitia realizar direito.

- A ideia não era necessariamente ruim.
- O momento era.

Num funcionamento clássico, a exploração teria produzido alguns documentos e depois teria sumido numa pasta depois da decisão de abandoná-la.

- Um ou dois anos depois, teria sido preciso reencontrar esses arquivos, lembrar as hipóteses de partida e entender por que tínhamos parado.

Com um Contexto, a exploração pode ser arquivada junto com o percurso dela: os sinais iniciais, os cenários imaginados, as tentativas, os limites encontrados e a razão da parada.

- Quando a tecnologia muda, posso retomar esse trabalho sem transformar um antigo resultado negativo numa ideia definitivamente ruim.

Reencontro o que queríamos fazer, mas também por que não funcionava. Posso então conferir se o obstáculo ainda existe, guardar o que continua valendo e retomar a reflexão a partir dali.

O amadurecimento de um tema de produto não é, portanto, sempre um trajeto contínuo.

- Ele pode começar por uma frase, enriquecer-se durante meses, interromper-se e depois recomeçar quando o produto, o mercado ou a tecnologia mudou.

### **Fazer aparecer ordem sem frear o dia**

Meu dia continua fragmentado.

- Os clientes não coordenam as observações deles com os pedidos do suporte.
- Os desenvolvedores não escolhem a hora em que acabei de ler um PDF.
- As decisões a preparar não param enquanto escrevo uma documentação ou um artigo.

O Contexto não transforma essa realidade numa sucessão perfeita de tarefas.

Ele permite que cada fragmento guarde uma continuidade.

- Uma frase pequena e um documento longo podem entrar sem receber o mesmo tratamento.
- Vários assuntos podem avançar sem se confundir.
- Um conhecimento já trabalhado pode servir a vários entregáveis sem ficar trancado no primeiro.
- Uma ideia interrompida pode esperar sem perder toda a história dela.

É assim que o uso no product management: não para deixar meu dia menos agitado, mas para evitar que o movimento dele disperse o que eu aprendo.

O caos continua entrando.

O que ele já não faz é sair em forma de perda.

Esses três usos têm um ponto em comum que ainda não nomeei: cada um termina com algo que sai — uma resposta enviada, um artigo publicado, uma decisão tomada.

A parte seguinte olha o que fica quando o entregável já saiu.

---

**PARA LEMBRAR — VÁRIOS FIOS EM PARALELO**

Uma frase de cliente e um PDF de cinquenta páginas pedem um tratamento e um destino diferentes. O Contexto dá a cada fragmento um lugar, uma origem e uma chance de ser reencontrado na hora certa. Uma exploração abandonada guarda as razões da parada: ela volta a ser retomável quando a tecnologia ou o mercado muda.

---



# Parte 4

## O que os entregáveis deixam para trás

---

Cada um dos três usos anteriores termina com um entregável enviado. Esta parte olha o que fica quando ele já saiu.

Serve de fio uma mesma pergunta, feita duas vezes com seis meses de intervalo. Na primeira vez, a resposta cabe numa célula de planilha. Na segunda, essa resposta ainda existe mas já não basta: falta o raciocínio que a tinha produzido, e as relações entre os documentos que o sustentavam.

## 4.1

# A resposta cabe numa célula, o raciocínio não

---

Voltemos a um dos questionários da parte anterior.

A pergunta chega na célula D47 de uma planilha:

| *Os seus backups são criptografados em repouso?*

A resposta final vai caber em três linhas. Para escrevê-las, porém, preciso reunir vários elementos:

- nossa documentação descreve como os backups funcionam em geral;
- um arquivo de configuração indica a configuração realmente ativa;
- o time de Ops confirma como essa configuração se aplica em produção;
- o jurídico esclarece o que podemos comunicar a um cliente potencial.

Vou da planilha para a documentação, da documentação para a configuração, e depois de uma troca com o time de Ops para uma decisão jurídica. Volto enfim à célula D47 e escrevo ali a resposta.

O documento sai.

### **Três linhas no fim de uma cadeia**

**A célula contém o resultado do trabalho. Ela não mostra o caminho que o produziu.**

Ela não diz:

- que documentação consultei;
- que configuração conferi;
- quem confirmou que ela se aplicava;
- que formulação foi autorizada;
- que ressalvas foram tiradas do texto enviado ao cliente potencial.

Esses elementos existem, mesmo assim. Eles determinaram cada uma das três linhas e, enquanto respondo, a relação entre eles está clara na minha cabeça.

Uma vez enviado o documento, essa unidade desaparece. A planilha guarda a resposta, mas não o raciocínio que permitia confiar nela.

### **Cada ferramenta fez o trabalho dela**

O problema não vem de uma má arrumação.

- A documentação está mesmo no espaço documental.
- A configuração fica no produto.
- A resposta do time permanece na conversa em que foi dada.
- A decisão jurídica tem o próprio suporte.
- A célula D47 está no lugar dela no documento do cliente potencial.

Cada ferramenta guardou o objeto que lhe tinha sido confiado. Nenhuma tinha por função guardar o raciocínio inteiro.

Para reconstruí-lo é preciso saber que elementos procurar, onde procurá-los e que papel cada um desempenhou. Esse conhecimento continua com a pessoa que preparou a resposta.

### **O entregável achata o que o produziu**

Um entregável precisa apresentar um resultado ajustado ao destinatário dele.

- O cliente potencial espera uma resposta clara, não o diário completo da investigação feita para obtê-la.

Essa simplificação é necessária. Ela fica cara quando o entregável é o único rastro do trabalho.

O resultado final é sempre mais curto que o caminho que leva até ele: a resposta enviada, o artigo publicado ou a decisão de produto. Se só o resultado permanece, o trabalho seguinte precisa refazer parte desse caminho.

### **Dar um endereço ao raciocínio**

O Contexto não substitui nem a planilha, nem a documentação, nem a configuração, nem as trocas com os times.

- Ele dá um endereço comum ao trabalho que os liga.

A pergunta D47, os elementos consultados, as validações obtidas e a resposta produzida pertencem ao mesmo assunto.

---

Eles podem guardar a forma e o lugar de origem deles e continuar ligados dentro do Contexto.

A célula continua cumprindo a função dela: responder ao cliente potencial. O Contexto cumpre outra: guardar o que vai permitir entender, conferir e retomar essa resposta.

Seis meses depois, essa diferença fica decisiva.

---

**PARA LEMBRAR — O ENTREGÁVEL NÃO GUARDA O RACIOCÍNIO**

Uma resposta pode caber em três linhas e mesmo assim depender de várias fontes, validações e ressalvas. As ferramentas guardam cada uma o seu objeto; quem continua carregando as relações entre elas é o humano. O Contexto dá um endereço ao trabalho que liga esses elementos, sem substituir as ferramentas de origem deles.

---

## Seis meses depois, a resposta já não basta

---

Seis meses depois da célula D47, um novo cliente potencial faz a mesma pergunta sobre a criptografia dos backups.

- Reencontro a resposta antiga em dez segundos.
- Ela é clara, bem escrita e pronta para ser copiada.

E, mesmo assim, fico diante dela sem conseguir usá-la.

Bastam três perguntas para me travar:

- de onde vinha essa resposta: da documentação, da configuração observada ou do que o time de Ops tinha me explicado naquele dia?
- o jurídico tinha validado essa formulação exata ou uma versão mais prudente?
- a configuração ainda é a mesma hoje?

**A resposta sobreviveu; o que permitia confiar nela sumiu.**

### Reencontrar ainda não é retomar

A célula antiga me devolve um texto. Ela não me devolve:

- o nível de prova dele,
- os limites dele,
- nem as conferências necessárias antes de usá-lo de novo.

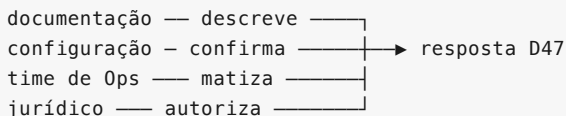
Sem outra memória, preciso refazer toda a investigação do capítulo anterior, na ordem inversa. Não parto exatamente do

zero, já que tenho uma resposta. Mas parto sem saber que parte do trabalho continua valendo.

É o erro da certificação datada de 2017 em vez de 2018, visto desta vez do lado das nossas próprias respostas: a frase é nítida, ela vem de nós, e a prova dela ficou invisível.

### Reencontrar o papel de cada elemento

No Contexto, a resposta antiga não fica sozinha. Ela guarda as relações de que o raciocínio que a produziu precisava:



4

Os verbos mudam o que posso fazer de cada elemento.

- A documentação descreve o funcionamento geral; sozinha, não prova a configuração ativa.
- A configuração confirma essa configuração num dado momento; não decide o que pode ser comunicado.
- O time de Ops traz uma nuance vinda da operação real.
- O jurídico autoriza uma formulação destinada ao cliente potencial.

Já não reencontro apenas quatro objetos próximos de uma resposta. Reencontro a função que cada um cumpria.

Essa precisão impede que uma fonte prove mais do que ela de fato estabelece. Mostra também que conferência precisa ser refeita quando passou tempo.

### **Conferir o que pode ter mudado**

O Contexto não transforma a resposta antiga em verdade permanente: ele prepara a conferência dela.

- A documentação pode continuar válida enquanto a configuração mudou.
- A configuração pode ter ficado idêntica enquanto a formulação autorizada precisa ser revista.
- Uma nuance trazida pelo time de Ops pode ter sumido depois de uma evolução do produto.

As relações tornam visíveis essas diferenças. Posso checar os elementos que podem ter mudado sem refazer a esmo toda a investigação.

A resposta final talvez seja idêntica à de seis meses atrás. A diferença é que sei por que ainda posso usá-la.

### **Guardar as relações que servem ao trabalho**

Não se trata de ligar todos os documentos entre si.

- Dois arquivos podem compartilhar uma palavra sem participar do mesmo raciocínio.

Uma relação merece ser guardada quando muda a maneira de entender, de conferir, de decidir ou de produzir.

- No caso de D47, bastam quatro relações: descrever, confirmar, matizar e autorizar.

Elas permitem responder a perguntas concretas:

- em que se apoia esta resposta?
- que parte precisa ser conferida hoje?
- quem tinha validado o que podíamos comunicar?

- 
- o que tornaria esta resposta caduca?

O Contexto não busca, portanto, fabricar um mapa total de tudo o que sei. Ele guarda a rede de relações necessária a um trabalho determinado.

### **Abrir a máquina**

O Livro I mostrou como o Contexto nasceu, o que ele muda na relação com a IA e como acompanha vários usos. O caso D47 faz aparecer o que falta entender.

- Como guardar a missão do trabalho sem repeti-la a cada sessão?
- Onde colocar as fontes, as ideias, as decisões e os entregáveis para que guardem papéis distintos?
- Como carregar o que é útil agora sem trazer todo o material para a conversa?

O Livro II vai abrir essa máquina. Ele vai acompanhar as informações ao longo de todo o percurso num Contexto: da chegada à transformação, ao que produzem e à retomada do trabalho.

---

### **PARA LEMBRAR — REENCONTRAR O QUE PERMITE CONFIAR**

Reencontrar uma resposta não basta: é preciso reencontrar o que ainda permite usá-la. Uma relação nomeada indica o papel preciso de uma fonte, de uma validação ou de uma ressalva. O Contexto prepara a conferência mostrando o que sustenta a resposta e o que pode ter mudado.

---



## LIVRO II

# Construir e manter vivo o Contexto

---

- 5 Por dentro de um Contexto
- 6 O Contexto no seu ecossistema



# Parte 5

## Por dentro de um Contexto

---

Até aqui o Contexto foi descrito pelo que ele permite fazer. Esta parte o abre.

Ela apresenta primeiro a árvore completa dele e depois acompanha uma sessão de trabalho do começo ao fim: abrir e reencontrar o ponto em que o trabalho está, fazer entrar material, transformá-lo em ideias, produzir um entregável, arbitrar e salvar para que a sessão seguinte recomece sem reconstruir nada.

O exemplo que atravessa estes seis capítulos é inteiramente fictício: entender por que alguns técnicos de campo entregam relatórios de serviço incompletos, antes de decidir se o produto deve mudar.

## O mapa completo

Na parte 2 deixei o Contexto dedicado à qualidade dos relatórios de serviço:

- três fontes,
- duas notas atômicas,
- uma nota temática,
- um guia para a próxima entrevista.

Na época eu olhava o que ele permitia entender. Agora abro a pasta em si.

Esta é a árvore completa da versão que uso neste livro:

```

contexte-qualite-comptes-rendus/
├─ README.md           navegação geral, gerado
├─ offline.md          a caixa de correio fora de sessão
├─
├─ 00-frame/           o que dá uma direção duradoura
│   ├── README.md      navegação da zona, gerado
│   ├── mission.md     o objetivo, a pergunta, os limites
│   ├── directives.md  as regras próprias da missão
│   └─ loading.md      o que será lido na abertura
├─
├─ 01-state/           o presente do trabalho
│   ├── README.md      navegação da zona, gerado
│   ├── state.md       pontos firmes, dúvidas, dívidas
│   ├── synthesis.md   o que se tornou duradouro
│   ├── todo.md        as ações a cumprir
│   ├── decisions.md   as escolhas que estruturam o Contexto
│   ├── contradictions.md o que continua em tensão
│   ├── handoff.md     como retomar o trabalho
│   ├── checkpoints.md a série compacta dos estados
│   └─ health.md       o resultado do último controle
├─
├─ 02-history/         o rastro factual e datado
│   └─ README.md       o papel desse rastro, gerado
└─

```

|                           |                                      |
|---------------------------|--------------------------------------|
| └─ YYYY-MM-DD.md          | os acontecimentos do dia             |
| └─ 03-sources/            | o que foi recebido ou dito           |
| └─ README.md              | a lista das fontes, gerado           |
| └─ ingestion-log.md       | o que foi lido, o que rendeu         |
| └─ documents/             |                                      |
| └─ README.md              | a lista dos documentos, gerado       |
| └─ raw/                   | os arquivos originais                |
| └─ fulltext/              | o texto integral aproveitável        |
| └─ SRC00001-....md        | identidade, procedência, estado      |
| └─ sessions/              |                                      |
| └─ README.md              | a lista das sessões, gerado          |
| └─ SESSION-....md         | o que foi falado durante o trabalho  |
| └─ 04-notes/              | o conhecimento tirado das fontes     |
| └─ README.md              | o mapa das notas, gerado             |
| └─ atomic/                | uma ideia por nota – AT000001-....md |
| └─ thematic/              | um ângulo que liga – THM00001-....md |
| └─ glossary/              | um termo fixado – GL000001-....md    |
| └─ decision-records/      | uma arbitragem – DRE00001-....md     |
| └─ 05-outputs/            | o que se produz para um uso          |
| └─ README.md              | a lista dos resultados, gerado       |
| └─ _draft/                | os entregáveis em curso              |
| └─ questions-entretien.md | um guia para continuar               |
| └─ note-de-cadrage.md     | uma síntese para decidir             |
| └─ presentation-equipe.md | os mesmos pontos, reformulados       |

Os números impõem uma ordem de leitura, não uma hierarquia de valor:

- o quadro e o estado orientam o trabalho;
- a história permite conferir como ele se desenrolou;
- as fontes guardam o material, as notas o transformam em conhecimento;
- os entregáveis lhe dão uma forma para um destinatário.

### Seis zonas, seis funções

00-frame/ responde à pergunta “por que estou aqui, e sob quais regras?”. Essa zona muda pouco, e com razão: mexer na

missão transformaria a minha maneira de julgar todas as fontes, todas as tarefas e todos os resultados.

```

contexte-qualite-comptes-rendus/
├── 00-frame/                o que dá uma direção duradoura
│   ├── README.md           navegação da zona, gerado
│   ├── mission.md          o objetivo, a pergunta, os limites
│   ├── directives.md       as regras próprias da missão
│   └── loading.md          o que será lido na abertura

```

01-state/ responde a “onde estou agora?”. Essa zona se move com o trabalho e me permite distinguir um ponto firme de uma hipótese, uma ação futura de uma decisão já tomada, uma dificuldade passageira de uma contradição que merece continuar à vista.

5

```

contexte-qualite-comptes-rendus/
├── 01-state/                o presente do trabalho
│   ├── README.md           navegação da zona, gerado
│   ├── state.md            pontos firmes, dúvidas, dúvidas
│   ├── synthesis.md        o que se tornou duradouro
│   ├── todo.md             as ações a cumprir
│   ├── decisions.md        as escolhas que estruturam o Contexto
│   ├── contradictions.md   o que continua em tensão
│   ├── handoff.md          como retomar o trabalho
│   ├── checkpoints.md      a série compacta dos estados
│   └── health.md           o resultado do último controle

```

02-history/ responde a “o que aconteceu de fato?”. Essa zona não narra a sessão com capricho: guarda os fatos úteis dela, uma fonte acrescentada, uma pista abandonada, uma decisão tomada, uma nota criada. Consulto pouco essa zona, mas ela se torna decisiva no dia em que o estado presente me parece incoerente.

```

contexte-qualite-comptes-rendus/
├── 02-history/                o rastro factual e datado
│   ├── README.md             o papel desse rastro, gerado
│   └── YYYY-MM-DD.md         os acontecimentos do dia

```

03-sources/ responde a “que material eu recebi?”. Um documento de fora entra congelado; a minha própria fala entra ao longo das sessões. Nos dois casos, a fonte continua separada do que o Contexto vai entender dela.

```

contexte-qualite-comptes-rendus/
├── 03-sources/                o que foi recebido ou dito
│   ├── README.md             a lista das fontes, gerado
│   ├── ingestion-log.md      o que foi lido, o que rendeu
│   ├── documents/
│   │   ├── README.md         a lista dos documentos, gerado
│   │   ├── raw/              os arquivos originais
│   │   ├── fulltext/         o texto integral aproveitável
│   │   └── SRC00001-....md    identidade, procedência, estado
│   └── sessions/
│       ├── README.md         a lista das sessões, gerado
│       └── SESSION-....md     o que foi falado durante o trabalho

```

5

A ficha SRC00001-....md não contém nem o documento nem o resumo dele. Ela responde a tudo o que preciso saber a respeito dele:

- de onde vem e por que eu o acrescentei;
- onde reencontrar o original e o texto integral;
- se a extração deu certo;
- que nível de confiança dar a ele;
- se a leitura dele já rendeu notas.

Ela torna visível, portanto, o estado da fonte sem tocar no material em si.

04-notes/ responde a “o que eu aprendi?”.

- Uma nota atômica fixa uma ideia.
- Uma nota temática liga várias ideias.
- O glossário detalha uma palavra.
- Um decision record guarda as razões de uma arbitragem.

```

contexte-qualite-comptes-rendus/
├── 04-notes/                o conhecimento tirado das fontes
│   ├── README.md           o mapa das notas, gerado
│   ├── atomic/             uma ideia por nota – AT000001-...md
│   ├── thematic/           um ângulo que liga – THM00001-...md
│   ├── glossary/           um termo fixado – GL000001-...md
│   └── decision-records/    uma arbitragem – DRE00001-...md

```

05-outputs/ responde a “o que vou fazer com esse conhecimento?”. Os rascunhos e os resultados vivem ali, na forma que o público deles espera:

- um guia de entrevista;
- um artigo;
- uma especificação;
- uma apresentação;
- uma decisão, entre outras formas possíveis.

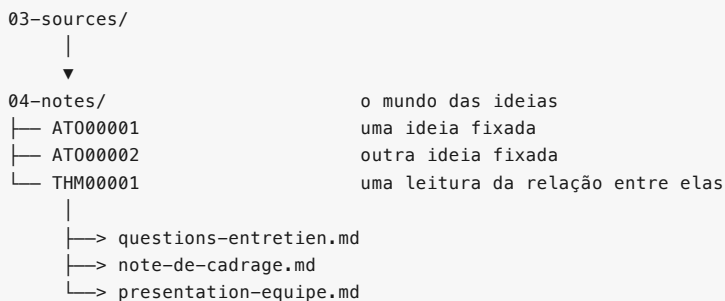
```

contexte-qualite-comptes-rendus/
├── 05-outputs/              o que se produz para um uso
│   ├── README.md           a lista dos resultados, gerado
│   ├── _draft/             os entregáveis em curso
│   ├── questions-entretien.md um guia para continuar
│   ├── note-de-cadrage.md   uma síntese para decidir
│   └── presentation-equipe.md os mesmos pontos, reformulados

```

## O mundo das ideias: tijolos para várias construções

A separação entre 04-notes/ e 05-outputs/ carrega uma das forças principais do Contexto. As notas e o glossário formam ali um *mundo das ideias*: um conjunto de tijolos de conhecimento, reutilizáveis sem limite, que já não são material bruto e ainda não são um resultado destinado a alguém.



5

No meu exemplo, o atrito do preenchimento e a necessidade de dados estruturados foram fixados antes que eu soubesse em que forma seriam apresentados.

- o guia de entrevista tira dali perguntas,
- a nota de enquadramento tira dali o que está em jogo,
- a apresentação tira dali uma narrativa destinada ao time.

Esses três entregáveis não possuem o conhecimento que usam.

- Eles selecionam e recompõem uma parte dele para uma necessidade, um público e um formato específicos.
- Um entregável não esgota, portanto, as notas de que é feito: os tijolos continuam disponíveis para outra construção.

Sem esse mundo intermediário, cada resultado novo me obrigaria a reabrir as fontes, a refazer as aproximações e a reformular as mesmas coisas.

**O Contexto não guarda só o que já foi produzido: ele constrói uma capacidade duradoura de produzir de novo.**

### **Nem todos os arquivos mudam da mesma maneira**

A árvore só fica de fato legível quando sei como cada um dos elementos dela muda.

Alguns arquivos são reescritos para mostrar a melhor situação disponível agora:

- `state.md`;
- `handoff.md`;
- os rascunhos de `05-outputs/_draft/`.

5

Outros apenas se alongam, para guardar o passado:

- as fontes de sessão;
- a história datada;
- a síntese;
- a série dos checkpoints.

Os documentos recebidos, esses, continuam congelados, e uma nota substituída por outra continua consultável. Os arquivos `README.md`, enfim, são gerados a partir do que existe: corrigi-los à mão criaria uma segunda versão da realidade, que acabaria divergindo.

Essa diversidade explica por que um único arquivo grande já não me bastava. Teria sido preciso pedir ao mesmo texto:

- que mostrasse o presente,
- que guardasse o passado,
- que recebesse material bruto,
- que fixasse ideias,
- que carregasse um resultado acabado.

### **Um esqueleto em vez de uma receita para copiar**

Esse esqueleto não é uma documentação colocada ao lado do

Contexto: é uma pasta de verdade, que eu copio.

- Cada arquivo chega ali com as seções que deve conter e algumas indicações, de modo que dois Contextos criados com meses de intervalo compartilham as mesmas referências.

Mesmo assim, a estrutura continua adaptável. Um Contexto pode receber:

- uma regra específica,
- um entregável próprio do ofício dele,
- uma fonte de um tipo inédito.

A base comum guarda sobretudo as três fronteiras de que o meu trabalho precisa:

- o estável e o presente,
- o bruto e o interpretado,
- o conhecimento e a forma dada a ele.

## Falar com o Contexto

Até aqui mostrei pastas e arquivos. Daria para concluir que preciso abrir cada uma delas, saber onde escrever e mover eu mesmo cada informação. Não é assim que trabalho.

Falo com a IA:

*Abra o Contexto dedicado à qualidade dos relatórios de serviço.*

*Acrescente este documento como fonte.*

*Transforme esta fonte em notas.*

*Salve o trabalho.*

Atrás de cada um desses pedidos existe um procedimento que escrevi uma vez e que a IA pode retomar a cada sessão.

- No Claude Code e no Codex, esse tipo de procedimento reutilizável tem um nome: um **skill**.

O skill de abertura indica, por exemplo:

- o que é preciso absorver antes de começar;
- que arquivos ler sistematicamente;
- o que deve ficar em disco até que a tarefa exija;
- como sinalizar um problema;
- que informação me devolver para que eu possa agir na hora.

Não preciso repassar essas etapas todas as vezes.

- Peço simplesmente que abra o Contexto.

A conversa carrega a minha intenção do momento.

- O skill carrega a maneira confiável de executar os gestos que voltam sempre.

Meu sistema tem assim alguns verbos estáveis:

- criar um Contexto;
- abri-lo e reencontrar o trabalho em curso;
- acrescentar uma fonte ou guardar uma sessão;
- transformar as fontes em notas;
- salvar o estado do trabalho;
- conferir a coerência dele;
- arquivá-lo quando ele atingiu o objetivo.

Nem todas as minhas trocas com a IA passam por um skill. Continuo a pensar, a questionar, a corrigir e a arbitrar em linguagem corrente.

**Os skills enquadram apenas as operações que exigem respeitar sempre as mesmas etapas.**

- Eles impedem, por exemplo, que uma fonte seja transformada sem guardar a origem dela, ou que uma sessão termine sem preparar a retomada.

O procedimento continua estável, mas o julgamento não se torna automático. Dois modelos podem formular uma nota de outro jeito. Ainda assim, devem consultar os mesmos elementos, respeitar as mesmas fronteiras e fazer as mesmas conferências.

Compartilhar os skills não dá a ninguém, portanto, o meu Contexto. Dá a ele a mesma maneira de criar o seu e de mantê-lo vivo.

Não direi mais nada sobre a mecânica deles.

Os skills não são nem uma invenção nem uma particularidade do Contexto. O Claude Code e o Codex os oferecem justamente para descrever procedimentos reutilizáveis, e a documentação deles explica melhor do que eu faria aqui como criá-los e usá-los.

O que pertence ao Contexto é a escolha das competências que se dão a ele: criar, abrir, alimentar, transformar, salvar, conferir e arquivar. É também a ordem em que elas se encadeiam e as fronteiras que cada uma deve respeitar.

Os arquivos guardam o que o Contexto sabe. Os skills organizam os gestos pelos quais ele muda.

O mapa está aberto. A visita pode agora começar como começa uma sessão de verdade: abrindo o Contexto.

5

---

#### PARA LEMBRAR — O ESQUELETO DE UM CONTEXTO

Um Contexto novo chega já mobiliado: seis zonas, cada uma respondendo a uma pergunta diferente. Os números impõem uma ordem de leitura, não uma hierarquia de valor. O que torna a árvore legível não é a forma dela, mas saber como cada arquivo muda. O Claude Code e o Codex fornecem o mecanismo dos skills; o Contexto define os gestos recorrentes, a ordem deles e os limites deles.

---

## Abrir: reencontrar a missão e o presente

---

Passaram-se duas semanas desde a minha última sessão sobre os relatórios de serviço — aqueles documentos em que um técnico anota o que constatou e o que fez depois de uma intervenção em um equipamento.

- As entrevistas e as notas continuam ali, no lugar delas.

Mas se eu as reabrir a esmo, vou passar meia hora reconstruindo o que já sabia, e vou reconstruir mal.

- A abertura começa, portanto, por um núcleo bem menor: o quadro, o estado e a retomada.

Não preciso pedir tudo isso. Chamo o skill de abertura, e ele desenrola o procedimento dele.

### Reencontro primeiro por que estou aqui

O primeiro arquivo que a IA lê (pelo skill de abertura de contexto) é `00-frame/mission.md`. O modelo dele faz cinco perguntas:

```
# Missão

## 0 objetivo
0 que este `Contexto` deve permitir?

## A pergunta central
Que pergunta guia a triagem e a exploração?

## 0 que entra no assunto
Que material pode entrar?

## 0 que está fora do assunto
0 que se recusa, mesmo que pareça interessante?

## 0 entregável esperado
0 que se quer produzir, se já se sabe?
```

No meu, o começo está preenchido assim:

```
## 0 objetivo
Entender o que degrada a qualidade dos relatórios de
serviço.

## A pergunta central
Como recolher uma informação aproveitável sem tornar o
preenchimento desestimulante?

## 0 que está fora do assunto
- Escolher uma interface antes de ter confrontado várias
fontes.
- Generalizar a partir de uma única entrevista.

## 0 entregável esperado
Preparar as decisões de produto e as próximas entrevistas.
```

Uma missão que dissesse apenas “melhorar os relatórios de serviço” não me orientaria muito. A pergunta central, essa, já torna alguns documentos mais pertinentes que outros, e o

fora do assunto impede que uma ideia sedutora transforme em silêncio a minha exploração num projeto de solução.

- O entregável, por sua vez, pode continuar desconhecido.
- Escrevo essa incerteza em vez de inventar uma saída prematura: o Contexto sabe então que ainda está explorando.

### **As regras que eu tinha estabelecido me esperam**

Duas missões próximas podem exigir cuidados diferentes.

`directives.md` guarda essas regras junto do trabalho:

- confidencialidade,
- nível de prova,
- vocabulário,
- validação ou forma esperada
- ...

O modelo é propositalmente simples: uma regra, uma data, uma formulação no imperativo.

```
# Diretrizes
```

- (2026-09-11) Anonimizar cada entrevista antes de integrá-la: substituir o nome da pessoa entrevistada pela função dela, mas guardar o nome da empresa e o país.
- (2026-09-11) Distinguir o que as fontes mostram do que nós deduzimos delas.
- (2026-09-11) Não propor nenhuma solução antes de ter documentado as práticas reais.

Uma diretriz só vale se puder guiar uma ação futura.

- “Ser prudente” me deixa toda a interpretação por refazer;

- “distinguir a observação da dedução” me dá uma conferência precisa na hora de escrever uma nota ou um entregável.

Mas uma diretriz não é apenas uma instrução que reencontro ao abrir um arquivo. É uma regra permanente que o Contexto deve aplicar sempre que uma ação entra no campo dela.

Acrescento, por exemplo, a transcrição de uma entrevista nova. Peço simplesmente ao Contexto que a integre (pelo skill dedicado a isso), sem pensar em lembrar que ela contém o nome da pessoa entrevistada. A diretriz de anonimização atua mesmo assim durante a inclusão: antes de registrar a transcrição como fonte, o skill substitui esse nome pela função da pessoa entrevistada, guardando ao mesmo tempo o nome da empresa e o país.

Não preciso lembrar da regra no momento exato em que ela se torna necessária. O Contexto lembra por mim.

As diretrizes não deixam, portanto, de agir depois que a abertura termina. Elas continuam aplicáveis durante toda a sessão e acompanham os diferentes gestos do Contexto:

- ao acrescentar uma fonte, a regra de anonimização protege a identidade da pessoa entrevistada;
- ao criar uma nota, a regra sobre o nível de prova separa o que a fonte mostra do que nós deduzimos dela;
- ao preparar um entregável, a última diretriz impede que se apresente cedo demais uma solução como coisa resolvida.

O skill que executa uma ação deve começar identificando as diretrizes que dizem respeito a ela e conferir depois que o resultado dele as respeita.

```
pedido
↓
diretrizes aplicáveis
↓
ação do skill
↓
conferência do resultado
```

As diretrizes cercam assim a ação: elas a limitam antes da execução e servem para conferir o resultado dela depois. Não agem sozinhas e não sabem tudo. São os skills que as tornam eficazes no momento certo.

A missão indica para onde ir. As diretrizes acompanham cada gesto do Contexto, para que eu não tenha de lembrar, a cada etapa, dos cuidados já decididos.

### **Nem tudo merece ser reaberto**

Minha pasta contém dezenas de fontes e de notas. Carregar todas a cada abertura afogaria as prioridades no monte.

loading.md escolhe, portanto, um núcleo estável:

```
## Sempre

- `00-frame/mission.md`
- `00-frame/directives.md`
- `01-state/state.md`
- `01-state/todo.md`
- `01-state/handoff.md`

## Sob demanda

- as fontes necessárias a uma conferência
- as notas ligadas ao tema em que se trabalha
- o rascunho ativo
```

Tudo continua acessível, mas nem tudo é carregado no contexto da IA. `loading.md` designa os elementos indispensáveis para entender a missão, interpretar um pedido e agir sem perder as regras já estabelecidas.

- O resto fica em disco e só é consultado em caso de necessidade.
- Uma fonte antiga pode ser indispensável para conferir uma afirmação sem ocupar lugar no contexto ativo durante todas as sessões.

O conteúdo de `loading.md` depende do Contexto.

- Um livro vai carregar o roteiro dele a cada abertura; uma exploração de produto, a nota temática central dela.
- O princípio continua o mesmo:
  - começar com informação suficiente para agir,
  - e ir buscar o resto quando a tarefa justificar.

## Em que ponto eu estava de verdade

A missão continua estável; 01-state/state.md, esse, é feito para mudar. Ele cabe em três blocos:

```
# Estado

## 0 que se sabe
- O preenchimento adiado leva a uma perda de detalhes na primeira entrevista.
- Alguns campos obrigatórios são contornados na segunda.

## 0 que continua incerto
- Não sabemos que campos são de fato reutilizados depois do preenchimento.

## As dívidas
- O white paper foi acrescentado, mas o grau de confiabilidade dele ainda está por qualificar.
```

O estado não conta a história toda nem resume todas as notas: ele expõe a melhor compreensão que tenho do trabalho em si.

state.md se dirige ao mesmo tempo à IA e ao humano.

Para a IA, ele constitui uma memória de trabalho permanente.

- Carregado a cada abertura, indica a ela os pontos firmes a que pode recorrer, as incertezas que deve preservar e as dívidas que não deve esquecer.
- Sem ele, ela teria de reconstruir esse estado a partir das fontes e das notas a cada nova sessão.

Para mim, ele faz o papel de um painel de controle.

- Em poucas linhas posso conferir o que o Contexto toma por estabelecido, o que ainda considera incerto e que trabalhos continuam incompletos.
- Posso então corrigir esse estado se ele não corresponder à minha compreensão.

Algumas dezenas de linhas bastam assim para a IA retomar o trabalho sem confundir o que está estabelecido com as hipóteses, e para eu conferir a compreensão em que ela se apoia.

todo.md extrai dele as ações:

```
## Em curso
- T-001 – Preparar uma entrevista sobre o uso real dos
  campos obrigatórios.

## A fazer
- T-002 – Conferir que dados servem às auditorias.

## Feito
- T-003 – Comparar as duas primeiras entrevistas. –
  concluída em 2026-09-10
```

Não abro necessariamente todo.md para modificá-lo eu mesmo. Posso simplesmente dizer:

*Acrescente uma tarefa para conferir que dados servem às auditorias.*

O Contexto cria uma linha, atribui um identificador a ela e a coloca na seção certa.

Uma tarefa também pode aparecer durante o trabalho.

- Se a análise de uma fonte faz surgir uma conferência necessária, a IA me propõe acrescentá-la em vez de deixá-la sumir na conversa.
- Ela não transforma em silêncio cada ideia numa obrigação nova: quando uma tarefa muda as prioridades ou acarreta trabalho a mais, a arbitragem continua sendo minha.

Quando o trabalho está feito, posso pedir:

*Encerre a tarefa T-002 (ou uma descrição aproximada da tarefa, e a IA vai reencontrar a referência certa).*

O Contexto não apaga a linha. Ele a move para Feito, acrescenta a data de encerramento e, quando é útil, indica o resultado ou o elemento que permite considerá-la concluída.

- Uma tarefa concluída não desaparece.
- A presença dela me evita refazer o mesmo trabalho e me explica por que uma pista deixou de estar ativa.

todo.md não é, portanto, uma lista escrita de uma vez por todas. É a fila das ações do Contexto: a IA a consulta para saber o que falta fazer, a atualiza quando o trabalho avança e guarda as tarefas concluídas para não recomeçá-las.

### Por onde recomeço

handoff.md é o último arquivo do núcleo. Ele é reescrito a cada salvamento para responder a uma situação bem concreta: alguém chega agora, uma pessoa ou uma IA, sem nada da conversa anterior. O que essa pessoa precisa saber para recomeçar sem errar?

```
# Retomada
```

```
## Em que ponto estamos
```

```
Três fontes fizeram aparecer uma tensão entre facilidade de  
preenchimento e aproveitamento dos dados.
```

```
## O que é preciso saber antes de continuar
```

```
As duas entrevistas descrevem situações particulares,  
enquanto o white paper defende um ponto de vista geral.  
Esse conjunto faz aparecer uma tensão, mas ainda não  
permite medir a amplitude dela nem concluir que ela  
existe em todas as organizações.
```

```
## A próxima ação
```

```
Preparar as perguntas que permitirão identificar os campos  
realmente usados.
```

```
## O que ficou em aberto
```

```
Quanta estrutura as auditorias precisam continua  
desconhecido.
```

5

A próxima ação é única e imediatamente executável. Uma lista de dez possibilidades me devolveria o trabalho de definir prioridades que este arquivo existe justamente para me poupar.

O comando de abertura absorve, portanto, o que foi depositado durante o sono do Contexto, lê esse núcleo e depois me anuncia aquela ação.

- Ele reencontra a finalidade antes de reencontrar os detalhes, e minhas duas semanas de ausência já custam apenas um minuto de leitura.
- A sessão pode agora receber material novo sem perder a direção.

---

**PARA LEMBRAR — O NÚCLEO DE ABERTURA**

Reabrir um Contexto não consiste em reler o conteúdo dele, mas em carregar um núcleo pequeno: a missão, as diretrizes, o estado e a retomada. A missão diz para onde ir; as diretrizes determinam as condições do percurso. Uma retomada útil aponta uma só ação, imediatamente executável.

---

## Alimentar: fazer entrar o material

---

A sessão começa, e três coisas chegam ao mesmo tempo.

- Um responsável de operações me manda o registro de uma entrevista,
- uma ideia me ocorre durante a troca,
- uma nota que eu tinha deixado na véspera espera numa caixa de entrada (gerenciador de notas, e-mail...).

Esses três materiais não entram no Contexto da mesma maneira, e cada um tem o seu skill.

### Um documento chega como objeto acabado

Um documento externo é guardado com o arquivo de origem dele, o texto aproveitável dele e uma ficha que permite acompanhá-lo.

```
03-sources/documents/  
├─ SRC00001-entretien-exploitation.md  
├─ raw/  
│   └─ SRC00001-entretien-exploitation.pdf  
└─ fulltext/  
    └─ SRC00001-entretien-exploitation.md
```

A ficha SRC00001-entretien-exploitation.md pode tomar esta forma:

```
---
added: 2026-09-11
kind: pdf
intention: entrevista
raw: raw/SRC00001-entretien-exploitation.pdf
fulltext: fulltext/SRC00001-entretien-exploitation.md
extraction: ok
reliability: medium
---
```

# Entrevista com a responsável de operações

## Por que ela está aqui

Entender o que impede os técnicos de produzir relatórios de serviço aproveitáveis.

## Interpretação – não analisada

A responsável descreve uma oposição entre a rapidez do preenchimento em campo e a necessidade de dados estruturados para a análise.

## Cobertura

- Preenchimento no celular
- Campos obrigatórios
- Reutilização dos dados

## O que ela rendeu

- A completar depois da análise

Essa ficha não resume necessariamente o documento. Ela responde antes de mais nada a perguntas de gestão:

- de onde ele vem,
- por que eu o acrescentei,
- consigo ler o conteúdo dele,
- que confiança dar a ele?

A seção Interpretação – não analisada põe em quarentena uma primeira leitura. Ela evita que uma impressão formulada na hora da importação seja logo tratada como conhecimento estabelecido.

Quando uma fonte chega em forma de PDF, de página web ou de outro formato difícil de aproveitar direto, a extração produz uma versão em texto legível pelo humano e pela IA.

- Essa operação ainda não produz nenhum conhecimento: ela apenas torna o material acessível sem modificar o documento original.

A transformação vem depois.

- O texto extraído é analisado para fazer aparecer notas atômicas, notas temáticas, termos de glossário ou arbitragens.
- Essas notas alimentam o *mundo das ideias*, onde o material das fontes vira um conjunto de tijolos de conhecimento reutilizáveis.

O nível de confiabilidade não julga as fontes acrescentadas.

- Ele indica simplesmente como usar esse material: depoimento isolado, dado verificado, documento contratual ou hipótese a confirmar.

Nem todo o material útil já existe num documento. Uma parte está na minha experiência, nas minhas lembranças, nas minhas observações ou nas distinções que formulo enquanto trabalho.

Quando as explico durante uma sessão de troca com a IA, elas também entram no Contexto.

- Não viram automaticamente fatos estabelecidos
- Guardam a origem delas e poderão ser confrontadas com outras fontes.

Acrescentar material consiste, portanto, tanto em integrar um documento quanto em guardar o que a pessoa dona do Contexto traz para a conversa.

### O conhecimento da pessoa entra pela sessão

Uma sessão não é um documento entregue de uma vez por todas. Ela se constrói ao longo da troca. É registrada em `03-sources/sessions/SESSION-2026-09-12.md`, que pode tomar esta forma:

```
# Sessão de 12 de setembro de 2026
```

```
## 09:41 – IA
```

```
Os campos obrigatórios melhoram de fato a qualidade, ou só deslocam o esforço para o técnico?
```

```
## 09:42 – Autor
```

```
Nas entrevistas, os campos obrigatórios garantem sobretudo que um valor foi preenchido.
```

```
Não garantem nem a precisão dele nem a utilidade dele.
```

```
Quando o técnico não entende para que um campo servirá depois, ele o preenche o mais rápido possível para seguir em frente.
```

```
A obrigação melhora, então, a completude aparente, mas pode deslocar o esforço sem melhorar a informação realmente aproveitável.
```

```
...
```

Nem tudo o que a IA diz precisa ser guardado. Em compensação, uma pergunta, uma objeção ou uma reformulação que muda o trabalho merece ser.

- Minha fala e a da IA continuam separadas.
- Assim consigo reencontrar a origem de uma ideia e distinguir a minha intenção de uma proposta do sistema.

Uma sessão pode ser desordenada.

- O papel dela é capturar o movimento real do pensamento, não apresentar uma síntese perfeita.

### **A caixa de correio fala com o Contexto quando ele está fechado**

Uma ideia me ocorre fora de sessão, no trem ou no meio de uma reunião e sem acesso à internet.

- Ela entra então na caixa de correio.
- Essa caixa de correio é o arquivo `offline.md`, colocado na raiz do Contexto.
- Basta eu escrever nesse arquivo Markdown.

Ele pode tomar esta forma:

Perguntar se os técnicos contornam os campos obrigatórios com valores genéricos.

Na próxima abertura do contexto, essa nota é relida, aproximada do trabalho em curso e depois absorvida no lugar certo. Ela pode virar uma tarefa, uma pergunta de entrevista, uma nota, ou sumir se já não for pertinente.

### Acrescentar ainda não é entender

Uma vez entrado esse material, minha árvore contém várias fontes e o rastro da sessão em curso.

```
03-sources/  
├── documents/  
│   ├── SRC00001-entretien-exploitation.md  
│   ├── SRC00002-entretien-exploitation.md  
│   └── SRC00003-livre-blanc-donnees-intervention.md  
└── sessions/  
    └── SESSION-2026-09-11.md
```

Esses arquivos provam que o material está disponível. Não provam que ele foi entendido.

O arquivo `03-sources/ingestion-log.md` distingue as duas operações. Para cada fonte, ele indica se a análise está completa, parcial ou a rever, com uma impressão digital do conteúdo e algumas notas de tratamento.

Essa distinção se torna essencial quando um documento é substituído, quando uma extração muda ou quando uma sessão é interrompida. O Contexto pode então retomar a análise sem fingir que sabe tudo.

O material entrou. Falta transformá-lo em ideias bastante precisas para serem ligadas, discutidas e reutilizadas.

**PARA LEMBRAR — A FONTE**

Um documento, uma fala de sessão e uma nota fora de sessão entram por três portas diferentes e cada uma guarda a forma de origem dela. Uma fonte guarda o material e a prova, nunca a interpretação que se tirará deles. Acrescentar não é entender: o registro de ingestão distingue o que está presente do que está assimilado.

---

## Transformar: passar das fontes às ideias

---

Tenho agora três fontes diante de mim.

- Duas entrevistas descrevem o atrito provocado pelos formulários,
- Um white paper defende a necessidade de dados estruturados.

A tentação seria decidir na hora e escrever que os campos obrigatórios prejudicam a qualidade. Seria ir rápido demais: o Contexto deve preservar essas contribuições sem fundi-las numa conclusão única.

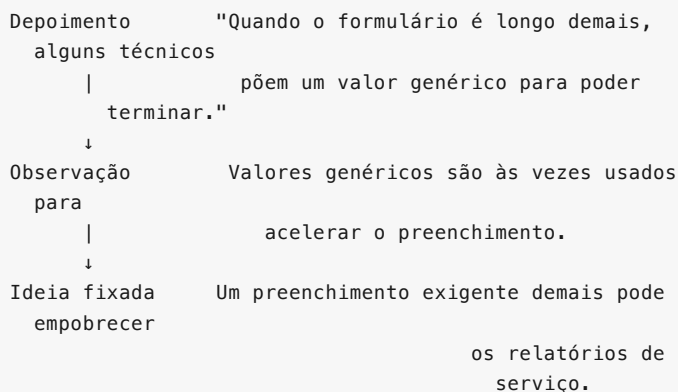
Chamo, portanto, o skill que cria e atualiza as notas: ele lê o que ainda não foi tratado e tira ideias disso.

### **Extraio uma ideia que se sustenta sozinha**

Uma fonte raramente contém uma ideia já pronta para ser reutilizada.

- Uma entrevista mistura fatos, exemplos, opiniões, hesitações e detalhes próprios da situação contada.
- Transformá-la não consiste em resumir tudo, mas em extrair dela as ideias.

O caminho cabe em três etapas:



Uma ideia forte não é uma fórmula espetacular nem uma verdade definitiva.

- Ela tem um centro de gravidade claro e firme,
- Pode ser entendida fora do texto que a fez nascer
- Continua delimitada o bastante para ser confirmada, matizada ou contrariada.
- Ela guarda também a origem e os limites dela, para não transformar um depoimento específico em regra geral.

A IA lê a fonte à luz da missão e das diretrizes e depois a confronta com as notas já presentes. Cria uma nota nova quando a ideia é nova; senão, enriquece a existente com a ressalva ou o limite a mais.

Ela faz esse trabalho sem me pedir para validar cada formulação. Posso depois reler e corrigir as notas.

- Quando aparece uma ambiguidade durante a sessão, ela me pergunta;

- Se mais tarde descobre uma contradição que não consegue resolver, ela a guarda em vez de decidir no meu lugar.

```
04-notes/  
└─ atomic/  
    └─ AT000001-la-contrainte-de-saisie-peut-appauvrir-les-comptes-rendus.md
```

---

id: AT000001

status: active

sources: [SRC00001, SRC00002]

---

# Um preenchimento exigente demais pode empobrecer os relatórios de serviço

## Ideia principal

Quando o formulário atrasa a intervenção, os técnicos podem inserir valores genéricos ou reduzir a descrição deles ao mínimo esperado.

## Por que isso importa

Um dado formalmente estruturado pode parecer limpo e ainda assim descrever pior a realidade de campo.

## Ressalvas e limites

As duas entrevistas não permitem medir a frequência do fenômeno.

## Origem / prova

- SRC00001: exemplos de valores genéricos usados para terminar mais rápido.
- SRC00002: queda de precisão observada depois da inclusão de campos obrigatórios.

## Ligações

- Contraste com AT000002.

O título carrega uma afirmação, não apenas um tema. A nota continua compreensível sem que eu tenha de reabrir os documentos de que ela vem, guardando ao mesmo tempo as ligações que me permitem conferir a origem dela.

Uma nota atômica não fica congelada no primeiro jato.

- Ela pode ganhar uma ressalva, uma prova nova ou uma relação sem perder a identidade dela.

O white paper rende uma segunda nota: os dados estruturados são necessários para comparar, agregar e automatizar.

- Essa ideia continua separada da primeira porque afirma outra coisa.

### **Me recuso a decidir cedo demais**

As duas ideias parecem se opor, e ainda não tenho material suficiente para arbitrá-las. Abro, portanto, uma contradição.

## C-001 – Estruturar sem degradar o preenchimento

- De um lado: os campos impostos podem empobrecer as descrições de campo.
- Do outro: o aproveitamento em escala exige dados comparáveis.
- O que está em jogo: determinar que informações devem ser estruturadas no momento do preenchimento e quais podem ser depois.
- Aberta em: 2026-09-11

5

A contradição me evita dois erros simétricos: escolher um lado cedo demais, ou suavizar o desacordo numa formulação vaga que não incomoda ninguém.

- Ela continua aberta até que uma decisão, uma fonte nova ou uma reformulação a resolva.

Nem toda incerteza merece esse tratamento.

- Reservo isso às tensões que pesam de fato sobre a continuação do raciocínio ou sobre a qualidade de uma decisão.

### **Ligo o que essas ideias dizem juntas**

Durante a ingestão, a IA não cria apenas notas atômicas.

- Ela examina também as relações entre as ideias extraídas e as notas já presentes.
- Quando várias delas formam um conjunto que não se reduz a uma só afirmação, ela cria na mesma operação uma nota temática e põe as ligações que as unem.

Não preciso lançar um segundo comando.

- A criação das notas atômicas, a aproximação delas e o eventual surgimento de uma nota temática fazem parte de uma mesma ingestão.

```
---  
id: THM00001  
status: active  
sources: [AT000001, AT000002]  
---
```

# A qualidade depende do preenchimento e do aproveitamento

## Ângulo

A qualidade de um relatório de serviço não pode ser avaliada só pelo número de campos preenchidos.

## Síntese

O sistema deve obter estrutura suficiente para aproveitar os dados sem tornar o preenchimento artificial ou custoso.

## Notas ligadas

- AT000001 – A exigência do preenchimento pode empobrecer os relatórios de serviço.
- AT000002 – Dados estruturados são necessários ao aproveitamento.

## Tensões

- C-001 – Estruturar sem degradar o preenchimento.

A nota temática não justapõe apenas resumos. Ela propõe uma leitura das relações entre elas e mostra o que continua em aberto.

### **Percebo que uma palavra não tem o mesmo sentido para todo mundo**

Parte do desacordo vem às vezes do vocabulário. O glossário fixa então o sentido adotado neste Contexto.

## Dado estruturado

Informação registrada segundo uma forma explícita que permite compará-la ou tratá-la automaticamente.

Não confundir com: campo que o usuário é obrigado a preencher no momento da intervenção.

Exemplos: tipo de falha, tempo de parada, equipamento envolvido.

Essa definição local não pretende se tornar universal.

- Ela simplesmente impede que duas notas, ou duas pessoas, usem a mesma palavra para falar de coisas diferentes.

### Tijolos neutros diante do entregável

Uma nota pertence ao *mundo das ideias* enquanto procura fixar um conhecimento em vez de produzir um efeito sobre um público.

- Ela não deve, portanto, virar cedo demais o gancho de um artigo, um slide, um argumento comercial ou um pedaço de especificação.

Essa neutralidade não significa que a nota continue vaga.

- Ao contrário: o título dela afirma algo preciso, o conteúdo expõe as ressalvas e as ligações permitem conferir os alicerces.
- O que continua neutro é a forma dela: não está organizada para servir a um único resultado.

AT000001 poderá assim virar uma pergunta sobre como os campos são contornados no guia de entrevista, um risco na

nota de enquadramento ou um exemplo na apresentação ao time. AT000002 poderá sustentar uma pergunta sobre os usos do dado, justificar uma exigência de produto ou explicar ao time por que o texto livre nem sempre basta.

A nota temática não funde esses tijolos num texto definitivo.

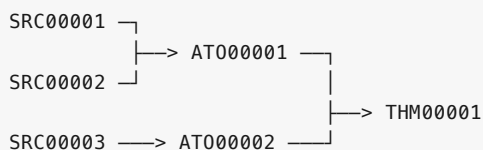
- Ela indica o que eles permitem entender juntos e guarda a tensão que precisará ser tratada de outro jeito conforme o entregável.

Essa camada intermediária evita recomeçar a análise a cada produção.

- As fontes continuam sendo as provas,
- as notas viram conhecimentos a que se pode recorrer,
- cada entregável compõe com elas conforme a própria necessidade.

### Posso subir de cada ideia até a prova dela

Nesse ponto, as relações se representam assim:



As ligações nomeadas me permitem distinguir uma fonte que confirma uma ideia de outra que a matiza ou a contraria.

- Um nível de confiança torna visível o que se apoia em várias observações e o que continua hipotético.

A cobertura de ingestão mostra que fontes contribuíram de fato para as notas. Ela me impede de considerar assimilado um documento simplesmente presente na pasta.

Essas notas não são escritas para um entregável específico.

- Elas formam um conhecimento comum que poderá alimentar vários resultados sem ser copiado nem trancado em um deles.

**\*\*O *mundo das ideias* é justamente esse capital reutilizável: um conjunto de tijolos firmes o bastante para serem reencontrados e independentes o bastante para serem recombina****dos.\*\***

---

#### **PARA LEMBRAR — A NOTA ATÔMICA**

Uma ideia fixada carrega uma afirmação e não um tema, e se entende sem reabrir o documento de que ela vem. Uma contradição é guardada enquanto pesa sobre o raciocínio, em vez de ser decidida cedo demais ou suavizada. Cada nota guarda a ligação com as fontes dela, o que permite subir da ideia até a prova.

---

5.5

# Produzir e decidir

Minha nota temática fez aparecer uma pergunta precisa: que informações precisam ser preenchidas no fim da intervenção, e quais podem ser completadas depois?

Três pessoas esperam agora alguma coisa de mim.

- Um técnico que preciso entrevistar,
- um comitê que precisa decidir,
- um time a quem preciso explicar tudo isso.

Não vou conduzir três investigações: vou recorrer três vezes às mesmas notas.

É o único momento da sessão em que nenhum skill me ajuda: produzir um entregável é um trabalho de escrita, não um gesto que se repete.

## Recorro três vezes ao mesmo estoque

A passagem das notas aos entregáveis não é uma cópia. Cada resultado seleciona os tijolos que lhe são úteis, os ordena e os reformula em função do destinatário dele.

| Tijolo comum                            | Guia de entrevista       | Nota de enquadramento                   | Apresentação ao time                  |
|-----------------------------------------|--------------------------|-----------------------------------------|---------------------------------------|
| AT000001 —<br>Um preenchimento exigente | Perguntar que campos são | Descrever o risco de perda de qualidade | Mostrar um exemplo de atrito em campo |

| <b>Tijolo comum</b>                                                 | <b>Guia de entrevista</b>                           | <b>Nota de enquadramento</b>                             | <b>Apresentação ao time</b>                         |
|---------------------------------------------------------------------|-----------------------------------------------------|----------------------------------------------------------|-----------------------------------------------------|
| demais pode empobrecer os relatórios de serviço                     | contornados e como                                  |                                                          |                                                     |
| AT000002 — Dados estruturados são necessários ao aproveitamento     | Perguntar que dados são de fato reutilizados        | Estabelecer a necessidade de comparação e de automação   | Explicar o valor esperado para o time               |
| THM00001 — A qualidade depende do preenchimento e do aproveitamento | Organizar a entrevista em torno das duas exigências | Formular o que está em jogo e os critérios de arbitragem | Construir o fio da narrativa                        |
| C-001 — Estruturar sem degradar o preenchimento                     | Transformar a tensão em perguntas a conferir        | Guardá-la como problema a resolver                       | Torná-la visível sem anunciar uma solução prematura |

O mesmo estoque rende assim três montagens, sem que nenhum entregável retome tudo nem substitua as notas que o alimentam.

Essa separação me permite sobretudo corrigir o conhecimento uma única vez.

- Se uma fonte nova matizar AT000001, meus próximos entregáveis vão partir desse tijolo enriquecido, em vez de reproduzir uma formulação antiga espalhada por vários documentos.

### **Começo pelo mais urgente: a próxima entrevista**

Minha primeira necessidade é preparar a entrevista com um técnico. O entregável começa no espaço dos rascunhos.

```
05-outputs/  
└─ _draft/  
   └─ questions-prochain-entretien.md
```

5

# Perguntas para a próxima entrevista

1. Que campos mais atrasam você durante uma intervenção?
2. O que você preenche quando a informação pedida não está disponível?
3. Que informações você poderia acrescentar depois da intervenção?
4. Que dados precisam estar exatos na hora para servirem ao aproveitamento?

As notas procuram guardar uma ideia reutilizável. O entregável, esse, escolhe um destinatário, uma forma e uma ação esperada.

Uma vez relido e validado, o arquivo sai de `_draft/` e vai direto para `05-outputs/`. O deslocamento indica um nível de maturidade, não uma diferença de gênero documental.

Até aqui eu só produzi: dei forma ao que já sabia, para três públicos. O restante deste capítulo é um gesto bem diferente. Em algum momento é preciso parar de restituir e decidir — e o que eu decido precisa poder ser reencontrado daqui a seis meses.

### **O decision record guarda a escolha que molda o resultado**

“Entrevistar um técnico antes de redigir uma especificação” não é uma arbitragem sobre o que será entregue.

- Se essa entrevista é simplesmente a próxima ação, ela pertence a `01-state/todo.md`.
- Se todos os trabalhos de descoberta devem respeitar essa regra, ela vira uma diretriz em `00-frame/directives.md`.

Em nenhum dos dois casos ela justifica um decision record. O decision record aparece quando o trabalho faz surgir várias direções possíveis e é preciso escolher a que vai estruturar o produto, o texto, o método ou outro resultado do Contexto. Ele guarda então o que se decidiu conceber ou entregar, as alternativas descartadas, as razões da escolha e as consequências aceitas.

No meu exemplo, a arbitragem não trata, portanto, da organização das entrevistas. Ela trata de uma decisão funcional:

- exigir todas as informações no fim da intervenção,

- guardar apenas texto livre,
- ou permitir que o relatório seja completado num segundo momento.

Um decision record útil não delega o cerne da escolha à pessoa que vai executá-la. Três pessoas que o leem devem entender o mesmo perímetro, os mesmos comportamentos esperados e os mesmos limites. Os detalhes técnicos que não estruturam nada podem ficar em aberto, mas nenhum deles deve poder mudar em silêncio a decisão.

```
04-notes/  
└─ decision-records/  
    └─ DRE00001-limiter-la-cloture-a-quatre-informations.md
```

Eis o que isso dá depois de escrito. O modelo é longo porque guarda todo o raciocínio que torna a decisão compreensível e revisável — quem já conhece os decision records pode pular direto para o comentário que vem depois.

5

```
---  
uuid: 0d5efbe2-1eac-4ad7-94c1-29f68adcf001  
id: DRE00001  
created: 2026-09-11  
updated: 2026-09-11  
status: accepted  
decided_by: time de produto  
decided_on: 2026-09-11  
supersedes: null  
superseded_by: null  
sources: [AT000001, AT000002, THM00001]  
---  
  
# Limitar a conclusão de uma intervenção a quatro
```

informações obrigatórias

## ## Decisão

Para que uma intervenção seja considerada concluída, exatamente quatro informações devem estar presentes:

1. O equipamento envolvido, já indicado no registro da intervenção ou preenchido no momento da conclusão.
2. A data e a hora reais do fim da intervenção.
3. O estado do equipamento no fim da intervenção: recolocado em serviço, mantido em funcionamento reduzido ou deixado fora de serviço.
4. Um resumo não vazio dos trabalhos feitos ou da razão pela qual o equipamento não pôde ser recolocado em serviço.

Nenhuma outra informação condiciona a conclusão da intervenção.

Depois da conclusão, o relatório pode ser completado com a natureza da falha, a causa identificada dela, a natureza dos trabalhos feitos, as peças trocadas e o tempo de indisponibilidade do equipamento, sem pôr em questão a conclusão da intervenção.

O técnico pode preencher essas informações complementares, mas o responsável de operações deve completá-las ou validá-las em vinte e quatro horas.

Um relatório ainda incompleto continua consultável.

Ele é sinalizado como pendente de conclusão, e depois o responsável de operações é avisado após vinte e quatro horas.

## ## Contexto

Os técnicos precisam poder encerrar o trabalho deles em campo sem percorrer um formulário de análise completo.

O setor de operações precisa, mesmo assim, de informações comparáveis para analisar as falhas, as causas delas, os trabalhos feitos, as peças trocadas e os tempos de indisponibilidade.

## ## Hipóteses

- O registro da intervenção identifica normalmente o equipamento antes do início dos trabalhos.
- O técnico conhece o estado do equipamento e consegue descrever brevemente os trabalhos feitos no momento da conclusão.
- O responsável de operações dispõe do tempo e das informações necessárias para validar o relatório completo em vinte e quatro horas.

## ## Alternativas

- Exigir as nove informações no momento da conclusão.
- Guardar apenas um resumo livre e extrair automaticamente dele as informações sobre as intervenções depois.
- Tornar todas as informações facultativas e conferir a qualidade só depois.

## ## Razões

As quatro informações imediatas bastam para identificar a intervenção, datar o fim dela, conhecer o estado do equipamento e entender os trabalhos feitos.

As cinco informações de análise continuam necessárias, mas não devem impedir o técnico de terminar o trabalho dele em campo.

O segundo momento atribui explicitamente ao responsável de operações a responsabilidade de completar ou validar o relatório.

## ## Base

- AT000001 descreve as formas de contornar ligadas a um preenchimento exigente demais.
- AT000002 estabelece a necessidade de dados comparáveis.
- THM00001 liga as duas exigências.

## ## Consequências

### ### Positivas

- A conclusão pede apenas quatro informações conhecidas na hora.
- As informações necessárias às análises das intervenções continuam disponíveis depois que o relatório foi completado.
- A responsabilidade pelos dados que faltam fica visível.

### ### Contrapartidas

- Os indicadores das intervenções podem ficar incompletos durante as vinte e quatro horas seguintes a uma intervenção.
- O responsável de operações recebe uma carga de validação a mais.

### ### Riscos

- Relatórios incompletos podem se acumular.
- Informações acrescentadas depois podem refletir com menos fidelidade a situação constatada em campo.

## ## Critérios de conformidade

- Uma intervenção não pode ser concluída se faltar uma das quatro informações imediatas.
- Uma intervenção pode ser concluída sem nenhuma das cinco informações complementares.
- As informações complementares podem ser modificadas sem mudar o estado de conclusão da intervenção.
- O relatório é apresentado como "pendente de conclusão" ou "completo e validado" em todos os usos que se fazem dele.
- O responsável de operações é avisado vinte e quatro horas depois do encerramento se o relatório não estiver completo e validado.

## ## Decisões ligadas

- Resolve C-001 – Estruturar sem degradar o preenchimento.

## ## Fontes

- SRC00001
- SRC00002
- SRC00003

O status `accepted` indica aqui que a arbitragem está validada e é precisa o bastante para guiar a implementação.

- Enquanto as informações esperadas, as responsabilidades ou os critérios de conformidade continuarem indeterminados, o decision record deve manter o status `proposed`.
- Uma vez aceito, ele é congelado como rastro histórico e não muda mais em silêncio.

Se uma decisão nova o substituir, o decision record antigo continua consultável e os dois arquivos ficam ligados por `supersedes` e `superseded_by`.

Essa continuidade guarda a cronologia completa das arbitragens: o que tinha sido decidido, com base em que informações, e depois que elemento levou a escolher de outro jeito.

Seis meses depois, portanto, não reencontro só a regra atual. Consigo entender por que uma primeira direção tinha parecido certa, o que a tornou insuficiente e como a decisão seguinte se construiu.

Sem essa cronologia, uma sucessão de escolhas acaba parecendo uma incoerência. Com ela, volta a ser a história compreensível de um produto que evolui.

- As alternativas mostram o que foi descartado.

- As contrapartidas tornam visível o que foi sacrificado.
- A base liga a decisão aos conhecimentos e às provas que a sustentam.

### **Produzir não esgota o que eu sei**

O guia de entrevista, a nota temática e o decision record tratam do mesmo problema sem se repetir, porque cumprem três funções diferentes.

- A nota guarda o que o Contexto acredita saber.
- A decisão guarda uma escolha e as razões dela
- O guia transforma esse conhecimento em ação junto a um destinatário preciso.

Outro entregável poderá retomar o mesmo conhecimento comum sob uma forma diferente. **Produzir não esvazia, portanto, o Contexto da substância dele: revela o que essa substância permite fazer.**

Falta registrar o estado alcançado, para que a próxima sessão retome sem ter de reconstruir o caminho percorrido.

5

---

### **PARA LEMBRAR — O DECISION RECORD**

As mesmas notas alimentam vários entregáveis, e nenhum entregável consome o conhecimento que usa. Um decision record guarda uma arbitragem que molda o que será concebido ou entregue, não uma ação a executar. Ele guarda o que foi descartado e o que foi sacrificado, para que a escolha continue revisável.

---

## Salvar e retomar

---

O decision record acaba de ser aceito e uma reunião começa em três minutos. Não terminei nem o estudo, nem sequer uma etapa que se parecesse com um fim.

É o caso comum: outra tarefa reclama a minha atenção, a máquina precisa reiniciar, eu troco de computador, ou a janela de contexto da IA fica carregada demais.

- Salvar não responde, portanto, a uma necessidade do próprio trabalho, do tipo “o estudo está terminado”.
- É uma garantia técnica: o estado do `Contexto` deve sobreviver à sessão em que foi produzido.

Preciso poder me interromper agora, reabrir depois e reencontrar o trabalho exatamente onde o deixei.

Um skill cuida disso, e ele precisa levar poucos segundos: um salvamento que sai caro não será acionado no momento em que se está com pressa, isto é, no momento em que ele mais serve.

### A IA congela o que se sabe neste instante

O arquivo `01-state/state.md` resume o que está estável o bastante para orientar a continuação.

```
contexte-qualite-comptes-rendus/  
└─ 01-state/  
   └─ state.md
```

No instante do salvamento, ele pode tomar esta forma:

```
# Estado atual
```

```
## Conhecimento fixado
```

- Um preenchimento exigente demais pode empobrecer os relatórios de serviço.
- O aproveitamento em escala exige, ainda assim, dados estruturados.

```
## Decisão tomada
```

- O encerramento exige o equipamento envolvido, a data e a hora de fim, o estado do equipamento e um resumo dos trabalhos feitos.
- A natureza e a causa da falha, a natureza dos trabalhos, as peças trocadas e o tempo de indisponibilidade são completados depois do encerramento e validados em vinte e quatro horas pelo responsável de operações.

```
## Incerteza principal
```

- Medir se essa separação reduz de fato o tempo de encerramento sem degradar a qualidade dos dados.

```
## Dívida de conhecimento
```

- Nenhum piloto mediu ainda o tempo de encerramento nem a proporção de relatórios completos depois de vinte e quatro horas.

- Uma tarefa concluída pode sumir da lista de tarefas.
- Uma contradição não resolvida continua visível no estado.

A síntese contém apenas o que o trabalho fixou o suficiente. O estado é reescrito para descrever o presente, enquanto a síntese se enriquece aos poucos para guardar a compreensão adquirida.

## Preparo uma só ação para o meu eu futuro

O arquivo de retomada não conta a história toda: ele prepara o próximo gesto útil.

```
contexte-qualite-comptes-rendus/  
└─ 01-state/  
   └─ handoff.md
```

O arquivo 01-state/handoff.md pode tomar esta forma:

```
# Retomada  
  
## Estado atual  
  
DRE00001 está aceito e C-001 está resolvida.  
As quatro informações de encerramento e as cinco  
informações complementares estão fixadas.  
  
## Ponto de atenção  
  
Não acrescentar uma informação obrigatória ao encerramento  
sem abrir um decision record novo que substitua DRE00001.  
  
## Próxima ação  
  
Iniciar a execução de DRE00001.  
  
## Em aberto  
  
- Definir os rótulos apresentados aos técnicos para os três  
  estados possíveis do equipamento depois da intervenção.  
- Preparar a medição do tempo de encerramento e da  
  proporção de relatórios completos depois de vinte e  
  quatro horas.
```

Uma retomada útil aponta uma só próxima ação. Uma lista de dez possibilidades obrigaria a sessão seguinte a refazer o trabalho de decisão que acabo justamente de fazer.

## Guardo um rastro para comparar minhas retomadas

O arquivo `01-state/checkpoints.md` guarda uma série de fotografias compactas da situação.

```
contexte-qualite-comptes-rendus/  
└─ 01-state/  
   └─ checkpoints.md
```

O arquivo pode tomar esta forma:

```
## 2026-09-11  
  
- Fontes: 3  
- Notas atômicas: 2  
- Notas temáticas: 1  
- Contradições abertas: 0  
- Decision records aceitos: 1  
- Próxima ação: iniciar a execução de DRE00001.  
  
## 2026-09-04  
  
- Fontes: 1  
- Notas atômicas: 1  
- Notas temáticas: 0  
- Contradições abertas: 0  
- Próxima ação: obter o ponto de vista da operação.
```

Essas fotografias me mostram a evolução do Contexto sem me obrigar a comparar todos os arquivos. Elas também me fazem perceber uma estagnação, ou uma dívida que cresce sem que eu note.

## A IA anota o que aconteceu, não o que ela acha

O histórico registra os acontecimentos importantes sem tentar reconstituir cada troca. No meu exemplo, os

acontecimentos do dia são acrescentados a 02-history/2026-09-11.md.

```
contexte-qualite-comptes-rendus/  
└─ 02-history/  
   └─ 2026-09-11.md
```

O bloco acrescentado pode tomar esta forma:

```
## 2026-09-11  
  
- Inclusão de SRC00003-livre-blanc-donnees-intervention.  
- Criação de AT000002 e THM00001.  
- Abertura de C-001.  
- Validação de `questions-prochain-entretien.md`.  
- Aceitação de DRE00001 e resolução de C-001.
```

- O estado diz em que ponto se está.
- O histórico diz o que aconteceu.
- A retomada diz o que fazer depois.

5

### Deixo a máquina conferir o que eu não veria

Na hora de salvar, algumas conferências mecânicas reduzem os esquecimentos. O resultado delas é escrito em 01-state/health.md.

```
contexte-qualite-comptes-rendus/  
└─ 01-state/  
   └─ health.md
```

- Todos os arquivos referenciados existem?
- Os identificadores são únicos?
- Cada nota indica as fontes dela?

- Uma fonte acrescentada mas não analisada é sinalizada?
- As decisões substituídas estão ligadas às que as substituem?
- A próxima ação está explícita?

Essas conferências não substituem o meu julgamento. Elas apenas impedem que um erro de estrutura se faça passar por um problema de pensamento.

### O ciclo completo, e o que ele permite

O movimento geral fica agora visível.

```
abertura
↓
leitura da caixa de correio
↓
carregamento da missão, do estado e da retomada
↓
inclusão de uma fonte
↓
criação ou atualização das notas
↓
produção de um rascunho
↓
conferência, validação ou decisão
↓
atualização do estado, da síntese e da retomada
↓
checkpoint e histórico
↓
conferência de salvamento
↓
próxima abertura
```

Não espero o fim dessa cadeia para salvar.

- Faço isso depois de incluir uma fonte, no meio da criação de uma nota, antes de uma decisão ou durante o preparo de

---

um entregável, quando vejo a minha janela de contexto estourar...

É justamente esse o papel do salvamento: **tornar cada um desses estados retomável, independentemente da conversa em curso.**

Minha reunião pode, então, começar.

**O Contexto não é só uma árvore de arquivos: é um sistema para pensar ao longo do tempo, que faz entrar o material, preserva as tensões, fixa ideias, produz vários resultados e prepara a retomada.**

A forma dele continua propositalmente simples e portátil — arquivos legíveis bastam para tornar o trabalho observável, conferível e transmissível.

Uma vez entendido esse motor, aparece outra pergunta: como o Contexto dialoga com as ferramentas, as bases de conhecimento e o produto real que existem fora dele?

---

#### PARA LEMBRAR — O SALVAMENTO

Salvar não marca o fim de uma etapa: é a garantia de que o estado sobrevive à sessão que o produziu. O estado diz em que ponto se está, o histórico o que aconteceu, a retomada o que fazer depois. O ciclo pode ser interrompido em qualquer ponto, e cada um desses pontos continua retomável.

---



# Parte 6

## O Contexto no seu ecossistema

---

Um Contexto bem cuidado dá vontade de colocar nele todo o conhecimento disponível.

Esta parte examina o que ele deixa de fora e por quê. Ele permanece pequeno para continuar capaz de raciocinar. Tudo o que guardasse apenas por precaução envelheceria sem aviso enquanto o original continuasse mudando em outro lugar. Por isso trabalha com vizinhos: ferramentas cuja função é conservar, medir ou executar, e outros Contextos que investigam perguntas diferentes da sua.

## Consultar conhecimentos estruturados

---

Os relatórios de serviço vão esperar: esta manhã trabalho em outro assunto do mesmo produto, as pessoas que desistem no meio do cadastro. Uma agência acaba de entregar seu estudo, e eu o acrescento ao Contexto: vários participantes hesitam diante das regras da senha.

Essa observação me lembra uma ideia que já encontrei: exigir um esforço grande antes que o usuário perceba o valor do produto pode levá-lo a desistir.

- Estou quase certo de ter material sobre o assunto.
- Simplesmente não sei mais onde ele está nem que palavras eu tinha usado.

Falei de esforço inicial, de atrito, de ativação ou de onboarding? A ideia talvez tenha sido

- tirada de um livro,
- desenvolvida numa nota de product management,
- ligada a outro percurso de usuário,
- ...

Eu poderia tentar várias buscas por palavras-chave e abrir os resultados um a um. Em vez disso, pergunto ao Contexto:

*O que já aprendemos sobre os esforços pedidos antes que o usuário perceba o valor do produto?*

**O Contexto consulta uma memória mais ampla**

A resposta não está no Contexto ativo. Em compensação, ela pode estar no meu LLM Wiki, uma biblioteca comum onde já estão estruturados os milhares de notas tiradas de livros, artigos, normas e outros documentos.

O LLM Wiki e o Contexto não têm a mesma natureza.

O wiki é uma base de conhecimentos.

- Ele guarda notas estruturadas (no meu caso reutilizei o conceito de notas atômicas) a partir do que outros escreveram: uma definição, um princípio, o resultado de um estudo, uma exigência de uma norma.
- Cada nota diz o que se sabe e de onde isso vem.
- Ele pode conter dezenas de milhares delas, acumuladas ao longo de anos e sobre assuntos sem relação entre si.

Nenhum Contexto jamais vai conter essa quantidade, e esse não é o papel dele.

Mas o wiki para aí.

- Ele não diz por que esse conhecimento importa no trabalho que conduzo hoje, nem o que fiz dele.
- Ele não liga uma ideia a uma decisão, nem uma decisão ao que ela me custou.

**O LLM Wiki não é um sistema de pensamento: ele armazena conhecimento em grande escala, mas não raciocina.**

O raciocínio vive no Contexto. É ele que guarda o porquê por trás do como:

- qual é a missão,
- que diretrizes e decisões estão estabelecidas,
- por que essa ideia foi escolhida para esta missão,
- quais alternativas descartei,
- o que aceitei perder,
- que contradição continua aberta.

Ele só consegue isso porque continua pequeno — algumas dezenas de notas que um humano consegue reler.

### **Cada um paga o preço da própria força.**

- **O wiki ganha a escala e perde o fio.**
- **O Contexto guarda o fio e abre mão da escala.**

É justamente por isso que eles devem conversar em vez de se fundir.

### **Descrevo a ideia que procuro**

Não copio o wiki inteiro no Contexto dedicado ao cadastro.

Também não executo uma busca RAG seguida de uma busca num grafo.

Faço simplesmente a minha pergunta ao Contexto:

*O que já aprendemos sobre os esforços pedidos antes que o usuário perceba o valor do produto?*

Formulo a pergunta com as minhas próprias palavras.

- Não preciso conhecer nem o título de uma nota, nem onde ela está, nem o vocabulário usado quando ela foi escrita.

O Contexto transmite essa ideia ao LLM Wiki pela API dele.

Todo o resto acontece dentro do wiki.

### **O wiki busca pelo sentido e depois em volta do resultado**

O LLM Wiki começa recuperando as notas cujo sentido se aproxima da minha pergunta.

- Uma nota pode falar de atrito inicial, de ativação ou de tempo até o primeiro benefício sem usar a palavra “esforço”.

É isso que o RAG permite fazer: encontrar conhecimentos já estruturados próximos da minha ideia sem exigir o mesmo vocabulário.

Depois o wiki olha em volta dos melhores resultados. O grafo semântico segue as relações já estabelecidas entre as notas e acrescenta as vizinhas que podem esclarecer o assunto:

- uma definição,
- um limite,
- uma contradição,
- uma ideia complementar.

Uma dessas notas vizinhas talvez nunca tivesse respondido diretamente à minha pergunta. Ela se torna preciosa mesmo assim, porque mostra em que condições a ideia se aplica, ou em que ponto ela deixa de ser verdadeira.

```
uma ideia formulada com as minhas palavras
      |
      ▼
o wiki reencontra as notas próximas pelo sentido
      |
      ▼
ele acrescenta a vizinhança útil delas
      |
      ▼
voltam algumas referências com a fonte
      |
      ▼
o Contexto guarda o que serve à missão dele
```

**O RAG reencontra o que eu não teria sabido nomear. O grafo faz aparecer o que eu não teria pensado em procurar.**

Para mim, essas duas operações formam uma só. Peço referências que respondam a um problema; o LLM Wiki procura a ideia e a vizinhança dela e depois me devolve uma lista curta.

**O Contexto recebe referências, não uma resposta pronta**

O wiki devolve notas existentes com o conteúdo, a origem e as relações delas. Ele não fabrica uma síntese que misturaria em silêncio vários resultados.

O Contexto pode, então, ler cada resultado, compará-lo com as observações sobre o cadastro e voltar à fonte correspondente quando uma prova precisa ser conferida.

- Certas notas dizem respeito mesmo a um esforço imposto antes do acesso ao produto.

- Outras falam de um esforço pedido mais tarde, quando o usuário já conhece o valor dele.

A proximidade delas com a minha pergunta não basta para torná-las equivalentes.

O `Contexto` faz a triagem a partir da missão dele. Ele pode:

- guardar uma ideia que esclarece a desistência durante o cadastro,
- guardar uma contradição,
- descartar uma nota geral demais.

Uma nota guardada entra no `Contexto` com a origem dela e as fontes em que se apoia. O resto permanece no wiki.

Disponho assim de uma memória bem mais vasta sem transformar cada `Contexto` numa cópia de toda a minha biblioteca.

- Não preciso aprender como essa memória está indexada para usá-la: descrevo a ideia que procuro e depois confiro as referências que ela me traz.

Essa cadeia dá acesso ao conhecimento já estruturado.

- Ela ainda não diz o que o produto faz de fato hoje.

Para responder a essa outra pergunta, o `Contexto` precisa sair da biblioteca e consultar diretamente o código, os testes e o produto em funcionamento.

**PARA LEMBRAR — DOIS SISTEMAS, UMA PORTA**

O LLM Wiki ganha a escala e perde o fio; o Contexto guarda o fio e abre mão da escala. Descrevo uma ideia com as minhas próprias palavras; o wiki reencontra as notas próximas pelo sentido e acrescenta a vizinhança útil delas. O Contexto recebe algumas referências acompanhadas de suas fontes e guarda apenas o que serve à missão dele.

---

## Consultar o produto real

---

Estou relendo o estudo sobre as desistências durante o cadastro quando chega uma pergunta:

*Um usuário convidado precisa criar uma senha como os outros?*

A pergunta parece minúscula. A resposta, porém, é esperada por uma pessoa do suporte que precisa responder a um cliente.

Eu poderia:

- interromper a minha leitura,
- abrir um ambiente de teste,
- preparar duas contas,
- refazer os percursos,
- comparar o que acontece.

Era o que eu fazia antes.

- Meia hora depois, eu provavelmente teria a resposta.
- Também teria perdido o fio do meu trabalho.

Desta vez, faço a pergunta ao Contexto.

### **O Contexto consulta o produto**

O Contexto trabalha com as notas saídas do estudo com usuários e guarda esse estudo como fonte das afirmações delas.

- Ele carrega também a missão em curso.
- Ele entende por que a pergunta importa, mas esses elementos não bastam para revelar o comportamento atual do produto.

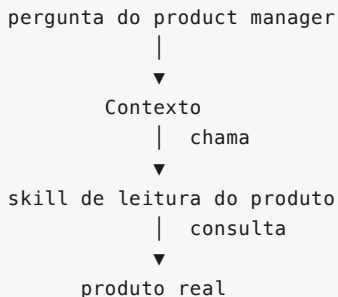
Não mantenho mais uma especificação encarregada de descrever esse comportamento de forma duradoura.

- O produto muda com frequência demais para que um documento escrito ao lado continue sendo a cópia fiel dele.

### **A única verdade atualizada é o que está entregue e rodando em produção: o código-fonte da aplicação.**

No meu sistema, skills externos ao Contexto sabem consultar o código-fonte e gerar documentação a partir da implementação. O Contexto chama aquele que responde às perguntas de produto e transmite a ele o meu pedido.

6



O skill começa localizando o percurso de convite no código. Depois segue esse percurso até a regra de criação da conta. O código mostra o comportamento programado. Um teste confirma que um usuário convidado consegue terminar o

cadastro pelo link recebido, sem criar uma senha na hora. A configuração indica que nenhuma variante própria de um cliente está ativa para esse percurso.

O skill me devolve uma resposta funcional:

*Um usuário convidado pode acessar a conta dele pelo link de convite sem criar uma senha durante essa etapa. Se depois ele escolher criar uma, valem as regras comuns. Os testes confirmam o percurso de convite e a criação voluntária da senha. O percurso de recuperação de uma senha esquecida não foi examinado.*

Posso repassar as duas primeiras frases ao suporte. Também posso abrir as referências fornecidas pelo skill se quiser conferir a análise dele com um desenvolvedor.

Por fim, posso registrar essa resposta no Contexto, com as provas dela, a data da conferência e o limite sinalizado.

- Responder ao suporte resolve um caso e para por aí.
- Registrá-la no Contexto faz entrar um fato verificado sobre o produto no *mundo das ideias* da missão: a próxima pergunta sobre o percurso de convite não vai partir do zero, e a nota saída do estudo ganha o que a observação sozinha não dava — o momento exato em que cada percurso pede a senha.

### **Uma resposta vale pelas provas que a acompanham**

O skill não se contentou em produzir uma resposta plausível.

Para cada afirmação, ele indica o que consultou:

- a regra presente no código,
- o teste que verifica o comportamento,
- a configuração que determina a variante ativa.

Esse rastro protege contra uma resposta convincente, porém falsa. Ele também permite ver na hora o que falta.

Aqui, o skill não conferiu a recuperação de uma senha esquecida. Ele diz isso em vez de estender a conclusão dele a um percurso que não leu.

Conforme a pergunta, outros elementos podem se tornar necessários.

- O código contém as regras possíveis, mas um feature flag indica qual está ativa.
- A estrutura do banco mostra o que o produto pode registrar, enquanto os dados autorizados permitem entender a situação de uma conta específica.
- O histórico Git e os pull requests ajudam a reencontrar quando uma regra mudou e as explicações guardadas com essa mudança.
- ...

O skill reúne apenas o que precisa para responder à pergunta feita. Ele não suga o repositório inteiro e não transforma cada arquivo em fonte do Contexto.

A resposta continua legível pelo suporte, pelo comercial ou por um product manager. As provas continuam disponíveis para quem precisa conferi-la ou levar a investigação adiante.

### Achar a regra não decide mudá-la

Volto ao meu estudo sobre o cadastro. Ele já tinha me ensinado que os dois percursos não apresentam a criação da senha no mesmo momento.

O skill acrescenta o que o estudo não conseguia ver. O usuário convidado não contorna a regra da senha: ele a encontra mais tarde, e ela é então exatamente a mesma. O atrito, portanto, não sumiu para ele, mudou de lugar. Essa precisão desloca a minha pergunta. Já não procuro saber quem sofre a regra, mas em que momento cada um a encontra: antes de ter percebido o valor do produto, ou depois.

O código me ensinou o que o produto faz.

- Ele não me diz se a regra é boa, se protege de fato as contas ou se o custo dela para os usuários é aceitável.

Para decidir, ainda preciso dos outros elementos do Contexto:

- as observações do estudo e o escopo delas;
- os objetivos de segurança;
- as razões que tinham levado a adotar essa regra;
- as consequências esperadas de uma mudança.

O skill estabelece um fato sobre o produto. O Contexto liga esse fato às notas e às decisões já presentes, guardando ao mesmo tempo as fontes que permitem conferir a origem dele.

### A decisão continua sendo minha.

Se eu decidir mudar a criação da senha, o decision record guarda a escolha, o escopo dela e as razões que a sustentam.

Depois da implementação, o skill pode voltar ao produto e conferir que o comportamento novo corresponde mesmo à decisão.

O movimento funciona, portanto, nos dois sentidos. O produto esclarece a reflexão, e depois a decisão transforma o produto.



## Guardar o porquê, reconstruir o quê

### A ABORDAGEM CODE-CENTRIC

O código já não é apenas o resultado final do desenvolvimento. Ele vira a fonte a partir da qual posso reconstruir o que descreve o comportamento atual do produto. O Contexto guarda o que o código não consegue explicar sozinho: as fontes, o raciocínio, as decisões, as alternativas descartadas e as razões da escolha.

6

Essa separação muda o lugar da documentação.

Guardo o que explica por que o produto tomou uma direção: as entrevistas, as restrições, as hipóteses e as decisões.

Reconstruo a partir do produto o que descreve o comportamento atual dele:

- uma resposta ao suporte,
- uma documentação funcional,
- um registro de mudanças,

- ...

Não mantenho mais manualmente duas versões do mesmo comportamento, uma no código e outra num documento que começa a envelhecer desde a publicação (e bem antes, inclusive).

Nem tudo vem do código, porém.

- Um estudo com usuários não está lá.
- Uma obrigação regulatória guarda a própria fonte.
- Uma decisão não se resume ao comportamento que resulta dela.

**O código registra o que foi implementado. O Contexto guarda o porquê e permite discuti-lo quando a situação muda.**

### **Voltar seis meses depois**

A resposta dada ao suporte fica associada à versão do produto examinada hoje. Ela não descreve automaticamente as versões seguintes.

- Seis meses depois, o percurso de convite pode ter mudado.
- Um feature flag pode ativar outra variante, ou um novo modo de acesso pode ter aparecido.

O Contexto guarda a pergunta, a resposta funcional, a data da conferência, a versão examinada, as provas citadas e o limite sinalizado. Assim reencontro o que sabíamos quando raciocinamos.

Para conhecer o comportamento presente, peço simplesmente ao skill que refaça a conferência.

- Ele parte do produto real em vez de copiar a resposta antiga.

Posso então distinguir duas perguntas que muitas vezes se confundem:

- por que tomamos aquela decisão na época?
- o que o produto faz hoje?

O Contexto responde à primeira guardando o raciocínio. O skill responde à segunda voltando à fonte de verdade.

O produto, portanto, não substitui o Contexto, e o Contexto não pretende conter o produto.

- Um fornece uma realidade a conferir;
- o outro guarda os conhecimentos necessários para dar sentido a ela e fazê-la evoluir.

Essa circulação não para nas ferramentas de fora. Quando vários Contextos trabalham em aspectos diferentes de um mesmo produto, os conhecimentos deles também precisam poder circular sem serem fundidos numa pasta única.

---

**PARA LEMBRAR — A ABORDAGEM CODE-CENTRIC**

Nenhuma especificação é mantida como descrição duradoura do produto: o que descreve o comportamento dele se reconstrói a partir do código. Uma resposta só vale pelas provas que a acompanham e pelos limites que ela sinaliza. O código registra o que foi implementado; o `Contexto` guarda o porquê e permite rediscuti-lo.

---

## O dia em que tive três Contextos

---

Um técnico me conta que precisou ligar de volta para um cliente por causa de uma informação que faltava num relatório de serviço.

Anoto a frase. Depois paro, porque não sei onde colocá-la.

- Tenho um Contexto sobre a qualidade dos relatórios de serviço — o que me ocupou durante toda a segunda metade deste livro.
- Tenho um segundo sobre a intervenção em si: o que ela contém, as regras que a governam, o que já foi decidido a respeito dela.
- E tenho um terceiro, mais amplo, aberto algumas semanas antes: como levar os operadores a assumir as intervenções de primeiro nível.

A frase do técnico diz respeito aos três:

- ela documenta um relatório deficiente,
- ela diz algo da intervenção,
- ela esclarece a questão dos operadores.

É o momento em que o problema muda de natureza.

- Enquanto eu tinha um só Contexto, a pergunta era: isso entra, ou está fora do assunto?
- Com três, a pergunta passa a ser: em qual deles?

### As duas maneiras de errar

A primeira tentação é juntar tudo.

- Um único espaço grande, todo o conhecimento no mesmo lugar, e nenhuma dúvida sobre onde guardar.

É exatamente do que eu fugia. Um espaço único que contém tudo recria o problema dos sistemas de gestão documental: um lugar onde

- a informação está presente e a triagem recomeça a cada consulta,
- os objetivos se misturam,
- as fontes se acumulam,
- e nenhuma nota tem mais um dono claro.

Contei esse beco sem saída no começo do livro; recriá-lo com um vocabulário novo não o tornaria melhor.

A segunda tentação é a oposta: que cada Contexto fique na casa dele. Cada um é limpo, delimitado, coerente — e cego.

O resultado aparece rápido. Meu Contexto sobre os relatórios passou três sessões consolidando uma regra que o da intervenção já carregava. Não era uma duplicata visível: era trabalho refeito, sem que ninguém pudesse notar.

É preciso, portanto, um terceiro caminho, e ele começa por saber o que cada Contexto carrega.

### Um número na frente do nome

Meus Contextos se chamam 01-saisie-compte-rendu, 02-

intervention, 03-autonomie-operateurs, 04-exploitation-catch-all.

O número não é uma posição, é uma convenção de nomes. Ele diz o que o Contexto carrega, antes mesmo de eu abri-lo.

- **02- uma entidade do produto.** Um pedaço do software que roda em produção e que tem as regras dele: 02-facturation num ERP, 02-intervention no meu. Seu conteúdo está estabelecido. Esse Contexto permanece ativo.
- **01- uma hipótese sobre uma entidade.** 01-saisie-compte-rendu: acho que o preenchimento deveria mudar, ainda não sei como. Nada do que ele contém é estabelecido, então crio um Contexto 01- para separar bem o que é estável (em produção) de uma hipótese.
- **03- um problema.** 03-autonomie-operateurs: levar os operadores a assumir as intervenções de primeiro nível. Não é uma funcionalidade, atravessa várias entidades e dá origem a 01-.
- **04- o catch-all.** Uma frase ouvida, um PDF, uma intuição: material cujo assunto ainda não tem nenhum Contexto. Sem ele, seria preciso criar um por fragmento de ideia — comecei assim e me vi com mais de cinquenta Contextos sem consistência em poucas semanas.

### **Um 01- sempre acaba saindo do trabalho ativo**

No dia da entrada em produção, o novo preenchimento deixa de ser uma hipótese: passa a ser uma parte da intervenção.

O que descreve agora o produto é proposto a 02-intervention.

O resto permanece no Contexto exploratório, que depois é

arquivado. Já não existem dois lugares ativos onde procurar como funciona um relatório de serviço.

Uma hipótese que não dá em nada é arquivada junto com o que a interrompeu. Não é um fracasso a apagar: é o que vai evitar refazer o mesmo caminho daqui a dois anos.

Já o 04- se esvazia no outro sentido. No dia em que um problema surge, crio o Contexto correspondente e depois volto a ver o que o catch-all tinha guardado a respeito. O que não serve para nada é apagado.

Tudo isso resolve uma única pergunta — onde coloco isso — separando o que estou investigando do que está estabelecido.

- Sem essa separação, cada hipótese abandonada deixa marcas na descrição do produto.
- E ninguém mais ousa supor nada num lugar onde tudo deveria ser verdade.

### **O erro que levei meses para ver**

Comecei abrindo 03-tarification. Parecia óbvio: é um assunto grande, envolve várias entidades e dura muito tempo.

Seis meses depois, ele continha:

- a tabela de preços,
- dois comparativos com concorrentes,
- uma discussão sobre descontos,
- um chamado de suporte sobre uma fatura errada,
- uma nota sobre o IVA belga,
- três capturas de tela de um concorrente,
- ...

Nada disso estava fora do assunto: tudo falava mesmo de precificação.

Esse é exatamente o problema. Para recusar um material, é preciso poder dizer “isso não me ajuda”. Ora, esse Contexto não procurava nada — **ele não tinha pergunta**, só um rótulo.

03-autonomie-operateurs, esse, tem uma: como levar os operadores a assumir as intervenções de primeiro nível.

Um 03- não é, portanto, um assunto grande. É uma pergunta. A precificação, por sua vez, não sumiu por isso: ela atravessa os meus Contextos — 02-facturation carrega as regras dela, um 01- pode depender dela — sem comandar nenhum.

Minha convenção não tem, então, casa para os domínios, nem para muitas outras coisas. Ela serve para guardar o que tem um lugar evidente, e nada mais. O resto espera no catch-all, e eu o reencontro descrevendo o que procuro em vez de tentar lembrar onde o guardei.

6

Hoje tenho dois, uma solução que aprendi por tentativa e erro. Um único catch-all misturava o que vinha do meu trabalho de produto com o que vinha do domínio da operação industrial — dois vocabulários, dois públicos, duas maneiras de nomear os mesmos objetos. Ele ficou ilegível mais rápido do que se esvaziava.

- 04-exploitation-catch-all recebe o que pertence ao domínio de negócio dos meus clientes: a operação;
- 04-productops-catch-all recebe o que diz respeito ao meu trabalho e à organização dele: product ops.

---

Cada um continua legível, então cada um se esvazia.

A frase do técnico, essa, não precisou do catch-all. Ela foi para 02-intervention, porque é ali que vive o que é verdadeiro do produto independentemente da pergunta que eu faço neste momento.

Resta saber como os meus três Contextos tomam conhecimento disso.

---

#### **PARA LEMBRAR — QUATRO PREFIXOS, UMA SÓ PERGUNTA**

O número na frente do nome diz o que o Contexto carrega: 02- uma entidade que roda, 01- uma hipótese sobre essa entidade, 03- um problema que atravessa várias, 04- a área de espera onde se coloca o que ainda não se sabe onde guardar. Um 01- existe para propor ao 02- dele o que descreve agora o produto, e depois para ser arquivado. Se não der em nada, é arquivado junto com o que o interrompeu. Um domínio não é um Contexto: ele atravessa vários sem comandar nenhum. Um Contexto sem pergunta nunca vai saber o que está fora do assunto.

---

## Ler na casa do vizinho sem se mudar para lá

---

Retomo `01-saisie-compte-rendu`, o Contexto dedicado à qualidade dos relatórios de serviço. Para avançar, preciso conhecer as regras da intervenção em si: o que um técnico tem direito de preencher, o que é obrigatório e o que já foi decidido.

Essas regras não vivem no meu Contexto. Elas estão fixadas em `02-intervention`, o Contexto vizinho que descreve o que existe realmente no produto.

Eu poderia copiá-las para tê-las à mão. Seis meses depois eu teria provavelmente duas versões da mesma regra e nenhum jeito simples de saber qual vale.

Começo, então, por uma relação mais leve: o meu Contexto declara que consulta `02-intervention`.

### Ler deixa o conhecimento com o dono dele

Essa relação está escrita no quadro estável do meu Contexto, ao lado da missão e das diretrizes dele. Ela abre um direito preciso: consultar o que `02-intervention` já sabe.

O material continua com o dono dele. Ele continua evoluindo lá, e a minha exploração se apoia nele sem pretender possuí-lo.

Assim posso conferir a regra atual sobre os campos obrigatórios sem duplicá-la em cada estudo que envolva um

relatório de serviço. Também posso voltar às fontes dela e às decisões que a construíram.

Ler não autoriza modificar. Se uma nota me parece incompleta ou falsa, o meu Contexto pode formular uma proposta, mas ele nunca corrige diretamente o vizinho. Essa primeira relação atende a uma necessidade comum: usar um conhecimento de fora deixando-o viver no lugar que é responsável por ele.

Ela não atende a todos os casos.

### **Absorver torna o conhecimento autônomo**

Às vezes uma simples consulta não basta.

Posso precisar congelar a regra exata em que se apoia uma decisão. Posso querer reformulá-la para o escopo específico da minha exploração. Também posso precisar que esse conhecimento continue disponível se o Contexto de origem for arquivado, movido ou reorganizado.

Nesse caso, escolho absorvê-lo.

Absorver faz entrar um conhecimento no meu próprio Contexto. Posso incorporar uma nota como ela está ou reformulá-la no meu vocabulário. Conforme a necessidade, copio também as fontes que permitem conferi-la, ou guardo referências para elas.

A cópia não vira, por isso, uma ideia sem passado. Ela guarda o Contexto de que vem, a data da absorção e o que foi reformulado.

A diferença está, portanto, menos no gesto técnico do que na responsabilidade que resulta dele:

- quando leio, o conhecimento continua pertencendo ao vizinho;
- quando absorvo, uma versão autônoma entra no meu próprio Contexto, e eu me torno responsável pelo uso dela.

**Ler cria uma dependência. Absorver cria uma responsabilidade nova.**

### O movimento pode voltar no outro sentido

Minha exploração não se limita a consumir o que 02-intervention já sabe. Ela produz, por sua vez, observações, hipóteses e às vezes um conhecimento duradouro.

Estudando a qualidade dos relatórios, posso descobrir que uma regra de preenchimento pertence na verdade à intervenção em si. Esse conhecimento não deveria ficar trancado num Contexto exploratório que talvez acabe arquivado.

Não o escrevo diretamente em 02-intervention. Eu o submeto a esse Contexto.

O sentido da circulação acaba de mudar:

01-saisie-compte-rendu — lê → 02-intervention

01-saisie-compte-rendu — propõe → 02-intervention

└ decide se absorve ou não

Quem produz um conhecimento não decide o lugar definitivo dele. O Contexto que o recebe o avalia segundo a própria missão.

Ele pode:

- incorporar a nota como ela está;
- reformulá-la no vocabulário dele;
- guardar uma divergência quando as duas leituras continuam válidas;
- recusar a absorção quando o que é útil à exploração não descreve a intervenção em si.

A recusa não é um fracasso da circulação. É o que impede que um resultado local se transforme silenciosamente numa verdade geral.

### **Quando a exploração vira uma realidade do produto**

Algumas semanas depois, o novo preenchimento dos relatórios entra em produção. 01-saisie-compte-rendu cumpriu o papel dele: o que era só uma hipótese virou uma realidade do produto.

Nem por isso movo todo o conteúdo dele para 02-intervention. Passo em revista o que a exploração produziu e separo dois conjuntos.

O primeiro contém o que agora descreve ou explica o produto:

- as notas que estabelecem as novas regras de preenchimento;
- o vocabulário fixado para nomeá-las;
- as decisões estruturantes e as razões dessas escolhas;

- as contradições que continuam abertas depois da entrada em produção;
- as fontes ou as referências necessárias para conferir esse conhecimento.

Esses elementos são propostos a *02-intervention*. O Contexto estável não os acumula às cegas: ele enriquece as notas que já tem, reformula o que precisa ser reformulado e só cria uma nota nova quando uma ideia é realmente nova.

No nosso exemplo, a regra que limita o encerramento a quatro informações vai para *02-intervention* com a decisão que a explica. Ela descreve agora uma realidade do produto.

O segundo conjunto conta a história do trabalho sem descrever o produto atual:

- as hipóteses abandonadas;
- os protótipos e as alternativas descartadas;
- os rascunhos intermediários;
- as tarefas próprias da exploração;
- as discussões que permitem entender o caminho percorrido.

Esses elementos ficam em *01-saisie-compte-rendu*. Eles não se tornaram inúteis: explicam o que foi tentado, por que certos caminhos foram descartados e o que não deverá ser recomeçado sem uma razão nova.

Terminada essa seleção, o Contexto exploratório é arquivado por inteiro. Ele continua legível e pode ser retomado mais tarde, mas já não pretende descrever o trabalho ativo.

Se a exploração não tivesse dado em nada, ela também teria sido arquivada, com as razões do abandono, sem impor nada a 02-intervention.

A entrada em produção não funde, portanto, dois Contextos. Ela desencadeia uma revisão: o que agora descreve o produto vai para o Contexto estável; o que conta a exploração fica no Contexto arquivado.

Essa seleção protege os dois lados:

- a exploração continua livre para errar;
- o Contexto estável só recebe o que escolhe reconhecer como seu.

### **As relações fazem parte do quadro**

Ler, absorver e propor não devem continuar sendo gestos implícitos.

O quadro estável do Contexto indica:

- que outros Contextos ele pode consultar;
- o que essa consulta autoriza;
- segundo que regras um conhecimento pode ser absorvido;
- como a origem dele e as eventuais reformulações são guardadas.

Essa governança não pede uma infraestrutura pesada. Ela apenas dá um endereço às relações para que a procedência, as responsabilidades e as fronteiras continuem visíveis.

A rede não se constrói, portanto, ligando tudo a tudo. Ela se constrói relação por relação, cada vez que um trabalho precisa de fato de um vizinho.

- Ler na casa do vizinho evita duplicar o que ele sabe.
- Absorver permite tornar autônomo o que deve viver na minha casa.
- Propor permite, por fim, que o vizinho decida se quer integrar o que a minha exploração produziu.

Essa circulação parece limpa no papel. Ela tem, no entanto, um preço: uma referência pode envelhecer sem avisar, uma cópia pode perder a origem dela e dois Contextos podem guardar leituras diferentes.

É o que o capítulo seguinte vai examinar.

---

#### **PARA LEMBRAR — TRÊS GESTOS, TRÊS RESPONSABILIDADES**

Ler consulta um conhecimento que fica com o dono dele; absorver cria uma versão autônoma, acompanhada de suas fontes, no Contexto que a recebe. Uma exploração propõe o que aprendeu, mas o dono do Contexto de destino decide o que entra na casa dele. A entrada em produção não funde as pastas: o Contexto estável absorve o que agora descreve o produto, enquanto a exploração guarda o resto antes de ser arquivada.

---

## O que a circulação custa

---

Passaram-se seis semanas.

Meu trabalho sobre os relatórios de serviço está pronto para ir à revisão, e alguém aponta uma inconsistência. A regra em que me apoio — a que diz quais campos são obrigatórios no encerramento — já não é a do produto.

Ela mudou. Enquanto eu trabalhava, outra exploração chegou ao fim, e o Contexto da intervenção absorveu seus resultados.

Ninguém me avisou. Não o reli, porque o tinha lido seis semanas antes e nada sinalizou que fosse preciso voltar lá. O capítulo anterior estabelece uma regra que parece simples: referencia-se no sentido da leitura, copia-se no sentido da absorção. Eu a tomei por uma solução definitiva.

Ela não elimina o risco. Ela o desloca, e é melhor saber onde ele recai.

### **Uma dependência que não se vê**

O que acaba de acontecer tem um nome: uma dependência cega.

O vínculo está vivo, então o que eu leio pode mudar sem mim. Não é esse o problema — muitas vezes é exatamente o que quero. Uma regra que evolui na casa do dono dela é o sistema funcionando.

O problema é não saber que ela mudou.

O que falta é simples de enunciar: nada acompanha o que eu leio. Meu Contexto sabe de quem depende; ele não sabe quando aquilo de que depende mudou. É a metade do mecanismo, e é justamente a parte custosa.

Uma cópia teria me protegido disso. O Contexto que recebe se torna autônomo: já não depende nem da existência nem da evolução da fonte dele.

Mas ele perde outra coisa em troca. Se nada diz de onde vem a cópia, a procedência desaparece — e um ano depois tenho uma nota correta da qual já não sei em que ela se apoia. É precisamente a situação que toda a rastreabilidade deste livro procura impedir.

A conclusão não é, portanto, escolher um lado. É registrar o que cada ação precisa preservar:

- uma **referência** exige saber quando a fonte dela muda;
- uma **cópia** exige dizer de onde ela vem — que Contexto, em que data, e o que foi reformulado no caminho.

---

### EVOLUÇÃO POSSÍVEL

Outra resposta, ainda por vir, será tecnológica: os Contextos poderão avisar seus observadores quando o estado mudar, sobretudo quando a mudança afetar o escopo acompanhado por outro Contexto. O Contexto é um tema vivo, em que muitas possibilidades continuam por explorar.

---

### **Os direitos que a leitura não dá**

Quando leio muito um Contexto, ele acaba parecendo um pouco meu. Várias vezes tive vontade de corrigir diretamente uma nota na casa do vizinho — ela estava incompleta, eu sabia o que pôr nela, e teria levado trinta segundos.

A regra não muda por isso: o que tenho a lhe dizer é uma proposta, e a decisão pertence a quem recebe.

### **Apagar sem quebrar a rastreabilidade**

A exclusão exige uma regra à parte, e eu a aprendi esvaziando uma área de espera.

Quando uma nota muda de Contexto, as fontes que a produziram ficam para trás. Elas podem ficar isoladas, sem mais nenhuma nota ligada a elas.

O reflexo é fazer faxina. Ele só faz sentido em parte.

Uma fonte já lida e transformada em notas pode ser apagada: o que ela carregava vive em outro lugar. Uma fonte que ninguém leu ainda não pode ser apagada.

- Ela carrega um material que ninguém extraiu, e jogá-la fora equivale a perder o que nem sequer foi aberto.

O critério não é, portanto, nem a idade do arquivo nem o espaço que ele ocupa.

- É saber se a fonte já foi lida e verificar o que ainda a cita.

Há uma condição, e ela não se negocia. Antes de apagar, conferir que nenhuma nota, nenhuma entrada de vocabulário, nenhum rascunho e nenhuma linha do registro cita mais essa

fonte. Sem essa conferência, a regra quebra a rastreabilidade a que diz servir.

### **Uma divergência não é um erro**

Resta o caso em que dois Contextos não dizem a mesma coisa do mesmo assunto.

O reflexo é ver aí uma incoerência e decidir rápido para restabelecer a ordem. Muitas vezes, essa conclusão está errada.

Uma divergência pode ser uma diferença de escopo: cada um pode estar correto dentro dos seus próprios limites. Pode ser uma hipótese ainda aberta que um investiga e o outro ignora. Podem ser duas leituras que se sustentam as duas, porque olham o assunto de dois lugares.

Ela merece então ser guardada como tal, e cabe à pessoa decidir isso. O que o sistema deve garantir é que se saiba que ela existe.

6

Uma divergência esquecida sempre reaparece no pior momento: quando alguém se apoia numa das duas versões sem saber que a outra existe. Foi o que me aconteceu com a regra de encerramento, com a diferença de que não eram duas leituras, mas uma só que tinha envelhecido.

Manter a rede confiável tem um custo, e prefiro nomeá-lo a descobri-lo na revisão. Ligar dois Contextos não elimina o trabalho de vigilância: ele o torna possível, e o desloca para o humano.

É, no fundo, o que este livro conta desde o começo.

---

**PARA LEMBRAR — O PREÇO DA REDE**

Uma referência pode envelhecer sem avisar, uma cópia pode perder a procedência dela: cada gesto tem uma dívida, e é preciso escrever qual. Ler não autoriza escrever; o que se tem a dizer ao vizinho é uma proposta, nunca uma correção. Uma fonte já transformada em notas pode desaparecer; uma fonte nunca lida não pode — e nada se apaga sem conferir quem ainda a cita. Dois Contextos que se contradizem não estão necessariamente errados: grave é ignorar isso.

---



# Antes de fechar

---

Encerramento



## Encerramento

---

O livro para por aqui, mas o assunto não para com ele.

O epílogo diz em que ponto está este trabalho, o que ele apenas abordou de passagem e onde serão publicados os próximos desdobramentos. A seção sobre abordagens relacionadas situa o Contexto entre aquelas que o precederam ou que o cercam, para que você saiba o que ele toma emprestado e o que não retoma.

## Epílogo

---

Obrigado por ter chegado até aqui.

Tentei deixar este livro o mais legível possível, e não consegui totalmente. Restam trechos mais áridos do que eu gostaria, lugares em que a mecânica passa à frente da narrativa. Você aguentou mesmo assim: é um presente que valorizo de verdade.

### **Este livro apenas tangenciou o assunto**

Apesar da espessura dele, aborda o Contexto apenas superficialmente.

Várias partes continuam leves no plano conceitual, e eu sei disso ao publicá-las. As notas atômicas, por exemplo: dependendo de como eu as implemento, obtenho resultados muito desiguais. O que funciona num Contexto exige adaptações assim que se passa para um LLM Wiki, onde as informações não têm nem o mesmo volume nem o mesmo uso.

Essas zonas merecem ser reforçadas. Não foram aqui porque um livro precisa parar em algum momento, não porque o assunto esteja encerrado.

### **Um trabalho que continuará sendo um projeto paralelo**

O Contexto vai continuar avançando, mas sempre como side project.

É uma atividade levada à margem do meu ofício, e já dediquei a ela noites demais. Vou voltar a ele — sempre volto — mas está na hora de diminuir o ritmo e aproveitar a vida.

### **O que eu gostaria de explorar depois**

A pesquisa de trabalhos anteriores eu fiz. Ela trouxe à tona conceitos muito próximos do meu, às vezes em campos distantes do produto e da gestão do conhecimento, muitas vezes em outras línguas — o japonês e o chinês principalmente, cujos trabalhos continuam em grande parte invisíveis para o mundo francófono e anglófono. Escrevi um artigo sobre essa busca; ele sairá no blog.

O que falta é o passo seguinte: analisar esses trabalhos.

Os nomes estão identificados, os artigos estão ali, e eu não os li. São décadas de pesquisa feitas por centenas de pessoas sobre questões que abordei sozinho, à noite. Estudá-los a fundo reforçaria esse quadro muito mais do que continuar inventando-o.

### **A continuação não será um volume II**

Este livro serviu para cristalizar o conceito. Era o papel dele: reunir o que estava espalhado, torná-lo transmissível, dar-lhe uma forma estável o bastante para que se possa discuti-la.

Por um tempo, as novidades serão publicadas em [malorean.net](http://malorean.net), e não num volume II nem numa nova versão desta obra. Um artigo se corrige, se completa, se contradiz. Um livro congela — e o que está descrito aqui ainda não está maduro o bastante para ser congelado duas vezes.

Se você quiser acompanhar a continuação, é lá que ela será escrita.

## Parentescos e vizinhanças

---

Contexto não apareceu num vazio intelectual.

A maioria dos mecanismos que o compõem tem parentes, alguns bem antigos, na gestão do conhecimento, nos métodos de tomada de notas, na engenharia de software, nos sistemas de apoio à decisão, nas ciências cognitivas ou ainda na memória organizacional.

Algumas dessas influências foram conscientes desde o começo. Outras só foram descobertas depois de o método estar desenhado, durante uma pesquisa de trabalhos anteriores feita com o livro quase pronto.

É preciso, portanto, distinguir duas coisas.

Uma **influência** é um método ou uma ideia que eu conhecia e de que retomei ou adaptei conscientemente alguns princípios.

Um **parentesco** ou uma **vizinhança** designa um sistema anterior que resolve um problema próximo, às vezes com mecanismos surpreendentemente parecidos, sem ter influenciado diretamente o Contexto.

Essa distinção é importante. Duas ideias podem ser próximas sem que uma derive da outra.

## **Zettelkasten: o conhecimento como rede de unidades reutilizáveis**

O parentesco com o Zettelkasten é provavelmente o mais evidente.

No sistema de Niklas Luhmann, uma nota não é simplesmente guardada numa pasta correspondente ao projeto do momento. Ela entra num sistema duradouro de notas ligadas entre si e pode ser reutilizada em novos raciocínios.

Contexto compartilha plenamente essa ideia: um conhecimento produzido durante um trabalho não deveria sumir junto com o entregável dele.

As notas atômicas do Contexto pertencem claramente a essa família.

A diferença principal está no estatuto delas. Uma nota atômica precisa guardar a procedência e se inscreve numa arquitetura que distingue explicitamente fontes, contradições, decisões, estado atual, síntese e entregáveis.

O Zettelkasten oferece, portanto, um parentesco muito forte para a **construção progressiva de um patrimônio de conhecimentos**, mas não para o conjunto do sistema.

### **PARA: organizar a informação em função da ação**

PARA, de Tiago Forte, faz parte das influências conscientes do Contexto.

PARA organiza a informação em quatro categorias: Projects, Areas, Resources e Archives. Uma das intuições importantes

dele é que uma organização da informação deve servir à ação, e não a uma classificação abstrata do mundo.

Contexto guarda algo dessa intuição.

Um Contexto não existe porque um assunto existe. Ele existe porque um humano persegue uma missão.

A fronteira não é, então: “tudo o que eu sei sobre bancos de dados”.

Ela poderia ser: “escolher uma arquitetura de dados para o produto X”.

A diferença é que os próprios Contextos viram unidades de trabalho autônomas, com o próprio estado, as próprias fontes, os próprios conhecimentos, as próprias decisões e o próprio ciclo de vida.

### **Domain-Driven Design: a fronteira antes do conteúdo**

O próprio termo Contexto evoca naturalmente o **Bounded Context** do Domain-Driven Design.

O parentesco está sobretudo numa ideia: uma informação ou um conceito só faz sentido dentro de certa fronteira.

A mesma palavra pode ter um significado diferente em dois contextos diferentes sem que seja preciso fundi-los.

Contexto transpõe essa intuição.

Mas, em vez de fazer a fronteira depender de um modelo de negócio coerente, ele a faz depender de uma **missão humana**.

Dois Contextos podem, portanto, guardar compreensões diferentes do mesmo assunto porque perseguem objetivos diferentes.

### **Git: divergência, autonomia e reintegração**

Git é uma influência mais diretamente técnica.

Quando uma exploração ganha importância suficiente, pode ser melhor fazê-la evoluir num Contexto separado em vez de perturbar o espaço estável.

Essa lógica se parece com um branch.

A exploração pode divergir, produzir as próprias fontes e conhecimentos, e depois algumas conclusões podem ser reintegradas ao espaço de origem.

Mas, como no Git, integrar não significa necessariamente fundir tudo.

Essa intuição também influenciou as relações entre Contextos: autonomia local, divergência possível e integração seletiva.

### **Project Memory: guardar o porquê do projeto**

Os trabalhos sobre a **memória de projeto** são um parentesco que eu não tinha identificado ao conceber o Contexto.

O problema colocado é quase exatamente o que aparece quando um trabalho dura vários meses: as organizações guardam com certa facilidade os documentos finais, mas muito menos bem o contexto que explica como e por que o projeto chegou àquele resultado.

Por que esta decisão?

O que tentamos antes?

Que restrições existiam naquele momento?

O que uma pessoa nova precisaria entender antes de continuar?

Contexto responde às mesmas perguntas com várias memórias distintas: o estado atual, a história, as decisões, a síntese e o handoff.

A proximidade funcional é, portanto, muito forte.

### **Experience Factory: o entregável não é o patrimônio**

A Experience Factory, desenvolvida em torno dos trabalhos de Victor Basili e experimentada sobretudo na NASA, traz um parentesco ainda mais importante.

Ela distingue a organização que realiza um projeto daquela que transforma a experiência do projeto em conhecimento reutilizável para os projetos seguintes.

Essa ideia se conecta diretamente a uma distinção fundamental do Contexto:

#### **| o entregável não é o patrimônio.**

Um projeto pode produzir um excelente resultado e não deixar quase nada atrás de si.

Ao contrário, a experiência dele pode ser analisada, transformada e guardada de modo a aumentar a capacidade de resolver os problemas seguintes.

Contexto integra essa capitalização diretamente no espaço onde o trabalho é feito.

## **Knowledge-Centered Service: produzir o conhecimento trabalhando**

Knowledge-Centered Service — KCS — é provavelmente um dos vizinhos mais interessantes do Contexto.

O princípio dele é que a documentação não deve ser uma atividade separada, feita depois do trabalho.

Quando um problema é resolvido, o conhecimento existente é procurado e reutilizado. Se ele é imperfeito, é melhorado. Se não existe, é criado.

O simples fato de trabalhar vira assim um mecanismo de criação e de manutenção do conhecimento.

É exatamente uma das ambições do Contexto:

*cada uso do sistema deveria poder aumentar o capital informacional dele.*

O KCS, no entanto, é pensado principalmente para ambientes de serviço e de suporte, enquanto o Contexto procura fornecer uma estrutura genérica para muitas missões.

## **HWMSE: humanos, máquinas e conhecimentos num mesmo espaço**

O parentesco mais surpreendente descoberto durante a busca vem provavelmente da China.

O **Hall for Workshop of Meta-Synthetic Engineering**, ou HWMSE, é desenvolvido em torno dos trabalhos de Qian Xuesen a partir do início dos anos 1990.

Ele é concebido para tratar problemas complexos combinando explicitamente:

- especialistas humanos;
- sistemas máquina;
- sistemas de conhecimentos.

Mais interessante ainda, os trabalhos sobre HWMSE não falam apenas de usar conhecimento existente. Falam também de **criação de conhecimento** e de um sistema de conhecimento próprio do problema estudado que se enriquece continuamente durante o uso.

A proximidade com o Contexto é, portanto, importante:

- uma atividade orientada para um problema;
- uma colaboração humano-máquina;
- recursos e modelos mobilizados durante o trabalho;
- conhecimento produzido pela atividade;
- um patrimônio que se acumula.

O HWMSE não tem, no entanto, a mesma gramática operacional do Contexto. Não se encontra lá, sob essa forma, a separação entre fonte, nota atômica, estado, síntese, contradição, handoff e entregável.

Mas, entre os sistemas estudados, é provavelmente um dos vizinhos arquiteturais mais próximos.

### **ScholOnto e SWAN: uma proposição não é a fonte dela**

Outro parentesco forte aparece nos sistemas dedicados ao discurso científico.

ScholOnto e, mais tarde, SWAN distinguem os documentos científicos das **proposições produzidas a partir desses documentos**.

Um recurso pode servir de origem a uma afirmação.

Essa afirmação guarda a procedência dela.

Ela pode ser sustentada, discutida ou contradita por outros elementos.

Isso é extremamente próximo do princípio que sustenta as notas atômicas do Contexto:

*a fonte continua sendo uma fonte; o conhecimento  
construído a partir dela é outro objeto.*

Esses sistemas mostram também que uma proposição registrada num sistema de conhecimentos não deve ser automaticamente considerada verdadeira.

Ela tem uma procedência e um estatuto dentro de um raciocínio.

É um parentesco especialmente forte com a arquitetura epistêmica do Contexto.

7

### **Sistemas federados: ler não é copiar**

Por fim, algumas ideias sobre as relações entre Contextos têm parentes surpreendentemente antigos nos sistemas de bancos de dados federados.

Essas arquiteturas precisam permitir que vários bancos autônomos cooperem sem abrir mão da autonomia deles.

Um sistema pode expor certas informações.

Outro pode consultá-las à distância ou decidir guardar uma cópia local.

E assim que uma cópia é criada aparece o problema clássico: o que acontece quando o original muda?

É exatamente o que resume no Contexto a fórmula:

***referência = dependência; cópia = responsabilidade.***

Os mecanismos de informática já existiam, portanto, havia muito tempo.

Contexto os transpõe em regras de governança do conhecimento entre espaços de trabalho compartilhados por humanos e IA.

### **Uma síntese mais que uma invenção do nada**

Essas vizinhanças tornam difícil apresentar o Contexto como a invenção de mecanismos inteiramente novos.

- Não é o que a história mostra.
- A procedência existia.
- As notas atômicas existiam.
- A memória dos projetos existia.
- A capitalização entre projetos existia.
- A criação de conhecimento durante o trabalho existia.
- Os ambientes que combinam humanos, máquinas e sistemas de conhecimentos existiam.
- As asserções com fonte e contestáveis existiam.
- As arquiteturas autônomas capazes de compartilhar ou copiar informação existiam.

O que parece caracterizar mais o Contexto é a maneira como esses mecanismos são **selecionados, combinados e normalizados dentro de um mesmo método**.

O método lhes dá uma gramática comum:

- fontes;
- notas atômicas;
- contradições;
- decisões;
- estado;
- história;
- síntese;
- ações;
- handoff;
- entregáveis;
- relações entre Contextos.

E, sobretudo, ele aplica essa montagem a um problema muito atual: manter uma colaboração duradoura entre um humano e uma IA sem transformar cada conversa num episódio isolado e amnésico.

Contexto tem, portanto, muitos parentes.

Mas, como costuma acontecer, conhecer os próprios parentes serve sobretudo para entender melhor o que distingue o próprio trabalho.