

The background is a vibrant yellow, overlaid with a complex network of black and white dots connected by thin black lines. Some lines are solid, while others are dashed, creating a sense of depth and connectivity. The dots vary in size, with some being larger and more prominent than others. The overall effect is reminiscent of a molecular structure or a network diagram.

LAURENT DÉSÉCHALLIERS

コンテクト

AIが重要なことを理解し、
知識を結びつけ、
学びを目的に役立てるための
コンテクトをつくる。

Le Contexte

AI が重要なことを理解し、知識を結びつけ、学びを目的に役立つ形で蓄積できるコンテキストを、どう築くか。

翻訳について

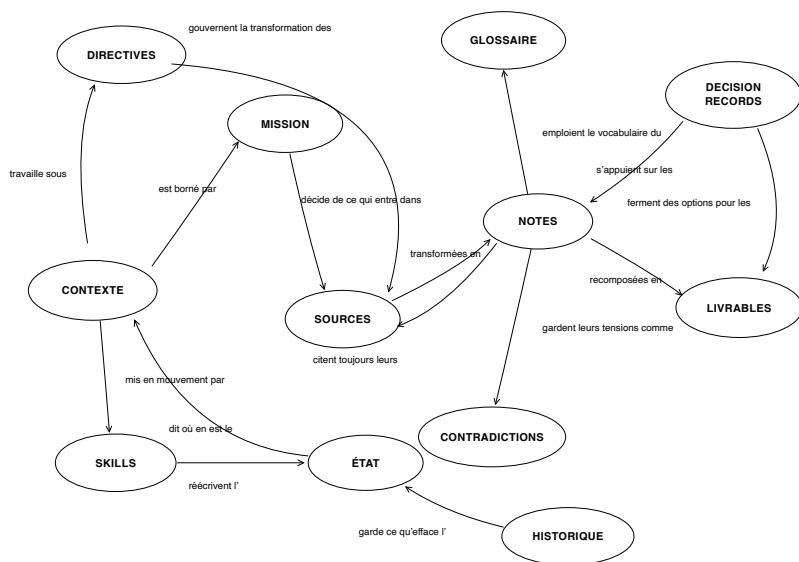
本版は、フランス語の原文をもとに AI が作成した機械翻訳です。

人間の翻訳者による校閲は行われていません。内容、構成、事例は原文に忠実ですが、言い回しがぎこちなく感じられる箇所や、原語の痕跡が残る箇所があります。

本書をフランス語だけでなく複数の言語で届けるために、あえて受け入れた代償です。表現に疑問がある場合は、フランス語版を基準としてください。

コンテキストの定義

コンテキストとは、人間が定めた目的によって範囲を区切ったシステムである。そこではAIが、わかっていること、その根拠、情報同士のつながり、現在の作業状況を整理し、必要に応じて活用する。人間がゼロからやり直さずに目的を達成できるよう支援するためだ。こうしてシステムは、長く使える情報資産を築き、育てていく。生み出された知識は利用できる状態で残り、時間とともに充実し、別の目的にも再利用できる。



はじめに

これほど多くの情報が手元にある時代はありませんでした。それでも私たちは、情報を探し直し、並べ直し、数日前に理解していたはずのことを組み立て直すことに、いまだ多大な労力を割いています。

役に立つ考えが、整った形で現れることはめったにありません。やり取りのなかで生まれ、文書のなかで輪郭を得て、会議で反論に出会い、決定を生み、やがて記事や仕様書やプロジェクトになります。私たちの道具は、その一つひとつの痕跡を残してくれます。しかし、それらをつなぐ糸を保つことは苦手です。

人工知能は、この問題の一部を変えました。AI は情報を読み、照らし合わせ、言い換え、形式の異なる情報を変換できます。けれども、AI との対話は途切れやすいものでもあります。話が進み、やり取りが長くなると、目的も、下した決定も、使った資料も、残っている作業も、改めて伝え直さなければなりません。この難しさに答えるために、私はコンテキストを築きました。

コンテキストとは、ひとつのテーマに特化した作業空間です。そこで人間と AI は、目的、資料、アイデア、ルール、作業状況を保ち続けます。毎回すべてを一から整理し直さずに、成果を生み出すためです。

すると AI は、実際の履歴を踏まえて作業できるようになります。アイデアの出どころを見つけ、過去の選択を思い出させ、矛盾や根拠の不足を指摘し、中断した草稿を続きから再開します。

AI は思考を整理し、検討する手助けをします。意味づけと意思決定の責任は、あくまで人間にあります。

このシステムは、私の日々の仕事のなかで、たった一つのファイルと、AI へのごく単純な求めから生まれました。これから話すことを整理してほしい、という求めです。以後、試行錯誤を重ねて育ちました。困難にぶつかるたびに、足りないものが見えてきたのです。今も変わり続けています。ここで示す解のいくつかは複数の場面で持ちこたえましたが、いくつかは、私がいまだ出会い続ける問題への暫定的な答えにすぎません。

ですから本書は、完成した理論から出発しません。出発点は、実際に直面した問題です。追いきれなくなった五百通のメール、いくつものツールに散らばったアイデア、すでに済ませた仕事を失わずに回答しなければならない顧客文書、中断後に再開する原稿です。

本書は、情報を理解や意思決定、成果物へと変える人たちに向けて書かれています。プロダクトマネージャー、コンサルタント、ライター、研究者、アナリスト、そして広く知識を扱うすべての人へ。自分の情報を自分で管理しながら、AI を長期的な仕事のパートナーにする方法を示します。

本書は、補い合う二冊の書で構成されています。**第一書——コンテキストを理解する**は、このシステムの誕生と、それが AI との関係において何を変えるのかを語ります。**第二書——コンテキストを築き、生かし続ける**は、その築き方と、長く使い続ける方法を説きます。

続けて読むことも、別々に読むこともできます。**本書を読む順序**の章に、いくつかの道筋を示しました。

事例についての注記

本書の考察と事例には、私がITのさまざまな職種で積んできた30年以上の経験が生かされています。そのもとになったのは、さまざまなプロジェクトや職場で経験した出来事に加え、他社のプロダクトマネージャー、CTO、エンジニアリングマネージャーなどとの数多くの対話です。つまり、複数の組織と、私の経歴のさまざまな時期にまたがっています。

本書で説明に用いるため、こうした状況には変更を加え、一般化し、組み合わせ直しています。ユースケース、数値、指標は、繰り返し生じる問題と、そこから私が得た学びを説明するためのものです。いずれも説明のために構成したものであり、実在する顧客、製品、組織を描いたものではありません。また、私の雇用主のデータ、慣行、見解を反映するものでもありません。私生活から取った事例——マンションの管理組合、私の記事、私自身の文章——は、実際に経験した出来事に基づいています。ただし、個人が特定されうる細部は変更してあります。

本書で述べる分析、方法、意見は私個人のものであり、私以外の誰かの立場を示すものではありません。

著者について

私はソフトウェア開発の世界で数十年働き、現在はプロダクトマネージャーを務めています。開発とアーキテクチャから始まり、チームの編成へ、そしてプロダクトとその運用、マーケティングへと歩んできました。

職種が変わるたびに、情報との関わり方も変わりました。技術的な解決策を知っていることや、文書を見つけられることだけでは足りません。散らばった兆候を集め、それらが全体として何を示しているのかを読み取り、複数の選択肢を比較して判断し、まったく異なる読者に合わせた文章を書く必要がありました。

人工知能と分散システムへの関心は、ずっと以前からのものです。それが今日、きわめて具体的な問いに出会っています。**自分の情報、自分の決定、自分の思考の主導権を手放さずに、どうすれば AI と働けるのか。**

専門誌に寄稿し、M2 Édition と Eyrolles で技術校閲を務めました。現在は malorean.net で、プロダクトマネジメント、AI、仕事の方法、情報システムについて書いています。

また、戦略・実務の両面からマーケティングを扱う独立系雑誌 *Proposition Marketing* にも寄稿しています。マーケティングや経営に携わる読者に向けて書くには、共通の専門用語を前提とせずにプロダクトの問題を説明しなければなりません。本書も同じ姿勢で書いています。

似た仕組み、異なる仕組み、あるいは補完し合う仕組みをつくっている方々と意見を交わせれば、たいへんうれしく思います。このテーマは、議論と比較を重ね、さまざまな実践から学ぶ価値があります。近い問いに取り組む研究者の方々とも交流したいと考えています。私は CNRS の二つの研究所で数年を過ごしました。その世界と再びつながれれば幸いです。

ローラン・デゼシャリエ

- laurent@malorean.net
- <https://malorean.net>

謝辞

まず、あなたに感謝します。

この本を開くという、少しばかり大胆なことをしてくださったあなたに。予想より少し厚く、おそらく繰り返しもあり、取っつきにくいテーマを扱った本です。

これまで私は、ほとんど一人でコンテキストに取り組んできました。けれども、それが今後も続くとは思っていません。このテーマに興味を持ち、探究し、議論し、批判し、発展させたいと思う人が現れることを願っています。そうなれば、malorean.netで今後公開する記事のなかで、多くの協力者に感謝を伝えられるかもしれません。

なによりも、妻のクリスティーヌに感謝します。

彼女は、この本の前で夜と週末が延びていくのを見てきました。執筆のあいだずっと支えてくれて、同時に、もう遅い、そろそろこの忌々しいパソコンを閉じる時間だと、きちんと思い出させてくれました。

約束します、これはほんの寄り道でした。

そして本書のことに限らず、長い年月をともに歩み、二人でこのすばらしい人生を分かち合ってくれていることに感謝します。

本書を読む順序

本書は、私が経験した順に話を進めます。実際の問題、あり合わせの解決策、そして最後にひとつのシステム。したがって、コンテキストの全体像が示されるのは第二書に入ってからです。

「理解する」道筋が、この順序そのものです。初めから終わりまで読めば、システムの各部品は、それを生んだ欠落が示された直後に現れます。

「築く」道筋は、すでに問題を味わっていて、物語ではなく仕組みを求める人に向いています。

1. **コンテキストの定義**。何の話なのかを知るための数行。
2. **第5部——コンテキストの内側**を、六つの章の順に。地図を読む、開く、情報を入れる、変換する、生み出して決める、保存する。
3. **第6部——コンテキストとその生態系**。いま築いたものを、あなたの情報システムの残りにつなぐために。
4. **第一書**。設計上の選択が恣意的に思えたときに。その選択を引き起こした問題がそこにあります。

この近道には代償があります。第二書は、第一書で扱った状況をすでに理解している読者を想定して書かれており、そこへの言及もあります。そのため、背景が少し足りないと感じる事例もあるでしょう。それが代償です。それでも、自分が何を求めているのかをすでにわかっているなら、十分に見合います。

読む前に試す

本書はひとつの手法を述べたものですが、手法は言葉で伝えるに
くいものです。理論を読む前に、あるいは読みながら、実際に
動いているところを見ることができます。ある部品が何のため
にあるのかは、そのときにいちばんよく分かります。

三つのリポジトリが github.com/laurent-malorean で公開されて
います。

Contexte Nano Notion は、もっとも早く始められます。開発
環境も技術的な知識も要りません。ChatGPT と Notion があれば
足ります。ChatGPT がコンテキストの知能を担い、Notion がその記
憶になります。ソース、ノート、決定、作業状況が書き込まれ、
会話をまたいで読み返されるのがそこです。数分で、支えにな
るコンテキストが手に入ります。

Contexte V2 は完全な実装で、私が毎日使っているもの、そし
て本書を書くために使ったものです。Claude Code でも Codex
でも導入でき、どちらを選んでもかまいませんが、前のものよ
り少しだけ準備が必要です。

LLM Wiki はコンテキストではありません。書籍、規格、記事、講
座などの文書から築かれる知識の図書館です。AI がそこから短
く、互いに結びついたノートを取り出し、あとから普通の言葉
で問い合わせられます。コンテキストはひとつの使命の筋道を保ち、
LLM Wiki は読んだものを大量に保ちます。二つがどう協働する
のかは、パート 6 で述べます。

そこから始めなければならない決まりはありません。けれども読んでいて抽象的に感じた瞬間があれば、具体に戻るいちばんの近道です。

同じ著者の本



Malorean Corp — 少し前までは、一つのアイデアを試作品にするだけでも、何週間もかかることがあった。今では人工知能に助けられた一人の人間が、わずか数日で実現できる。この変化は驚くべきものだ。しかし、本当の大変革は別のところにあるのかもしれない。私たちが一緒に働く、その方法である。

刊行予定：2026 年 11 月末

詳しくは <https://malorean.net>.

目次

書 I

コンテキストを理解する

部 1 — 必要性の芽生え	21
1.1 五百通のメールと、消えていく経緯	22
1.2 最初の定義	31
部 2 — コンテキストが変わるもの	39
2.1 会話から、再開できる仕事へ	40
2.2 二人の顧客、一冊のホワイトペーパー、そして保つべき矛盾	47
部 3 — 三つの具体的な使い方	57
3.1 済ませた仕事を失わずに提案依頼書へ応える	58
3.2 まだ形をなさない考えから記事を書く	64
3.3 混乱のなかでプロダクトのテーマを育てる	72
部 4 — 成果物があとに残していくもの	81
4.1 答えはセルに収まるが、筋道は収まらない	82
4.2 六か月後、その答えではもう足りない	86

書 II

コンテキストを築き、生かし続ける

部 5 — コンテキストの内側	93
5.1 全体の地図	94
5.2 開く——使命と現状を取り戻す	105
5.3 与える——情報を取り込む	115
5.4 変える——ソースからアイデアへ	121
5.5 生み出して決める	130
5.6 保存して再開する	139
部 6 — コンテキストとその生態系	147
6.1 構造化された知識に問いかける	148
6.2 実際の製品に問いかける	155
6.3 コンテキストが三つになった日	163
6.4 隣のコンテキストを参照する——情報は移さない	169
6.5 行き来が払わせるもの	176

読み終える前に

結び	183
7.1 あとがき	184
7.2 系譜と隣接領域	187

書 I

コンテキストを理解する

- 1 必要性の芽生え
- 2 コンテキストが変わるもの
- 3 三つの具体的な使い方
- 4 成果物があとに残していくもの

部 1

必要性の芽生え

コンテキストは、人工知能についての思いつきから生まれたものではありません。手に負えなくなったメールボックスから生まれました。そして、時間をかけてたどり着いた、ひとつの気づきからです。問題はメールの数ではありませんでした。

この部では、その出発点をたどり、そこから最初の定義を導き出します。覚えられるほど単純でありながら、この先を読むための道しるべになるだけの内容を備えた定義です。

五百通のメールと、消えていく経緯

はじめてコンテキストをつくったとき、私は人工知能との仕事の方法を発明しようとしていたわけではありません。ただ、自分のマンション管理組合で何が起きているのかを、見失わずにいたっただけです。

私は理事会に席を置いています。その立場で、管理会社、区分所有者、賃借人、業者からのメールを受け取ります。

- あるメールは新しい問題を知らせ、
- 次のメールは見積もりの詳細を伝え、
- 三通目は工事の延期を告げ、
- 一週間後には、同じスレッドへの返信で別の棟の話が始まる。

その間に、ほかの三つの案件も進んでいます。

問題は量だけではありませんでした。五百通ほどのメールが、十五ほどの案件に散らばっていたのです。

- 工事、
- 財務、
- 保険、
- 総会、
- 滞納者、
- 入札、
- 事故や各種の依頼
-

それぞれが独自の速さで、ときに何か月にもわたって進んでいきます。

現場を日々追いかけている人は、自然とその経緯を頭に入れています。どの業者が入ったのか、なぜ日程が変わったのか、どの決定がまだ返事を待っているのかを知っています。新しいメールを開くとき、私にはいつもその連続性があるわけではありませんでした。新しい情報を理解する以前に、話をさかのぼり、前のやり取りを探し出し、状況を組み立て直さなければならなかったのです。

私は単純な質問ができるようになりたかったのです。

- 屋根の件は今どうなっている？
- 雨樋の最新の見積もりを出したのはどの業者？
- あの浸水について、私たちは何を決めた？
- 管理会社からの対応で、まだ待っているものは？

答えはメールボックスの中にありました。けれども、全体像は与えてくれませんでした。

一つのファイルと一つの文

私は考えうるかぎり簡素な仕掛けから始めました。ファイルが一つと、AI への指示が一つです。

｜ 情報をまとめて、テーマごとに切り分けてください。

私はメールを渡していきました。AI はそれらを案件ごとに整理し、それぞれの要約を更新しました。

- 工事についての情報が一か所にまとまりました。
- 財務の問題は別のまとまりをつくりました。

- 業者、保険、総会も、少しずつ居場所を見つけていきました。

この最初の試みだけでも、すでに得るものがありました。関係するメールをすべて読み返さずに、案件の進捗を尋ねられるようになったのです。AI はもはやメールボックスの中の一致を探すだけではありません。状況について理解したことを示してくれました。

新しい情報が届くたびに、ファイルの構成も変わっていきました。

- 新しい案件が重みを増すと、見出しがひとつ現れました。
- 一つの見出しに別々の事柄が混ざり始めると、二つに分けました。

情報は、ただ積み上がるだけのものではなくなりました。仕事に使える形に整理されていったのです。

やがてこの最初のシステムは、正確でありながら足りない答えを返しました。

工事は終わった。けれどその経緯は消えていた

工事が終わると、要約には「工事完了、検収済み」とだけ記されました。

たしかにそれが現在の状態でした。しかし、その更新によって失われたものもありました。

- 工事はいつ始まったのか。
- どの時点で遅れたのか。
- なぜ日程が変わったのか。
- 変更契約はいつ結ばれたのか。

これらの段階は、どれも意味を持っていました。最後の更新が、最終状態のためにそれらを消し去っていたのです。

私は要約を求めました。AI は古い状態を新しい状態に置き換えることで、それを最新に保っていました。指示には従っていました。けれども、その指示は経緯を守るものではなかったのです。

そこで、同じ案件について二つの見方が必要になりました。

- ひとつめは「今どこにいるのか」に素早く答えるもの。
- ふたつめは出来事を順に保つもの。工事の開始、遅れ、新しい合意、作業、検収。
- 現状がわかれば、次の行動を決められます。
- 経緯がわかれば、なぜその状態に至ったのかを理解し、疑問が生じたときに確認できます。

この区別は、きわめて具体的な欠陥から生まれました。私はそれを片づけ方の改善だと思っていました。それが、AI との仕事におけるもっとも重要な原則のひとつになったのです。**いま真であることと、起きたことは、同じ役目を果たさない。**

AI には私の決まりごとを察することができなかった
別の問題が、名前をめぐって現れました。

- 二人の業者が同じ姓を名乗っていたのです。
 - ・ 見分けるには、メールの内容を理解する必要がありました。
- ある区分所有者は、あるときは旧姓で、あるときは夫の姓で現れました。

とくに指示がなければ、AI は一人しかいないところに二人をつくりかねません。

ほかの違いは、仕事のやり方そのものに根ざしていました。

- 百ユーロ未満の滞納者は、ほかの滞納者と同じようには追いません。
- この線引きは、AI が知りうるような普遍的な規則から来たものではありません。
- 私たちの働き方から来たものです。

私はこうした知識を、今後も使うルールとして、ファイルにはっきり書き始めました。

処理のルール

- 屋根について言及されるMartinは屋根職人を指す。緑地について言及されるMartinは造園業者を指す。
- Dupont-LemoineとLemoineは同一の区分所有者、14号室を指す。
- 百ユーロ未満の滞納者は別のリストに示すこと。

これ以上に凝ったものは何もありません。会話のたびに言い直すはずだった三つの文。そして、二度と繰り返さずにすんだ三つの文です。

どのルールも、実際に起きた誤りや曖昧さから生まれました。私の頭の中にしかなく、AI が確実には推測できない知識を、明文化したものです。

こうしてシステムは、性質の異なる二種類の情報を保つようになりました。

- マンション管理組合についての情報。

- それをどう扱ってほしいかという私の意向。

助けてくれたファイルが、邪魔になり始めた

週を重ねるうちに、ファイルは大きくなりました。五百行を超え、やがて八百行、そして千行になりました。

この成長は、その有用さの証でした。

- 案件はよりよく追えるようになり、
- 業者は見分けられ、
- 決定は見つけられるようになりました。

けれども、システムを成り立たせていたものが、今度は使いにくさの原因になり始めました。

- 種類の異なる情報が、ひとつの場所に詰め込まれていました。
- 規則と事実が混ざり合っていました。
- 財務が、工事や人や保険や入札と隣り合っていました。

必要な情報を見つけ出すのに、ふたたび手間がかかるようになりました。

まず私は、処理の規則を分けました。それらはマンション管理組合を記述するものではなく、それに関する情報とどう付き合うかを示すものだったからです。切り離すことで、見つけやすくなり、適用しやすくなりました。

次に、情報をいくつかのテーマに分けました。工事、財務、業者、総会の決議、保険に、それぞれ専用の場所を設けました。全体を見渡す一覧には、それぞれの場所に何が入っているかを示しました。すると AI は、毎回すべての情報を調べる代わりに、質問に関係するテーマだけを開けるようになりました。

最後に、独特の語彙をまとめました。法律用語、略号、マンション管理に固有の言い回しです。百科事典をつくるのが目的ではありません。やり取りの理解がそこにかかっている言葉に、安定した意味を与えるのが目的でした。

それでもなお、私の頭に大きな設計図はありませんでした。問題が起きるたびに、それを解決するための仕組みをひとつずつ足していただけです。

出どころを失わずに信じる

AI はときどき誤りを犯しました。厳密には同じ話ではない二つの情報を結びつけたり、同姓の二人のうち間違った業者を選んだりすることがありました。ですからシステムは、理解を速めながらも、事実の唯一の版になってはならなかったのです。

私は元のメールをすべて残しました。回答を確認する必要があるときや、重要度が高いときには、受け取った元の情報に戻ることができました。要約は素早い入り口を与えます。出どころは検証を可能にします。

この選択が、使用を成り立たせていました。メールの整理に多少の不正確さがあっても受け入れられたのは、元の情報をいつでも確認できたからです。同じ水準の信頼は、銀行の書類にも、法令上の義務にも、取り返しのつかない決定にも向きません。危険はつねに、システムに何を委ねるのか、そして検証できるかどうかによって決まります。

ですからコンテキストは、私のメールボックスの代わりではありませんでした。その中身を作業用の記憶に変えながら、元のメールへ戻る扉を残していたのです。

現れていたもの

数週間のうちに、出発点だった「一つのファイルと一つの文」は、いくつもの要望に応えられる仕組みになっていました。現状を示し、経緯を残し、マンション管理に固有のルールを適用し、案件を区別し、用語の意味を統一し、元の情報に戻る仕組みです。

新しいメールが届いたとき、私はそれ以前のすべてを組み立て直す代わりに、それが何を変えるのかに目を向けられるようになりました。問いが浮かんだとき、適切なスレッドの中から適切なメールを探す代わりに、問題の言葉でそれを立てられるようになりました。AI が間違えたとき、規則を直すか、元の文に戻ることができました。

私はコンテキストをひとかたまりで設計したわけではありません。足りない点をひとつずつ直すなかで、形になっていったのです。要約は経緯を消し、名前は混同され、ルールは埋もれ、ファイルは大きくなりすぎました。修正を重ねるたびに、作業の内容が失われにくくなり、中断後も再開しやすくなりました。

この出発点には意味があります。コンテキストは、もっと分類したいという願いから生まれたものではありません。変わり続ける情報を理解し、それを検証するための元情報を残し、ゼロからやり直さずに仕事を続けたいという必要から生まれたのです。

要点——同じ案件に対する二つの見方

状態は「今どこにいるのか」に答える。経緯は「どうやってここに至ったのか」に答える。前の要約を上書きして最新に保たれた要約は、前者を保ち、後者を失う。どちらも必要である。ひとつは行動するため、もうひとつは理解し、確認するためだ。

最初の定義

あの五百通のメールから生まれたシステムに、ここで最初の定義を与られます。

コンテキストとは、ひとつのテーマに特化した作業空間である。そこでは人間と AI が、長期にわたって理解し、判断し、成果を生み出すために必要な情報を保ち続ける。

この定義は、あえて単純なままにしています。本書で扱うコンテキストには明確な構造があり、後半で詳しく説明します。それが何をもたらすのかを理解するには、まず六つの要素を押さえれば十分です。

- ひとつの目的、
- 元となる情報、
- アイデア、
- 作業状況、
- その仕事に固有のルール、
- 生み出すべき成果物。

ひとつのテーマと、ひとつの目的

マンション管理組合のコンテキストは、私が知っていることすべてを収めようとはしていませんでした。目的はひとつでした。進行中の案件を理解し、その経緯を残し、理事会で行動するために必要な情報を見つけることです。

この目的が境界になっていました。屋根についてのメールには居場所がありました。管理組合と関係のない面白い読み物には、ありませんでした。この線引きによって、システムが「興味深い」というだけの情報を何でもため込む場所になるのを防げました。

ですからコンテキストは、ひとつの意図から始まります。それは次のようなことに役立ちます。

- 本を書く、
- プロダクトの問題を探る、
- 決定を準備する、
- 提案依頼書に応える、
- 業務上のひとつの対象を追う、
-

出どころを保った情報

情報は二つの入り口からコンテキストに入ります。

ひとつめの入り口は、作業を始める前から存在しているものを受け取ります。

- メール、
- 記事、
- 本、
- ウェブページ、
- 面談の書き起こし
-

これらが**文書ソース**です。すでに完成した文書として持ち込まれ、内容は変わりません。管理組合の場合、工事の延期を告げるメールがこの第一の種類にあたります。

ふたつめの入り口は、AI との作業中に言葉になったものを受け取ります。私が状況を説明し、解釈を正し、ニュアンスを明確にし、経験から得た確信を述べると、その言葉は**セッションソース**になります。やり取りのなかで少しずつ形づくられていくものです。二つの名前が同じ区分所有者を指すと説明した場合、その言葉はまず二つめの入り口から入ります。

違いは、情報がいつ、どのように現れるかにあります。

- 資料は作業の前から存在し、そこへ持ち込まれます。
- セッションソースは作業のあいだに生まれ、それとともに育ちます。

その後、この二つのソースは同じ役割を果たします。解釈される前の元情報を残すことです。

- そこに含まれるすべてが真であることを保証はしません。
- 受け取られた、あるいは口にされたものを見つけ出し、その確かさを見極めることを可能にします。

コンテキストは、この元情報からアイデアやルール、作業状況を引き出せます。こうして整理された知識は、元情報とは別の場所に置かれます。日付が間違っているように見えるとき、二人の人物が混同されたとき、ある解釈が疑わしくなったとき、こうしてその出どころに戻ることができます。

使える形になったアイデア

ソースを積み上げるだけでは足りません。きちんと保たれた五百通のメールは、読み返すべき五百通のメールのままです。

ですからコンテキストは、この元情報を少しずつ整理し、使える形に変えていきます。そこから現れるのは次のようなものです。

- テーマ、
- 重要な事実、
- 規則、
- 区別、
- 未解決の問い、
- 関係。

孤立したやり取りは、それが属する工事の経緯に合流します。同じ業者への複数の言及は、共通の知識を豊かにします。ひとつの訂正が、混同の再発を防ぎます。

この中間段階が重要なのは、情報を実際に使えるようにするからです。

- 受け取った元情報には記録が残ります。
- 整理された知識があれば、理解し、行動できます。

より複雑な仕事では、同じアイデアが何度も役立つことがあります。同じプロダクト知識を、意思決定にも、仕様書にも、ドキュメントにも活用できます。検証済みの同じ事実が、複数の顧客アンケートへの回答に役立つこともあります。成果物が変わっても、アイデアとその出どころは残ります。

作業を続けるための状況

生きている仕事にはつねに現在があります。わかっていること、まだ不確かなこと、決まったこと、そして次に起きるべきことです。

管理組合では、現状が目の前の問いに答えていました。この工事は今日どこまで進んでいるのか。その経緯は別の問いに答えていました。どうやってここに至ったのか。作業をきちんと再開するには、システムに両方が必要でした。

この区別は、ひとつの工事を追うことを超えています。文章の執筆やプロダクト上の問題の検討を中断したあと、私が取り戻したいのは単なる資料の山ではありません。私が知りたいのは次のことです。

- 何が確かになったのか、
- どの点がまだ脆いのか、
- 次に何をすれば作業を再開できるのか。

コンテキストは、この連続性を会話の外に保ちます。

- AI とのセッションは終わってもかまいません。
- 作業状況は残り、次のセッションでそこから再開できます。

その仕事に固有の規則

AI は一般的な知識を備えています。その一方で、ある状況を特別なものに行っているものの多くを知りません。

- 同じ姓を名乗る二人の業者、
- あるチームが顧客からの兆候をどう分類するか、
- ある主張を公にする前に求められる根拠の水準、

- ある本のために選ばれた調子、
-

1 コンテキストは、こうしたルールを、それが適用される仕事とともに保ちます。経験から限界が見えたときは、ルールを修正できます。そのあとは、人間が依頼のたびに言い直さなくても当てはまります。

これらの規則は、仕事の責任を AI に渡すものではありません。人間が AI に何を期待しているのかを、はっきりさせるものです。人間が目的を定め、情報を持ち込むか検証し、解釈を正し、決定を下します。AI はこの枠組みをもとに、吸収し、照らし合わせ、整理し、問い、言い換え、成果を準備します。

必要に合わせた成果

コンテキストはつねに、それが可能にすべきことへ向いています。管理組合では、成果は次のようなものでした。

- 問いへの答え、
- 工事の現況、
- 話し合いの準備。

ほかの用途では、こうなるでしょう。

- 記事、
- 仕様書、
- 文書。

同じ作業空間から複数の成果物を生み出せるのは、最終的な形に整える前の情報を残しているからです。ソースは、それを最初

に使った文書に閉じ込められません。アイデアは、別の問いや別の読者にも使える形で残ります。

この点が、コンテキストと単なる保管庫との違いです。

- 保管庫は情報を保存します。
- コンテキストは、保存した情報を再び取り上げ、別の形に変えるための仕組みを整えます。

きわめて単純に始められる形

私の実践では、コンテキストはファイルの姿をとります。読むことができ、動かすことができ、それを扱う道具とは無関係に保てるファイルです。この形なら、自分で管理でき、長く使い続けられます。同じ働きを守るのであれば、ほかの実装もありえます。

とはいえ、最初のコンテキストに凝ったところは何もありませんでした。ファイル一つに収まっていた。その限界が目に見えるようになったときに、大きくなったのです。この段階的な歩みは重要です。本書で述べるシステムを最初からすべて築かなくても、考え方を試すことはできます。

ひとつのテーマ、ひとつの目的、そして最初の情報があれば十分です。

- AI がそれらの整理を助けます。
- 人間は欠けているものを見つけ、規則を足し、ソースを保ち、役割の違う二つの情報を分けます。
- 構造は、問題にぶつかるたびに少しずつ形になります。

ですからコンテキストが指すのは、特定のフォルダー構成というより、人間と AI、そして目的に沿って集めた情報のあいだに築かれる、継続的な仕事の関係です。

次の部では、この関係が何を変えるのかを見ていきます。AI との会話は、それだけでも実り多いものになりえます。コンテキストへの移行は、その対話が、目的、経緯、作業状況をセッション終了後も保てる環境に置かれたときに始まります。

要点——コンテキストを構成する六つの要素

範囲を決める目的、出どころを保った元情報、使える形にしたアイデア、作業を続けるための状況、その仕事に固有のルール、生み出すべき成果物。始めるには、この六つで十分である。構造はその後、直面した問題に応じて整えていけばよい。

部 2

コンテキストが変えるもの

AI との会話だけでも、多くの成果を得られます。AI は説明し、言い換え、提案し、ときには一人では思いつかなかったような答えを返してくれます。

ですからこの部は、コンテキストがそこに何を加えるのかを探ります。答えは、回答そのものの質ではなく、セッションが変わっても何が残るかにあります。中断したところから再開できる仕事と、食い違いを消さずに残せる情報です。

会話から、再開できる仕事へ

AI との会話は、ごく簡単に始められます。

まだ漠然とした考えを渡します。AI がそれを言い換えます。構成案を求め、もうひとつ求めます。思いもしなかった反論を返してくれます。意図をはっきりさせ、ある筋を退け、別の筋を掘り下げます。数分のうちに、白紙を前に長く止まっていたかもしれないものが前へ進みます。

これがチャットの大きな強みのひとつです。始めるには問いがひとつあれば足ります。この形は短い作業にととてもよく合います。要約や説明やいくつかの選択肢を求める。答えが届き、それを使い、会話は終わってかまいません。

関係が変わるのは、仕事が長く続くときです。

書き始める前の五つの文

先週始めた文章を再開しようと、六日前の会話を開き直します。一行書く前に、私はこう打ち込みます。

報告書の品質についての文章を再開してください。読者は開発者ではなく、運用責任者です。フォームを短くする案は退けました。結論を出すにはまだ早すぎます。監査についての数字は一件の面談から来ています。仮説として扱ってください。三つめの節まで進んでいました。

六日前にはすでに目の前にあった仕事に、AI を戻すための五つの文です。

いま打ち込んだものはすべて、打ち込む前から存在していました。切り口も、読者も、決定も、不確かな点も、原稿の進捗も、前の会話の中にありました。私はそれを自分の記憶から取り出ただけです。会話は、それらを語られた順に並べていたからです。

一方の仕事は、別の並び方に従います。

- 最初に与えた指示は、終わりでもなお当てはまらなければなりません。
- 昨日加えた資料は、明日書く段落を支えられなければなりません。
- 過去の決定は、同じ議論の蒸し返しを防がなければなりません。
- 保留中の点は、しかるべきときに現れ直さなければなりません。

会話が長くなるほど、人間はこの並び方を頭の中で保たなければなりません。新しいセッションを始めるたびに、AI のために状況を説明し直します。目的はこれ、変更点はこれ、文体はこれ、決定事項はこれ、作業はここまで進んでいた、と。

チャットは役に立ち続けます。けれども連続性の負担は、それを使う者にのしかかります。

残すべきものに場所を与える

コンテキストは、会話とは別の空間を仕事に与えることで、この関係を変えます。

話し合いは、やり取りの場であり続けます。問い、探り、はっきりさせ、直すことを可能にします。残すべきものは作業空間に移します。

- 目的、
- 資料、
- 規則、
- 固まったアイデア、
- 決定、
- 文章、
- 進み具合。

この違いは小さく見えるかもしれませんが。会議と、その会議が語るプロジェクトとを隔てる違いに似ています。良い会議はプロジェクトを前進させますが、プロジェクトのすべてが会議の記憶に収まるわけではありません。資料も決定も行動も、会議が終わったあとも存在し続けます。

チャットの全文を保存しても、同じ効果は生まれません。何が語られたかを正確に見つけ出したいときには貴重ですが、それは会話の形に従っています。問い、答え、訂正、脱線、本題への復帰。仕事を再開するには、なおこの連なりを読み返し、いま何が大事なのかを決めなければなりません。

一方のコンテキストは、語られたことを変えます。

- 訂正は、誤りの再発を防げる場所へ移ります。

- 決定は、決定として見えるまま残ります。
- 資料は、それが支えるものに結びついたままです。
- 文章は、最新の完全な版として存在します。

こうして記憶は行動へ向きます。それは過去を保つことだけを求めません。次の一手を準備するのです。

三十二の文章を見つけ直すための一ページ

本書の仕事を再開するなかで、私はそれが何を変えるのかを測りました。

当時、原稿は三十二の文章からなっていました。書けている部分もあれば、まだ空の部分もありました。ある節は、不釣り合いな場所を占めていたために分割されていました。二つの箇所が私の読み直しを待っていました。ひとつの主張はどの実体験にも支えられておらず、書き直す必要がありました。

一つひとつを取れば、どれも理解しにくいものではありません。けれども全体として見ると、役に立つ一行を書く前に、状況を整理し直さなければなりません。

三十二の文章を開き、構成案を読み返し、昔の会話をたどり、保留にした指摘を探すこともできたでしょう。この発掘作業には時間がかかり、しかも不安が残ります。誤りを防ぐために残しておいた注意点を、見落としてはいないだろうか。

その代わりに、私は本のコンテキストを開きます。そこには再開のためのファイルが差し出されます。その朝、それはおおよそこう告げていました。

再開

今どこにいるか

構成案から結論まで、三十二の文章が存在する。

技術的な部分は書けている。最初の二つの部はまだ理論に寄りすぎている。

続ける前に知っておくべきこと

想定する読者はこのテーマに初めて触れる。実体験より先に概念が来ると足を取られる。

文書管理システムの失敗についての主張は、まだどの個人的経験にも支えられていない。

次の一手

管理組合の五百通のメールの事例を軸に、冒頭を書き直す。

未解決の点

二つの箇所が読み直しを待っている。

いくつかの資料が、ノートに変えられないまま入っている。

一ページ、読むのに一分もかかりません。何をすべきかがわかり、三十二の文章はひとつも開いていません。

続けることと、再開することは違う

続けるとは、次の一手を知っていることです。再開するとは、その一手を打つ前に仕事の意味を取り戻すことです。

やることの一覧は、ある資料を処理すべきだと教えてくれます。けれども、なぜその資料が優先なのか、本のどの部分を照らすはずなのか、どんな危険を正すためのものなのかは説明しません。

再開するために、私にはもっと豊かなひとまとまりが必要です。

- 仕事が追っている目的。
- それが実際に到達した地点。
- いまなお先を方向づけている決定。
- 忘れてはならない不確かな点。

- やり直すのに十分なだけ具体的な、次の一手。

再開のファイルは、まさにこの順に読めます。「三十二の文章」は尺度を与えます。「実体験より先に概念が来ると足を取られる」は向きを与えます。次の一手はその二つから導かれ、本を開くべき具体的な事例を名指します。

再開は、一覧の中の位置だけでなく、ひとつの筋道を取り戻したのです。

とはいえ、前の時間の細部を頭の中にすべて再現しようとするわけではありません。短すぎる要約は、選択の理由を消します。長すぎる報告は、仕事が再び始まる時を遠ざけます。

中断は時を選ばない

はじめのうち、私は、作業をきれいに終われば次回も簡単に再開できると考えていました。セッションの終わりに作業状況を丁寧に要約し、次の一手を書き留めればよい、と。

実際はめったにこの筋書きどおりになりません。

- 会議が始まります。
- 電池が切れます。
- 一日が思ったより早く終わります。
- 別の緊急案件に追われます。

再開のための記憶がもっとも役に立つのは、まさに終わりを用意できなかったときです。

ですからコンテキストは、再開する力を仕事のあいだに保ち続けます。決定は下されたときに、作業は言葉になったときに、資料は入ってきたときに、それぞれの場所を見つけます。

再開は、使う者の英雄的な規律ではなく、システムの性質になります。

それでも人間には決定的な役割が残ります。提案された次の一手がもう優先ではないと判断したり、古い決定を開き直すべきだと判断したり、新しい事実が目的を変えると判断したりできます。記憶は道筋を凍りつかせません。それを変えると決めるときに、よりどころとなる道筋を見えるようにするのです。

会話を開き直すたびに打ち直していたあの五つの文を、私はもう打ち直しません。それらはどこかに書かれていて、私以外の誰かが、毎回の始まりに私の代わりに読んでくれます。

要点——チャット、環境、そして再開

チャットはやり取りの場であり、コンテキストは残すべきものの場である。会話は情報を語られた順に並べる。仕事は別の並びを求める。続けるとは次の一手を知ること。再開とは、それを打つ前に仕事の意味を取り戻すこと。中断は予告しない。再開する力は、仕事を閉じる瞬間ではなく、仕事のあいだに保たれる。

二人の顧客、一冊のホワイトペーパー、そして保つべき矛盾

ここまで私は、コンテキストと、それが生むいくつもの効果を述べてきました。では、実際に開いてみたとき、それはどんな姿をしているのでしょうか。ファイルの中には何が見えるのでしょうか。複数のソースは、別々に見れば正しくても両立させにくいアイデアを、どのように浮かび上がらせるのでしょうか。

この架空のコンテキストのテーマは単純です。一部の技術者がなぜ不完全な作業報告書を書くのかを理解し、そのうえで製品を変えるべきかどうかを決めること。

その使命は数行に収まります。

作業報告書の品質を損なっているものを理解する。入力の上
づらさ、組織上の制約、データ活用の必要を区別する。複数の
ソースを突き合わせる前に解決策を選ばない。

この使命が仕事に向きを与えます。求めているのは問題の理解であって、機能の設計ではまだありません。

最初の声が、ひとつのアイデアを浮かび上がらせる

コンテキストに入れる最初の資料は、運用責任者との面談の書き起こしです。

彼女はこう説明します。

「技術者はたいてい金曜日に報告書を書きます。現場では、フォームが長すぎると感じられるのです。週の終わりには、細部の一部を忘れてしまっています。」

AI はこの面談を**ソース**として保ちます。そこからアイデアをひとつ取り出し、別のファイルに記録します。私が**アトミックノート**と呼ぶものです。

入力負担が大きすぎると、一部の技術者は報告を先送りし、細部を失うことがある。

この言い方は、すべての技術者がそう振る舞うとも、フォームの長さが唯一の原因だとも述べていません。証言の中心を保っています。語られた組織では、入力先送りがされ、作業とその記述のあいだで情報の一部が消える、ということです。

ソースは手元に残ります。ノートはそれに取って代わりません。その主要なアイデアを、見つけやすく、あとで突き合わせやすくするだけです。

二つめの声が、同じノートを豊かにする

数週間後、二つめの面談がコンテキストに入ります。

今度は、運用責任者がこう語ります。

「データの品質を上げるために、十二の項目を必須にしました。今では、一部の技術者が検証を通すためだけに『異常なし』と書いたり、ハイフンを入れたりしています。」

この証言は、厳密には同じ振る舞いを述べていません。最初の面談の技術者は入力を先送りします。二つめの技術者はフォームを埋めますが、その要求をすり抜けます。

それでも二つのソースは、同じ中心的なアイデアをめぐっています。より多くの情報を得るための制約が、結局は入力されるものの実際の品質を下げうる、ということです。

ですから二つめの面談は、最初の**アトミックノート**に合流します。最初のソースに取って代わるのでもなく、競合するアイデアをつくるのでもありません。同じ問題の別の形を浮かび上がらせることで、ノートを豊かにします。先送りされた入力のあとに、見せかけの完全さが来るのです。

ノートはいまこう言えます。

入力の負担が大きすぎると、一部の技術者は報告を先送りしたり、すり抜けたりすることがあり、集まる情報が損なわれる。

二つのソースが参照されています。それぞれが自分の役割と限界を保っています。二つの証言で現象の広がりを読めると言い張ることなく、ノートはより堅くなります。

ホワイトペーパーが、別のアイデアを浮かび上がらせる

次に、作業データの活用についてのホワイトペーパーを加えます。

この資料は別の要請を掲げています。繰り返す故障を理解し、設備を比べ、ある種の監査に備えるには、企業には十分に構造化された報告書が必要です。技術者それぞれが自由記述欄に好

きなことを書く——あるいは何も書かない——なら、入力は楽になるかもしれませんが、データは集団としての有用性の多くを失います。

このソースは、先の証言の真実性を損なうものではありません。述べられた技術者たちは、変わらず入力を先送りし、すり抜けています。このPDFは別の側面を浮かび上がらせます。制約をすべて取り払うことにも、代償があるということです。

AIはこれらを最初のノートにまとめ、一般的な折衷案として書き直そうとするかもしれません。それは誤りです。ホワイトペーパーは面談で観察された事実には但し書きを付けているのではなく、別の命題を掲げているのです。

ですからそれは二つめのアトミックノートを生みます。

作業を比べ、作業データを活用するには、構造化された報告書が必要である。

コンテキストはいま、二つの異なるアイデアを抱えています。

- 入力の制約は、実際に集まる情報を損ないうる。
- 構造がないことは、その情報のあとでの活用を損ないうる。

二つのノートは同時に真でありえます。それでいて、製品の決定を正反対の向きへ引っ張ります。

対立はノートどうしのあいだに見えたままではなければならない
コンテキストの役目は、矛盾が消えるまで最初のノートを書き換えることではありません。二つの命題を保ち、その関係をはっきりさせることです。

そこで**テーマノート**が、両者を近づけることで見えてくるものを言葉にできます。

報告書の品質は、入力瞬間に求められる労力と、その活用に必要な構造とのあいだの緊張に左右される。

この三つめのノートは、外交的な平均ではありません。二つのアトミックノートのどちらにも取って代わりません。両者を並べて読んだときに現れる、新しい問題を示します。

矛盾は作業状況にも入ります。入力の先送りやすき抜けを招かずに、構造化された情報をどう得るのか。三つのソースのどれにも、まだ答えはありません。

画面に見えるのは、こういうものです

この例に使ったコンテキストは、小さなファイルの束として存在しています。その構造は概念そのものではありません。私が今それに与えている具体的な形にすぎません。

```
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
```

これで、本書の初めから使ってきた言葉を、目に見える対象に結びつけられます。

- **使命**は、この束がなぜ存在するのかを説明します。
- **ソース**は、解釈される前に受け取ったものを保ちます。
- **アトミックノート**は、別々のアイデアを安定させます。
- **テーマノート**は、その緊張を読み取れるようにします。
- **作業状況**は、その緊張が何を未解決のまま残しているかを告げます。
- **成果物**は、次の行動を準備します。

このどれひとつとして、謎めいたソフトウェアを必要としません。VS Code (Codex のアプリでも、Claude Cowork でもかまいません) は、ここでは私がコンテキストに近づくための窓です。

コンテキストそのものは、ファイルと、その役割と、そのあいだに保たれた関係のなかにあります。

返される言葉が、その場に根ざす

数日後、私は仕事を再開し、AI にこう持ちかけます。

「いっそ、すべての項目を任意にしてみましょう。」

切り離された会話であれば、AI はこの考えを後押しし、より軽い画面を提案し、その利点を並べ立てるかもしれません。コンテキストの中では、二つのアトミックノートと、それらを結ぶ緊張に出会います。

そこでこう答えられます。

その案は、面談で語られた負担の一部をおそらく減らすでしょう。ただし、ホワイトペーパーが挙げた構造化データの必要性とは相反します。ソースは現実にある二つの制約を示していますが、簡素化、段階的な入力、音声入力、作業の種類に応じた項目の変更のうち、どれを選ぶべきかまでは判断できません。

この応答は、プロダクトマネジメントの一般的な知識から来たものではありません。進行中の仕事のなかに根ざしています。AI は、私たちがどの問いを理解しようとしているのか、どのアイデアがどのソースに支えられているのか、どの矛盾が未解決なのかを知っています。

私の代わりに決めるのではありません。私の新しい思いつきが、すでに確かめられた現実のひとつを消してしまうのを防いでいるだけです。

最初の成果物が、次のソースを準備する

コンテキストの**作業状況**は、「新しいフォームを設計する」とは言いません。確かになったと思えるものと、不確かなままのものを分けます。

確かになったと思えること 入力 of 制約は、先送りとしり抜けを招きうる。報告書を活用するには、十分な構造が必要であり続ける。

未解決の点 どの摩擦がもっとも重いのか、どの情報が絶対に構造化されたままであるべきかは、まだわからない。

次の一手 ただちに解決策を示すのではなく、問題の原因を切り分ける面談を準備する。

ですから最初の成果物はささやかです。新しい面談のための質問の手引きです。入力の瞬間、現場の制約、各項目の実際の有用さ、そしてすり抜けの形を理解できるようにするものです。この成果物は矛盾に結論を出しません。将来その判断を可能にするかもしれない情報を準備します。

仕事の対象であって、何でも融かす機械ではない

この例は、AI が犯しうる誤りも示しています。テーマが似ているからという理由で、新しいソースを既存のノートに入れ込もうとすることです。

複数のソースが同じ命題をめぐるとき、**アトミックノート**は大きくなれます。二つの面談で起きたのがそれです。けれどもホワイトペーパーは、その命題の新しい証拠ではありません。自分のノートに値する、別のアイデアを浮かび上がらせているのです。

この区別には意味があります。対立するたびに前のノートを書き換えていけば、「アイデアの世界」はやがて緊張をならしてしまいます。知識は筋が通って見えますが、ソースが実際にもたらしたものとはもう対応していません。

コンテキストは、AI がつねに正しい境界を選ぶことを保証しません。人間は融合を拒み、新しいノートを求め、両者のあいだに作られた関係を確認められなければなりません。構造はこの訂正を可能にし、長持ちさせます。自動的に決めてくれるわけではありません。

これであなたは対象を見たことになります。

- 手つかずのまま残る**ソース**、
- それぞれの中心を保つ**アトミックノート**、
- 融かさずに近づける**テーマノート**、
- 未解決の**矛盾**、そして次の行動を準備する**成果物**。

続く章ではコンテキストの使い方に戻り、その後、ここで見えるようになった要素を手がかりに、その構造をさらに詳しく説明します。

要点——矛盾は残しておく

二つのソースは同時に真でありながら、決定を正反対の向きへ引っ張りうる。アトミックノートは、ソースが同じ命題を補強するとき
に大きくなる。別の命題は自分のノートに値する。緊張を消すために融合すれば、筋は通るがソースと対応しない知識ができあがる。

部 3

三つの具体的な使い方

前の二つの部では、コンテキストが何を変えるのかを述べました。この部では、それが働くところを示します。私が実際に行っている三つの使い方の上で。提案依頼書に回答すること、まだ形になっていない考えから記事を書くこと、細切れに中断される日々のなかでプロダクトのテーマを育てること。期間も、材料も、読者も異なる三つの仕事です。それでも流れは同じです。すでに理解したことを見つけ、それを別の形に変え、元の情報を失わずに成果を生み出します。

済ませた仕事を失わずに提案依頼書へ応える

このシステムで最初に扱った提案依頼書には、四十三の設問がありました。

製品のアーキテクチャ、セキュリティ、バックアップ、サービスの継続性、データの保護について答える必要がありました。情報の一部は私たちの技術文書の中にありました。ほかは設定ファイルの奥に埋もれていました。いくつかについては、サービスを運用しているチームに尋ねなければなりませんでした。私たちは七つの技術資料をかき集め、チームと一往復やり取りする必要がありました。この作業は、書くこと自体に上乘せされました。見込み客が何を求めているのかを理解し、情報を探し出し、それがまだ正しいかを確認、何を伝えてよいかを決め、そのうえで求められた形式に答えを収め直す。

これは例外的な状況ではありませんでした。新しい見込み客はそれぞれ自分の文書を送ってきます。あるときは表計算、あるときは Word のファイルか PDF。参照する基準も言い回しも変わりますが、同じテーマが戻ってきます。

それでも私たちは、毎回やり直している感覚を持っていました。回答には三時間から六時間かかり、何人もの協力が必要ですが、その見込み客が顧客になる保証はありません。そして文書は去り、仕事は送ったファイルの中に閉じ込められたまま残りました。

ですから私の最初の目標は単純でした。次の質問票が届いたとき、もうゼロから始めたくない、ということです。

もっともらしく、しかし誤った答え

もっとも分かりやすい解決は、古い質問票を保存し、その答えを使い回すことでした。

なぜそれでは足りないのか、私たちはすぐに知りました。

受け取った文書のいくつかには、すでに答えが入っていました。雛形から来たものかもしれず、別の供給者のものかもしれず、過去のやり取りから来たものかもしれません。あるものでは、ISO 9001の認証が2017年のものだと書かれていました。その一文は申し分なく使えるように見えました。正確で、すでに整った文で、正しい設問の向かいに置かれていたのです。

正しい年は2018年でした。

誤りは一年分にすぎませんが、はるかに重要な問題を照らし出していました。古い答えは知識に似ています。断定的で、人を安心させる形をしています。AIが文書の中でそれに出会えば、たやすく取り込んでしまいます。けれども、そこにあるという事実は、誰が書いたのかも、何に支えられているのかも、私たちの製品を本当に述べているのかも教えてくれません。

そのとき私は理解しました。つくるべきは、写すための答えの書庫ではないのだと。私たちが本当に答えられると分かっていることの記憶を、つくらねばならなかったのです。

受け取った質問票には、ひとつの価値が残りました。顧客が何を知りたいのかを示していることです。けれどもそれだけでは、

答えを証明できません。この区別がコンテキストの規則になりました。顧客の文書は求めを保ち、答えは私たちが検証した情報に結びついたままでなければならない。

用紙ではなく、事実を保つ

私たちは提案依頼書から、次に役立つものを取り出し始めました。

設問はもう、顧客別や番号順だけで並べられてはいません。

テーマで近づけたのです。バックアップ、認証、暗号化、委託先、ホスティング、サービスの継続性。ある見込み客のところで「14.6.l」と呼ばれる設問は、別のところで「CL-072」と呼ばれる設問と、同じテーマを扱うなら合流できました。

こうしたテーマごとに、コンテキストは検証済みの事実とその出どころを保ちました。答えは、公式の文書、確認済みの設定、試験、あるいは担当チームの確認に支えられます。検証の日付と、私たちが表明できることの限界も残しました。

この作業はいつも完全な答えを生むわけではありません。穴も浮かび上がらせました。

情報が欠けているとき、システムはそれを告げなければなりませんでした。担当チームへの具体的な問いを用意し、その答えが質問票のどの行を埋められるのかを示しました。こうして、情報の欠落はもっともらしい一文の陰に隠れるのをやめました。それは見分けのつく行動になったのです。

最後に、答えを検証するためのものと、顧客に送れるものを分けました。ある設定の詳細は、社内の校閲者には欠かせなくて

も、公開する文書に居場所がないことがあります。ですからコンテキストは、見込み客に向けた言い回しを準備しつつ、その傍らに、検証に必要な根拠と留保を保ちました。

少しずつ、目的が変わりました。目の前の質問票を終わらせることだけではなくなったのです。 **検証されたひとつの答えは、次の仕事に必要な労力を減らすものでなければならない。**

四つめの質問票

四つめの提案依頼書には百十五の設問があり、最初のものよりはるかに多いものでした。

今度は、ばらばらのファイルの山から始めるではありませんでした。コンテキストは繰り返し現れるテーマをすでに知っていて、それに対応する事実をどこで見つけるかを心得ていました。九十六の完全な答えが、積み上がった記憶と製品の資料から直接、新たな収集なしに生み出せました。

システムは文書の残りも仕分けました。堅いけれども最後の確認を要する答えがありました。不確かなままのものもありました。手元の情報では出せないものもいくつかありました。

とりわけ、欠落の性質が変わっていました。誰が何を知っているのかを手探りで探して何時間も費やすことは、もうありません。まだ開いている点は、商業上の決定——たとえば契約期間——か、個別の技術的確認に対応していました。残る仕事は、見えていて、割り当てられて、限られていました。

この結果が、提案依頼書に「自動で応える」とは本当はどういうことかを理解させてくれました。AI にすべての欄を自信たっ

ぷりに埋めさせることはありません。埋められるものをただちに準備し、各回答の確からしさの度合いを示し、人間の判断をなお要するものをはっきり切り分けることです。

十分は、百パーセントの確かさを意味しない

私の使い方では、提案依頼書の準備は三～六時間から、文書の八割から九割を生み出すのに十分ほどへと変わりました。

この数字は誤った印象を与えかねません。八割から九割の答えが即座に確実になり、会社を拘束できる状態になるという意味ではありません。文書がおおむね準備されるという意味です。わかっている回答は記入され、根拠は示され、不確かな点には印がつき、足りない情報は具体的な確認事項に変わります。

人間の読み直しは欠かせないままです。ただし、それは然るべき場所に向けられます。古いファイルを漁り、同じ人たちにもう一度頼むことに時間を使う代わりに、校閲者は約束の内容を確かめ、何を開示するかを判断し、新しい答えや不確かな答えに集中できます。

ですからもっとも大きな得は、数時間が数分になったことだけではありません。仕事の性質が変わったことです。

最初の質問票は調査を強いました。続く質問票は、その調査が明らかにしたすべての恩恵を受けました。日付のひとつの誤りが、用心の規則になりました。あるチームから得た確認が、使い回せる事実になりました。答えのない設問が、見落としではなく、分かっている欠落になりました。

見込み客は変わらず自分の文書を受け取ります。

会社のほうは、それを準備するなかで学んだことを、もう失いません。

覚えておくこと——答えはその根拠によって価値を持つ

古い答えは知識に似ている。断定的で、形が整っていて、出どころと日付については黙っている。すでに埋まった用紙ではなく、検証された事実とその出どころを保つこと。はっきりした欠落は、もっともらしい一文に勝る。それは誰かに宛てられた問いになる。

まだ形をなさない考えから記事を書く

私がブログに載せる記事は、きちんと用意された白紙を前に、パソコンの前で始まることはめったにありません。

考えは、何かにつまずいたあと、道具を試したあと、議論のあと、意見が変わったあとに現れます。歩いているとき、電車の中、ソファの上で来ることもあります。文章にする価値があると感じますが、正確な切り口も、どんな順で語るべきかも、まだ分かっていません。

そんなとき、私はよく携帯を手に取ります。

録音を残したり、あとで書き写す文章を口述したりするためだけに使うではありません。**AI との会話を始めるのです。** AI は私の話を聞き、答え、考えているあいだに問い返してきます。この会話は、パソコンの前で再び開くのと同じ作業環境につながっています。ですから机から遠く離れた場所で始められます。あとで移し替え、解釈し直さねばならない孤立したメモをつくらずに。

形になりかけの考えを降ろす

私はひとつの単純な目的でコンテキストをつくります。記事を書くことです。そして頭にあるものをすべて渡します。

ときに二時間ほど、最初の点を語り、次の点を語り、また最初の点に戻ります。ある経験を話し、結論へ飛び、細部を足し、新しい筋を開き、前の筋をいったん手放します。同じ考えが違う

形で戻ってくるので、四度繰り返すこともあります。あることを断言し、一時間後に会話のなかで考えが変わって逆のことを言うこともあります。

この乱れは、始める前に避けるべき事故ではありません。生まれかけの考えが取る、実際の形であることが多いのです。

AI との対話は、早すぎる構造を押しつけずに進むことを可能にします。ある答えが、記憶をはっきりさせてくれます。ある問いが、私がいま一段飛ばしたことを明らかにします。ある言い換えが、当たり前だと思っていたことがまだそうではないと示してくれます。私は直し、含みを足し、別の方向へ進み直します。

コンテキストはこの言葉をセッションソースとして保ちます。もっとも整った文だけを取るではありません。言い直しも、ためらいも、矛盾も残します。考えがどう動いたのかを理解させてくれるものだからです。

そのうえで AI は、同じ考えの四つの言い回しを、四つの別の考えとして扱う代わりに近づけます。両立しない二つの断言に印をつけ、一時間後に現れた含みを見つけ出し、ある例とそれが照らす原理を結びつけます。 **乱れを消すではありません。扱えるものにします。**

この最初の段階が、早すぎる選択から私を解き放ちます。どの文で記事始めるか、この考えが一節に値するか、出来事をどの順で語るかを、まだ決めなくていいのです。まずは頭にあるものを空にしようとしています。通り過ぎるだけの考えを逃さずに。

それでもコンテキストは、この材料がなぜそこにあるのかを知っています。行き先のない会話を受け取るのではありません。これから書く記事の、まだ整理されていない材料を受け取るのです。

熟成の相手として AI を使う

最初の材料を出し終えたら、私は AI に何が足りないかを尋ねます。

性急な断言、一度も説明されていない前提、私の経験を読者がすでに知っていなければ成り立たない箇所を、AI は挙げます。例を求めてきます。別々のときに言葉にされた二つの考えを近づけます。もっともらしく見える筋を示し、それが私の経験に合わないの私が退けることもあります。

私は答え、はっきりさせ、直します。この会話はまだ記事を生みません。こうして材料の内容を充実させます。

AI の役割は大きいものの、その立ち位置ははっきりしています。私が流し込んだものすべてを手にしていて、私より速くそこに緊張を探します。その一方で、私が本当に何を主張したいのかは、私が言葉にするまで知りません。AI は問いを開きます。その問いが文章を動かすに値するかは、私が決めます。

この対話は、弱点に出会うまでの時間を縮めます。一人なら、例が足りないことや、結論が説明されていない段に乗っていることに、二日後に気づいたでしょう。AI は、考えがまだ直しやすいうちに、その発見を引き起こします。

経験だけでは足りなくなったら、ソースを足す

私の記事はしばしば、自分が経験したことから出発します。それでも外部から得た情報には、別の役割があります。

- ある参照は、私が観察したものに名前を与えます。
- ある技術文書は、仕組みを確かめてくれます。
- 別の書き手が、近い立場を掲げ、あるいは私が出会わなかった限界を示します。
- 補いの調べものが、私の経験は特殊な一例しか覆っていないと明かします。

これらの資料は**文書ソース**としてコンテキストに入ります。その役割は、最初の私の言葉とは異なります。**セッションソース**は、私が断言することと、経験から理解していることを保ちます。

文書ソースは、その経験と照らし合わせられる情報をもたらします。

そうすれば記事は、私が観察したこと、あるソースが打ち立てたこと、そして解釈にとどまるものを、区別できます。この区別は、よくある誘惑から文章を守ります。個人の経験を一般的な規則として示すという誘惑です。

ソースを、使い回せるアイデアに変える

この段階で、コンテキストはソースから記事の構成案へ直接進みはしません。まず材料を**アトミックノート**に変えます。

どのノートも、会話か資料から出た自立したひとつのアイデアを安定させます。

- 私が主張したい断言。

- それに具体的な形を与える例。
- その射程を限る含み。
- 記事が答えねばならない反論。
- 私の経験を裏づける、あるいは覆す外の事実。

ノートは出どころを保ちます。ですから会話をすべて読み返さずにアイデアをつかめますし、言い回しを確かめたりその場の事情を思い出したりする必要があるれば、ソースに戻れます。

この変換が、記事の準備の仕方を変えます。

- 会話は私の考えの動きをなぞっていました。ノートはこれから同じアイデアを語るものをまとめます。たとえそれらが一時間の隔たりをおいて、あるいは二つの別のソースから現れたとしても。
- ひとつの中心的なノートが、記事の切り口を浮かび上がらせることがあります。
- ほかのノートが、例や、根拠や、反論や、帰結をもたらします。

これらのノートはまだ段落でもなければ、出来合いの構成案でもありません。 **ノートは特定の成果物に縛られません。同じアイデアを、この記事にも、プレゼンテーションにも、別の仕事にも活用できます。**

書き始める前に構成案を通る

材料が十分にそろったら、私は構成案を求めます。

この段階が、豊かな会話からその順序をなぞるだけの文章へ、じかに移ることを防ぎます。話し合いは、立ち戻りと訂正と連

想で進みます。記事は、その話し合いもその経緯も知らない読者を導かねばなりません。

構成案がこの運びを定めます。何を最初に理解させるべきか、主な例をどこに置くか、どの反論を扱うか、結論が本当に断言できることは何かを示します。

AI は手元の材料から構成を提案します。私はそれを動かし、締め、脇筋に重きを置きすぎる部分を退けます。はっきりした向きを得るには、二、三往復で足りることが多いものです。執筆はそのあとに来ます。

この段階で AI は、漠然とした指示から文章をつくるのではありません。語られた私の経験、問いへの答え、ソース、そして承認された構成案から取り出されたアイデアとテーマの集まりとともに働きます。繰り返しをまとめ、例を置き直し、曲がりくねった話し言葉を読める散文に変えます。

私は結果を読み直し、三つの具体的なずれを探します。

- 消えてしまった含み。
- 断定的になりすぎた箇所。
- アイデアを裏切る言い回し。

それらを直し、そのうえで公開する版を決めます。

短い仕事にコンテキストがもたらすもの

記事は一日か二日で書けます。何年も追いかける製品や、何十章もある本と同じ記憶が要るようには見えません。

この軽い事例こそが、長さだけが問題ではないと見せてくれます。

コンテキストは、最初の言葉、加えたソース、立てた問い、選んだ構成案、そして完成した文章を、ひとまとめに保ちます。仕事が中断しても、その現在地が見つかります。ある一文が疑わしくなれば、その出どころに戻れます。書いているあいだに現れたアイデアが記事の枠を超えるなら、それだけを別に安定させ、将来の仕事を養えます。

こうした材料を、すべていつまでも残す必要はありません。小さなコンテキストは目的に達し、記事を生み、そして保管に回されてかまいません。その価値は、形になっていくあいだ仕事に与えた連続性と、文章がなぜこの方向を取ったのかをあとで理解したくなったときに残る痕跡にあります。

公開そのものが、別の作業を呼び起こすこともあります。要約を用意する、複数の言語に合わせる、掲載に必要な要素を整える。これらはいずれも同じ成果の延長であり、コンテキストがすでに知っている規則のもとで行われます。

全体の流れは単純です。経験が言葉になり、その言葉が問いやソースと結びつき、材料が構成案になり、構成案が文章になります。

AI はそれぞれの変換を速め、コンテキストは一方から他方へ導く糸を保ちます。

この動きのどれひとつとして、長い執筆だけのものではありません。携帯の中で忘れられる録音よりましな扱いに値する考えが生まれた瞬間から、それは始まります。

要点——まとまっていない考えも材料になる

生まれかけの考えは繰り返し、矛盾し、後戻りする。それが普通の形であって、始める前に正すべき欠陥ではない。コンテキストはこの言葉をセッションソースとして保ち、そのうえで扱えるものにする。構成案は会話と文章のあいだに挟まる。読者は話し合いの順序を追わないからだ。

混乱のなかでプロダクトのテーマを育てる

前の二章は、ひとつのテーマを初めから終わりまで追いました。この章が述べるのは、プロダクトマネージャーの典型的な一日の現実です。いくつものテーマ、依頼、時間の尺度が同時に進んでいきます。

私は会社でただ一人のプロダクトマネージャーです。つまり、製品にかかわる情報の大半が、最後には私の前を通ることになります。

同じ一日のうちに、ある顧客が重要に思える一言を口にします。

- それは数語に収まります。
- ほかの利用者も抱える問題を映しているのか、個別の困りごとにすぎないのか、まだ分かりません。けれども失いたくはありません。

少しあとで、ある業務上の論点の変化についてのホワイトペーパーを手に入れます。

- 今度の情報は一文には収まらず、数十ページの PDF です。
- 何が役に立つのかも、この資料が現場で見ているものを裏づけるのかも、まだ分かりません。

それから開発者たちが、作りかけの機能について話しかけてきます。

- ひとつの問いに答え、ある振る舞いをはっきりさせ、数週間前に置いた仮定に立ち戻る必要があります。

その間に CPO は、製品の別の部分にかかわる二つの賭けについて状況を確かめたがっています。

サポートとの会話が最後に明かすのは、機能の説明文書が足りていないということです。

- その文書の一部は、まだ私の頭の中にしかありません。

一日の終わりには、三つか四つの異なるテーマ（あるいはもっと）を渡り歩いています。数語の引用、一冊まるごとの資料、進行中の開発についての問い、判断の依頼、そして文書の必要を受け取りました。

これはとりわけ悪い一日ではありません。ごく普通の日です。

大きさも行き先も違う入り口

プロダクトマネジメントの混沌は、情報の量だけから来るものではありません。その性質の違いから来ます。

この五つの入り口は、それぞれ製品の生涯の別の瞬間を見えています。

- 顧客の一言は弱い兆候であり、ほかの兆候を告げるものかどうかはまだ分かりません。
- PDF はある業務の全体像を与えます。
- 開発者との話し合いは、いま作っているものについてです。
- サポートからの依頼は、すでに存在し、説明されるべきものにかかわります。
- CPO からの問いは、ありうる複数の未来を比べることを強めます。

これらが単一の文書に入るべきいわれはありません。

- すべてが同じテーマを語るわけではなく、製品の生涯の同じ瞬間にいるわけでもありません。

これらをただメモの一覧にまとめれば、情報は保たれますが、理解する仕事を、それが必要になる日まで先送りすることになります。

コンテキストは、複数の糸を並行して保つことを可能にします。

- 引用は、意味がまだ定かでなくとも、属していそうなテーマに合流します。
- PDF は出どころとともに保たれ、本当に役立つわずかなアイデア（**アトミックノート**）を取り出すために分析できます。
- 開発者との会話は、進行中の機能の仕事に合流します。
- サポートが指摘した欠落は、会議の合間に耳にした指摘のままにしている代わりに、生み出すべき仕事になります。

すべてがただちに知識に変わるわけではありません。

- 顧客の一言は曖昧なままかもしれず、
- ホワイトペーパーはあまり的を射ていないかもしれず、
- 直感は翌週に覆されるかもしれません。

コンテキストは、入ってくるものに知性を与えません。場所と、出どころと、しかるべきときに見つけ出される機会を与えます。

コンテキストは混沌を収めます。なくすふりはしません。

情報が入ってくるあいだも、出す依頼は続く

一日の仕事は、情報を受け取って仕分けるだけではありません。成果を生み出す必要もあります。

たとえばサポートは、いま私たちが手がけている機能について、機能説明の文書を求めてきます。

- その働きを説明し、重要な場合を明らかにし、顧客に寄り添う人たちが正しく答えられるだけのものを渡さなければなりません。

同じ日に、会社のブログ向けの記事を仕上げたいと思うこともあります。

- この記事は、必ずしも同じ機能について語るわけではありません。
- それでも同じ業務上のテーマを扱うことはありえます。製品が応えている業務、利用者がたびたび出会う困難、あるいは彼らのやり方の変化です。

そのとき二つの文書は、同じ知識の一部から汲みながら、同じことを語りません。

- サポート向けの文書は、製品が実際にどう働くかを説明します。
 - ▶ 正確で、実務的で、利用者の問いに合っていなければなりません。
- 記事は、経験や、業務上の問題や、考察を展開します。
 - ▶ 同じアイデアの一部を使えますが、読者と目的に合わせて別の形に構成します。

私はそのテーマについて知っていることを、二度整理し直す必要があります。

- コンテキストは、まだ文書にも記事にもなっていない中間情報（アトミックノートとテーマノート）を保っています。
- この情報から、AI は複数の構成案を用意できます。ひとつは製品の使い方を軸に、もうひとつはより広い業務上の論点を軸にしたものです。

もちろん私は二つの結果を読み直します。

- サポート向けの文書は、製品の理解を左右します。
- 記事は、私の見解を左右します。

けれども私は、同じ要素を探し直して初めから書き直すことではなく、中身と向きを確かめることに時間を使います。

この動きは同じ一日のうちに起こりえます。新しい情報が入ってくるあいだに、すでに整理された知識が、会社や私自身の必要とする文書を生み出させてくれるのです。

小さな一言が、プロダクトのテーマになりうる

朝に耳にした引用が、必ず決定を引き起こすわけではありません。

- それだけを取れば、何も証明しないかもしれません。
- 数週間後、別の顧客が近い困難を口にするかもしれません。
- ある面談が含みをもたらすかもしれません。
- その日に手に入れたホワイトペーパーが、観察された変化に名前を与えるかもしれません。
- サポートのある数字が、思っていたより頻繁にこの問題が戻ってくると示すかもしれません。

これらは別々のときに、別々の形で届きます。

- コンテキストは、出どころを消さずにそれらを近づけることを可能にします。
- そして次のことができます。
 - ▶ 新しい指摘が、すでに保たれている兆候に似ていると告げる。
 - ▶ 矛盾を浮かび上がらせる。
 - ▶ 二つの解釈に決着をつけるには何が要るのかを尋ねる。

少しずつ、孤立した一言が、取り組むに値するテーマになりえます。そのとき私はすでに、その経緯の一部を手にはしています。最初の兆候、それを照らす資料、それを裏づけるもの、それに反するもの、そしてまだ開いている問いです。

コンテキストは、この情報からテーマが育つのを助けます。

- 問題が重要だと決めるのではなく、私の代わりに解決策を選ぶのでもありません。

プロダクトの賭けの責任は私にあり続けます。

- その一方で、おぼろげな記憶や散らばった資料の束とは違うものとともに、その決定を下せます。

同じシステムがそのあとの仕事にも寄り添います。

- 面談を加える、
- 仮説を事実に突き合わせる、
- 筋書きを用意する、
- CPO 向けの論拠を生み出す、
- 仕様を書く、
-

こうして、一日のただなかで拾われた小さな兆候が、本当の決定になるまで十分に長く生き続けられます。

見送ったアイデアも、あとで育て直せる

すべての探索が本番の投入まで進むわけではありません。

面白そうに見えながら、手元の技術ではまだきちんと実現できない人工知能の使い方に取り組んだこともあります。

- アイデアが悪かったとはかぎりません。
- 時機が悪かったのです。

従来のやり方なら、探索はいくつかの資料を生み、断念の決定のあとフォルダの中へ消えていったでしょう。

- 一年か二年後、それらのファイルを探し出し、出発点の仮説を思い出し、なぜ止めたのかを理解しなければならなかったはずです。

コンテキストがあれば、探索はその道のりとともに保管できます。最初の兆候、思い描いた筋書き、試したこと、ぶつかった限界、そして止めた理由です。

- 技術が変われば、過去の否定的な結果を永久に悪いアイデアへ変えてしまうことなく、この仕事を再開できます。

私たちが何をしていたのかだけでなく、なぜそれが動かなかったのかも見つかります。そのうえで、障害がまだ存在するかを確かめ、なお有効なものを保ち、そこから考えを再開できます。

ですからプロダクトのテーマは、途切れることなく育つとはかぎりません。

- 一言から始まり、何か月もかけて豊かになり、中断し、製品や市場や技術が変わったときにまた動き出すことがあります。

一日を遅くせずに、秩序を立ち上がらせる

私の一日は細切れのままです。

- 顧客は、自分の指摘をサポートの依頼と足並みをそろえてはくれません。
- 開発者は、私が PDF を読み終えた瞬間を選んでくれません。
- 準備すべき決定は、私が文書や記事を書いているあいだも止まりません。

コンテキストは、この現実を完璧な作業の連なりに変えたりしません。

ひとつひとつの断片が連続性を保てるようにします。

- 小さな一言と長い資料は、同じ扱いを受けずに入れます。
- 複数のテーマは、混ざらずに進めます。
- すでに手を入れた知識は、最初の成果物に閉じ込められずに、複数の成果物に使えます。
- 中断したアイデアは、経緯のすべてを失わずに待てます。

私がプロダクトマネジメントでこれを使うのは、そういうわけです。一日の揺れを減らすためではなく、その揺れが学んだことを散らさないようにするためです。

混沌は入り続けます。

ただ、それはもう損失の形では出ていきません。

この三つの使い方には、まだ名づけていない共通点があります。どれも、去っていく何かで終わるということです。送られた回答、公開された記事、下された決定。

次の部では、成果物が去ったあとに何が残るのかを見ます。

覚えておくこと——複数の糸を並行して

顧客の一言と五十ページのPDFは、違う扱いと違う行き先を求める。コンテキストはそれぞれの断片に、場所と、出どころと、しかるべきときに見つけ出される機会を与える。手放した探索は、止めた理由を保つ。技術や市場が変われば、ふたたび再開できるものになる。

部 4

成果物があとに残していくもの

前の三つの使い方は、どれも成果物を完成させて終わりました。この部では、成果物を届けたあとに何が残るのかを見ます。

六か月の間において同じ質問に二回答える事例を、軸にして考えます。一度目、答えは表計算の一つのセルに収まります。二度目にもその回答は残っていますが、それだけではもう足りません。欠けているのは、回答に至った過程と、それを支えた資料同士の関係です。

答えはセルに収まるが、筋道は収まらない

前の部の提案依頼書のひとつに戻りましょう。

設問は、表計算のセル D47 に届きます。

バックアップは保存時に暗号化されていますか。

最終的な答えは三行に収まります。 それを書くために、私はいくつもの要素を集めなければなりません。

- 私たちの技術文書は、バックアップの一般的な仕組みを述べています。
- ある設定ファイルが、実際に有効な設定値を示しています。
- 運用チームが、その設定値が本番でどう効いているかを確認します。
- 法務が、見込み客に何を伝えてよいかをはっきりさせます。

私は表計算から文書へ、文書から設定へ、そして運用チームとのやり取りから法務の判断へと移ります。 最後にセル D47 へ戻り、そこに答えを書きます。

回答書を送付します。

連なりの果てにある三行

セルは仕事の結果を収めています。 それを生み出した道のりは見せません。

セルは次のことを語りません。

- どの文書を参照したのか。

- どの設定値を確かめたのか。
- 誰がその適用を確認したのか。
- どの言い回しが許されたのか。
- 見込み客に送る文からどの留保が外されたのか。

それでもこれらの要素は存在します。三行のそれぞれを決めたのはそれらであり、答えを書いているあいだ、その関係は私の頭の中ではっきりしています。

回答書を送ってしまうと、このつながりは見えなくなります。表計算は答えを保ちますが、それを信頼させていた筋道は保ちません。

どの道具も自分の仕事をした

問題は片づけ方が悪いことから来るものではありません。

- 文書はきちんと文書の置き場にありますが。
- 設定は製品の中に残っています。
- チームの回答はその会話のスレッドの中にあります。
- 法務の判断にはそれ自身の媒体があります。
- セル D47 は、見込み客の文書の中のあるべき場所にありますが。

どの道具も、委ねられた対象を保ちました。筋道の全体を保つことを役目とする道具は、ひとつもありませんでした。

その過程を再現するには、何をどこで探し、それぞれがどんな役割を果たしたのかを知っていなければなりません。この知識は、答えを準備した人が抱えたままです。

成果物は、それを生み出したものを平らにする

成果物は、受け取る相手に合った結果を示さなければなりません。

- 見込み客が待っているのは明快な答えであって、それを得るために行った調査の全記録ではありません。

この単純化は必要なものです。それが高くつくのは、成果物が仕事の唯一の痕跡になったときです。

最終的な結果は、そこへ至る道のりよりつねに短いものです。送られた回答も、公開された記事も、プロダクトの決定も同じです。結果だけが残るなら、次の仕事はその道のりの一部をやり直さなければなりません。

筋道に宛て先を与える

コンテキストは、表計算の代わりでも、文書の代わりでも、設定の代わりでも、チームとのやり取りの代わりでもありません。

- それらを結ぶ仕事に、共通の宛て先を与えます。

設問 D47、参照した要素、得られた承認、生み出した答えは、同じテーマに属しています。それぞれが元の形と元の場所を保ったまま、コンテキストの中で結ばれていられます。

セルは自分の役目を果たし続けます。見込み客に答えることです。コンテキストは別の役目を果たします。この答えを理解し、確かめ、再開することを可能にするものを保つことです。

六か月後、この違いが決定的になります。

要点——成果物だけでは過程は残らない

答えは三行に収まりながら、いくつものソースと承認と留保に依存しうる。道具はそれぞれ自分の対象を保つ。そのあいだの関係は、いまなお人間が抱えている。コンテキストは、元の道具に取って代わらずに、それらを結ぶ仕事へ宛て先を与える。

六か月後、その答えではもう足りない

セル D47 から六か月後、新しい見込み客がバックアップの暗号化について同じ問いを立てます。

- 私は十秒で以前の答えを見つけます。
- 明快で、よく書けていて、そのまま写せる状態です。

それでも、その回答をすぐに再利用することはできません。

三つの問いが、再開を止めるのに十分です。

- この答えはどこから来たのか。技術文書か、観察した設定値か、あの日に運用チームが説明してくれたことか。
- 法務が承認したのは、この正確な言い回しか、それとももっと慎重な版か。
- 設定値は今も同じなのか。

答えは生き延びました。それを信頼させていたものは消えました。

見つけることは、まだ再開することではない

古いセルは文章を返してくれます。返してくれないものがあります。

- その根拠の水準も、
- その限界も、
- ふたたび使う前に必要な確認も。

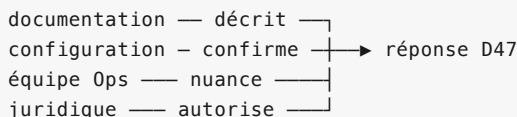
ほかに記憶がなければ、前の章の調査をすべて、逆の順でやり直さなければなりません。すでに答えを持っているのですから、

まったくのゼロから始めるわけではありません。それでも、仕事のどの部分がまだ有効なのかを知らずに始め直すことになります。

認証年を 2018 年ではなく 2017 年と記した以前の誤りと、同じ問題が今度は私たち自身の回答に起きています。文章は明快で、自社が作成したのですが、その根拠が見えなくなっています。

それぞれの要素の役割を取り戻す

コンテキストの中では、古い答えは独りで残りはしません。それを生み出した筋道に役立つ関係を保っています。



動詞が、それぞれの要素で何ができるのかを変えます。

- 技術文書は一般的な仕組みを述べます。それだけでは有効な設定値を証明しません。
- 設定はある時点でその値を確認します。何を伝えてよいかは決めません。
- 運用チームは、実際の運用から来た含みをもたらします。
- 法務は、見込み客に向けた言い回しを許します。

私はもう、答えの近くにある四つの対象を見つけるだけではありません。それぞれが果たしていた役割を取り戻します。

この厳密さが、あるソースが実際に打ち立てている以上のことを証明してしまうのを防ぎます。時が経ったときに、どの確認をやり直すべきかも示します。

変わったかもしれないもの確かめる

コンテキストは、古い答えを永続する真理に変えたりしません。その検証を準備します。

- 技術文書はまだ有効でありながら、設定が変わっていることがあります。
- 設定は同じままでありながら、許された言い回しを見直すべきことがあります。
- 運用チームがもたらした含みは、製品の変化のあとに消えていることがあります。

関係が、こうした違いを目に見えるようにします。調査全体を手当たりしだいにやり直さずに、変わったかもしれない要素を点検できます。

最終的な答えは、六か月前と同じになるかもしれません。違いは、なぜそれをまだ使えるのかを私が知っている点にあります。

仕事に役立つ関係を保つ

すべての資料を互いに結ぶことが目的ではありません。

- 二つのファイルは、同じ筋道に加わらないまま、ひとつの語を共有することがあります。

関係が保たれるに値するのは、それが理解や検証や決定や制作のしかたを変えるときです。

- D47 の場合、四つの関係で足りる。述べる、確認する、含みを足す、許す。

これらが、具体的な問いに答えさせてくれます。

- この答えは何に支えられているのか。
- 今日、どの部分を確認めるべきか。
- 何を伝えてよいかを承認したのは誰か。
- 何が起きれば、この答えは通用しなくなるのか。

ですからコンテキストは、私が知っていることすべての完全な地図をつくろうとはしません。特定の仕事に必要な情報のつながりだけを保ちます。

機械を開く

第一書は、コンテキストがどう生まれたのか、AI との関係で何を変えるのか、いくつもの使い方にどう寄り添うのかを示しました。D47 の事例は、まだ理解すべきことを浮かび上がらせます。

- 毎回の作業で繰り返さずに、仕事の使命をどう保つのか。
- ソース、アイデア、決定、成果物を、それぞれ別の役割を保ったままどこに置くのか。
- すべての情報を会話に持ち込まずに、いま必要なものだけをどう読み込むのか。

第二書は、この機械を開きます。情報がコンテキストに入り、知識へ変換され、成果物となり、次の作業に引き継がれるまでの流れを追います。

要点——信頼の根拠を取り戻す

答えを見つけるだけでは足りない。それをまだ使えると言えるものを取り戻さねばならない。名前のついた関係は、あるソース、ある承認、ある留保の正確な役割を示す。コンテキストは、答えを支えているものと、変わったかもしれないものを見せることで、検証を準備する。

書 II

コンテキストを築き、生かし続ける

- 5 コンテキストの内側
- 6 コンテキストとその生態系

部 5

コンテキストの内側

ここまでは、コンテキストで何ができるのかを中心に説明してきました。この部では、その中身を開きます。まずディレクトリ構成の全体を示し、そのうえで一回の作業セッションを初めから終わりまで追います。コンテキストを開いて現状を把握し、情報を取り込み、アイデアに変え、成果物を生み出し、判断を下し、次のセッションですぐ再開できるよう保存します。この六つの章を貫く例はすべて架空のものです。現場の技術者の一部がなぜ不完全な作業報告書を書くのかを理解し、そのうえで製品を変えるべきかどうかを決めること。

全体の地図

第2部では、作業報告書の品質に捧げたコンテキストをそのままにしていました。

- 三つのソース、
- 二つのアトミックノート、
- ひとつのテーマノート、
- 次の面談のための手引き。

あのとき私は、それが何を理解させてくれるかを見ていました。

いま、ディレクトリそのものを開きます。

本書で使っている版の、ディレクトリ構成の全体です。

```

contexte-qualite-comptes-rendus/
├─ README.md           全体のナビゲーション、自動生成
├─ offline.md          セッション外の受信箱
├─
├─ 00-frame/           長期的な方針
│   ├── README.md      区画のナビゲーション、自動生成
│   ├── mission.md     目的、問い、限界
│   ├── directives.md  使命に固有のルール
│   └─ loading.md      開くときに読まれるもの
├─
├─ 01-state/           仕事の現在
│   ├── README.md      区画のナビゲーション、自動生成
│   ├── state.md       確かなこと、不確かさ、未済
│   ├── synthesis.md   定着した内容
│   ├── todo.md        行うべき行動
│   ├── decisions.md   コンテキストを形づくる選択
│   ├── contradictions.md 緊張のまま残るもの
│   ├── handoff.md     再開の手引き
│   └─ checkpoints.md  状況の圧縮された連なり

```

└─ health.md	最後の点検の結果
└─ 02-history/	日付きの事実の痕跡
└─ README.md	この痕跡の役割、自動生成
└─ YYYY-MM-DD.md	その日の出来事
└─ 03-sources/	受け取った、または言葉にしたもの
└─ README.md	ソースの一覧、自動生成
└─ ingestion-log.md	読まれたもの、それが生んだもの
└─ documents/	
└─ README.md	資料の一覧、自動生成
└─ raw/	元のファイル
└─ fulltext/	扱える全文
└─ SRC00001-....md	素性、来歴、状態
└─ sessions/	
└─ README.md	セッションの一覧、自動生成
└─ SESSION-....md	仕事に注がれた言葉
└─ 04-notes/	ソースから引き出した知識
└─ README.md	ノートの地図、自動生成
└─ atomic/	一ノートーアイデア - AT000001-....md
└─ thematic/	結びつける切り口 - THM00001-....md
└─ glossary/	安定した用語 - GL000001-....md
└─ decision-records/	ひとつの判断 - DRE00001-....md
└─ 05-outputs/	ある用途のために生み出すもの
└─ README.md	結果の一覧、自動生成
└─ _draft/	作りかけの成果物
└─ questions-entretien.md	続けるための手引き
└─ note-de-cadrage.md	決めるためのまとめ
└─ presentation-equipe.md	同じ知見を言い直したもの

番号は読む順序を決めるものであって、価値の上下ではありません。

- 枠組みと状況が仕事の向きを定めます。
- 履歴がその進み方を点検させます。
- ソースが情報を保ち、ノートがそれを知識に変えます。
- 成果物が、受け取る相手に合わせた形を与えます。

六つの区画、六つの役割

00-frame/は「なぜ私はここにいるのか、どんなルールのもとで働くのか」に答えます。この区画はめったに変わりません。当然です。使命を変えれば、すべてのソース、すべての作業、すべての結果の判断の仕方が変わってしまいます。

```

contexte-qualite-comptes-rendus/
├─ 00-frame/                長期的な方針
│   ├── README.md          区画のナビゲーション、自動生成
│   ├── mission.md         目的、問い、限界
│   ├── directives.md      使命に固有のルール
│   └─ loading.md          開くときに読まれるもの

```

01-state/は「いま私はどこにいるのか」に答えます。この区画は仕事とともに動き、確かになったことと仮説、これからの行動とすでに下された決定、一時的な困難と見えたままにしておくべき矛盾を、見分けさせてくれます。

```

contexte-qualite-comptes-rendus/
├─ 01-state/                仕事の現在
│   ├── README.md          区画のナビゲーション、自動生成
│   ├── state.md           確かなこと、不確かさ、未済
│   ├── synthesis.md       定着した内容
│   ├── todo.md            行うべき行動
│   ├── decisions.md       コンテキストを形づくる選択
│   ├── contradictions.md  緊張のまま残るもの
│   ├── handoff.md         再開の手引き
│   ├── checkpoints.md     状況の圧縮された連なり
│   └─ health.md           最後の点検の結果

```

02-history/は「実際に何が起きたのか」に答えます。この区画はセッションを美しく語りません。役に立つ事実を残します。ソースが加わった、ある筋を手放した、決定を下した、ノートを

作った。参照することはめったにありませんが、現状が筋の通らないものに見えた日には決定的になります。

```

contexte-qualite-comptes-rendus/
├── 02-history/                日付きの事実の痕跡
│   ├── README.md            この痕跡の役割、自動生成
│   └── YYYY-MM-DD.md        その日の出来事

```

03-sources/は「どんな情報を受け取ったのか」に答えます。外から来た資料は固まったまま入り、私自身の言葉はセッションを重ねながら入ります。どちらの場合も、ソースはコンテキストがそこから理解するものとは切り離されたままです。

```

contexte-qualite-comptes-rendus/
├── 03-sources/                受け取った、または言葉にしたもの
│   ├── README.md            ソースの一覧、自動生成
│   ├── ingestion-log.md     読まれたもの、それが生んだもの
│   ├── documents/
│   │   ├── README.md        資料の一覧、自動生成
│   │   ├── raw/             元のファイル
│   │   ├── fulltext/        扱える全文
│   │   └── SRC00001-....md   素性、来歴、状態
│   └── sessions/
│       ├── README.md        セッションの一覧、自動生成
│       └── SESSION-....md    仕事に注がれた言葉

```

SRC00001-....md という管理ファイルには、資料そのものも、その要約も入っていません。その資料について私が知っておくべきことすべてに答えます。

- どこから来たのか、なぜ加えたのか。
- 原本とその全文はどこにあるのか。
- 取り込みは成功したのか。

- どの程度信頼してよいのか。
- それを読んだ結果、すでにノートが生まれているのか。

この管理ファイルによって、情報そのものを変更せずに、ソースの状態を確認できます。

04-notes/は「何を学んだのか」に答えます。

- アトミックノートはひとつのアイデアを安定させます。
- テーマノートは複数のアイデアを結びます。
- 用語集はひとつの語をはっきりさせます。
- 判断の記録は、ある決着の理由を保ちます。

```
contexte-qualite-comptes-rendus/
```

```
|— 04-notes/                ソースから引き出した知識
|   |— README.md           ノートの地図、自動生成
|   |— atomic/             ノートアイデア - AT000001-...md
|   |— thematic/          結びつける切り口 - THM00001-...md
|   |— glossary/          安定した用語 - GL000001-...md
|   |— decision-records/   ひとつの判断 - DRE00001-...md
```

05-outputs/は「この知識で何をするのか」に答えます。下書きと完成した成果物をここに置き、読者が求める形式に整えます。

- 面談の手引き。
- 記事。
- 仕様書。
- 発表資料。
- 決定。ほかの形もありえます。

```

contexte-qualite-comptes-rendus/
└─ 05-outputs/                  ある用途のために生み出すもの
   └─ README.md                 結果の一覧、自動生成
   └─ _draft/                   作りかけの成果物
   └─ questions-entretien.md    続けるための手引き
   └─ note-de-cadrage.md        決めるためのまとめ
   └─ presentation-equipe.md    同じ知見を言い直したもの

```

アイデアの世界——いくつもの建物に使えるブロック

04-notes/と05-outputs/の分離は、コンテキストの主な強みのひとつを担っています。ノートと用語集はそこで*アイデアの世界*を形づくりま。何度でも使える知識のブロックの集まりであり、もう生の情報ではなく、まだ誰かに宛てた結果でもありません。

```

03-sources/
|
▼
04-notes/                      アイデアの世界
└─ AT000001                    安定したひとつのアイデア
└─ AT000002                    安定した別のアイデア
└─ THM00001                    その関係のひとつの読み方
   |
   └─> questions-entretien.md
   └─> note-de-cadrage.md
   └─> presentation-equipe.md

```

私の例では、入力 of 摩擦と構造化データの必要は、どんな形で示されるかを私が知る前に安定していました。

- 面談の手引きはそこから質問を引き出し、
- 枠組みの覚書はそこから論点を引き出し、
- 発表資料はそこからチーム向けの筋立てを引き出します。

この三つの成果物は、使っている知識を所有していません。

- ある必要、ある読者、ある形式のために、その一部を選び、組み直しているだけです。
- ですから成果物は、自分を作っているノートを使い尽くしません。ブロックは別の建物のために残ります。

この中間の世界がなければ、新しい結果を出すたびにソースを開き直し、照らし合わせをやり直し、同じ確かなことを言い直さねばならないでしょう。

コンテキストはすでに生み出されたものを保つだけではありません。ふたたび生み出す力を持続させるのです。

すべてのファイルが同じように変わるわけではない

この木が本当に読めるようになるのは、それぞれの要素がどう変わるのかを知ってからです。

いま手に入るいちばん良い状況を示すために書き直されるファイルがあります。

- `state.md`。
- `handoff.md`。
- `05-outputs/_draft/`の下書き。

過去を保つために、ただ長くなっていくだけのものもあります。

- セッションのソース。
- 日付のついた履歴。
- 統合された知見。
- チェックポイントの連なり。

受け取った資料は固まったままで、別のノートに置き換えられたノートも参照できます。 `README.md` は、存在するものから生成さ

れます。手で直せば現実の第二の版ができてしまい、やがて食い違います。

この多様さが、ひとつの大きなファイルではもう足りなかった理由を説明します。同じ文章にこう求めねばならなかったはず

- 現在を示すこと、
- 過去を保つこと、
- 生のものを受け取ること、
- アイデアを安定させること、
- 完成した結果を担うこと。

写し取るレシピではなく、骨組み

この骨組みは、コンテキストの脇に置かれた説明書ではありません。本物のディレクトリであり、私はそれを複製します。

- どのファイルも、含むべき節といくつかの手引きを携えて現れます。おかげで何か月も隔てて作られた二つのコンテキストが、同じ目印を共有します。

それでも構造は変えられます。コンテキストは次のようなものを受け取れます。

- 固有の指示、
- その仕事ならではの成果物、
- これまでになかった種類のソース。

共通の土台が守っているのは、とりわけ私の仕事が必要とする三つの境界です。

- 変わらないものと、いまのもの。

- 生のものと、解釈されたもの。
- 知識と、その体裁。

コンテキストに話しかける

ここまで、私はディレクトリとファイルを見せてきました。それぞれを開き、どこに書くかを知り、情報をひとつずつ自分で動かさねばならないように思えるかもしれません。私の働き方はそうではありません。

私は AI に話しかけます。

| 作業報告書の品質についてのコンテキストを開いて。

| この資料をソースとして追加して。

| このソースをノートに変えて。

| 作業を保存して。

これらの求めのひとつひとつの背後には、私が一度書いた手順があり、AI は毎回のセッションでそれを使えます。

- Claude Code と Codex では、この種の再利用できる手順に名前があります。**スキル**です。

たとえば開くためのスキルは、次のことを示します。

- 始める前に取り込むべきもの。
- 必ず読むファイル。
- 作業がそれを求めるまでディスクに置いたままにするもの。
- 問題をどう知らせるか。
- 私がすぐ動けるように、何を返すべきか。

私はこの手順を毎回与え直す必要がありません。

- コンテキストを開いてと頼むだけです。

会話はその時々私の意図を担います。

- スキルは、繰り返される所作を確実に行うやり方を担います。

こうして私のシステムには、いくつかの安定した動詞が備わりました。

- コンテキストをつくる。
- それを開き、進行中の作業を取り戻す。
- ソースを追加する、またはセッションを保つ。
- ソースをノートに変える。
- 作業状況を保存する。
- その整合を点検する。
- 目的を果たしたら保管する。

AI とのやり取りがすべてスキルを通るわけではありません。考え、問い、直し、判断することは、普通の言葉で続けています。

スキルが枠をはめるのは、つねに同じ手順を踏むことが求められる作業だけです。

- たとえば、出どころを保たないままソースが変換されることや、再開の準備をしないままセッションが終わることを防ぎます。

手順は安定していても、判断が自動になるわけではありません。二つのモデルは、ノートを違う言葉で書くかもしれません。それでも同じ要素を参照し、同じ境界を守り、同じ点検を行わねばなりません。

ですからスキルを共有しても、誰かに私のコンテキストを渡すことにはなりません。自分のコンテキストをつくり、生かし続けるための、同じやり方を渡すことになります。

その仕組みについて、これ以上は述べません。

スキルはコンテキストの発明でも、その特異な点でもありません。Claude Code と Codex は、まさに再利用できる手順を書き表すためにそれを備えており、その作り方と使い方は、ここで私が述べるより公式の文書のほうがうまく説明しています。

コンテキストに属するのは、どんな能力を与えるかという選択です。つくる、開く、与える、変える、保存する、点検する、保管する。そして、それらがどんな順でつながるのか、それぞれがどんな境界を守るべきなのかもです。

ファイルはコンテキストが知っていることを保ちます。スキルは、情報を変換する手順を定めます。

地図は開かれました。見学は、本物のセッションと同じように始められます。コンテキストを開くところから。

覚えておくこと——コンテキストの骨組み

新しいコンテキストは家具つきで届く。六つの区画があり、それぞれが別の問いに答える。番号は読む順序を決めるもので、価値の上下ではない。この木を読むようにするのはその形ではなく、どのファイルがどう変わるのかを知っていることである。スキルという仕組みは Claude Code と Codex が与える。コンテキストが定めるのは、繰り返される所作と、その順序と、その限界である。

開く——使命と現状を取り戻す

作業報告書についての前回のセッションから二週間が経ちました。報告書とは、技術者が設備での作業のあとに、見たことと行ったことを書き留める文書です。

- 面談もノートも、いつもの場所にそのまま残っています。

けれども手当たりしだいにファイルを開けば、すでにわかっていたことを整理し直すだけで三十分かかり、それでも正確に把握できないかもしれません。

- ですから開く作業は、はるかに小さな核から始まります。枠組み、現在の状況、そして再開です。

このすべてを私が頼む必要はありません。開くためのスキルを呼べば、手順が流れていきます。

まず、なぜ自分がここにいるのかを取り戻す

AI が（コンテキストを開くスキルを通じて）最初に読むファイルは `00-frame/mission.md` です。その雛形は五つの問いを立てます。

使命

目的

この`コンテキスト`は何を可能にすべきか。

中心となる問い

選り分けと探索を導く問いは何か。

対象に入るもの

どんな情報が入れるか。

対象外のもの

面白く見えても、何を断るか。

期待される成果物

すでに分かっているなら、何を生み出したいか。

私のものでは、冒頭はこう埋まっています。

目的

作業報告書の品質を損なっているものを理解する。

中心となる問い

入力を嫌がらずに、活用できる情報をどう集めるか。

対象外のもの

- 複数のソースを突き合わせる前に画面を選ぶこと。
- ひとつの面談から一般化すること。

期待される成果物

プロダクトの決定と、次の面談を準備する。

「報告書を良くする」としか言わない使命は、私をあまり導いてくれません。中心となる問いは、それだけである資料をほかより重要にしますし、対象外の定めは、魅力的なアイデアが探索を黙って解決策づくりへ変えてしまうのを防ぎます。

- 成果物は、分からないままにかまいません。

- 早すぎる出口を発明するより、この不確かさを書きます。そうすればコンテキストは、まだ探索の最中だと分かります。

自分が置いたルールが待っている

近い使命でも、必要な用心は違うことがあります。 `directives.md` はそのルールを仕事とともに保ちます。

- 守秘、
- 根拠の水準、
- 用語、
- 承認や求められる形式
-

雛形はあえて単純です。ひとつのルール、ひとつの日付、命令形のひとつの文。

指示

- (2026-09-11) 面談は取り込む前に匿名化すること。話者の名前を役職に置き換え、会社名と国は残す。
- (2026-09-11) ソースが示すことと、そこから私たちが推し量ることを区別すること。
- (2026-09-11) 実際の使われ方を記録するまで、いかなる解決策も提案しないこと。

指示は、これからの行動を導けるときにだけ値打ちがあります。

- 「慎重に」では、解釈をすべて私がやり直すことになります。
- 「観察と推論を区別する」なら、ノートや成果物を書くときに具体的な点検を与えてくれます。

けれども指示は、ファイルを開いたときに見つかる心得にとどまりません。対象となる作業を行うたびにコンテキストが適用すべき、継続的なルールです。

たとえば私は新しい面談の書き起こしを加えます。それに被面談者の名前が入っていることを思い出さないまま、コンテキストに（専用のスキルで）取り込んでとだけ頼みます。それでも匿名化の指示は取り込みのあいだに働きます。書き起こしをソースとして登録する前に、スキルはその名前を役職に置き換え、会社名と国は残します。

ルールが必要になるまさにその瞬間に、私がそれを覚えている必要はありません。コンテキストが私の代わりに覚えています。ですから指示は、開く作業が終わっても働きをやめません。セッションのあいだずっと効いていて、コンテキストのさまざまな所作に付き添います。

- ソースを追加するとき、匿名化のルールが被面談者の身元を守ります。
- ノートを作るとき、根拠の水準についてのルールが、ソースが示すことと私たちが推し量ることを分けます。
- 成果物を準備するとき、最後の指示が、解決策を既定のものとして早く示しすぎることを防ぎます。

行動を実行するスキルは、まずそれに関わる指示を見分け、そのうえで結果がそれらを守っているかを確かめねばなりません。

依頼

↓

当てはまる指示

↓

スキルの行動

↓

結果の点検

こうして指示は行動を取り囲みます。実行の前にそれを縛り、実行のあとに結果を点検する役に立ちます。指示だけで働くわけではなく、すべてを知っているわけでもありません。しかるべきときにそれを効かせるのはスキルです。

使命はどこへ行くかを示します。指示はコンテキストのひとつひとつの所作に付き添い、すでに決めた用心を私が段階ごとに思い出さずにすむようにします。

すべてを開き直す必要はない

私のディレクトリには何十ものソースとノートがあります。開くたびにそのすべてを読み込めば、優先すべきものが量に沈んでしまいます。ですから `loading.md` が、安定した核を選びます。

```
## つねに

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

## 求めに応じて

- 検証に必要なソース
- 扱っているテーマに結びついたノート
- 作業中の下書き
```

すべてには手が届きますが、すべてが AI の文脈に読み込まれるわけではありません。 `loading.md` は、使命を理解し、依頼を解釈し、すでに定まったルールを失わずに動くために欠かせない要素を指し示します。

- 残りはディスクに置かれたままで、必要なときにだけ参照されます。
- 古いソースは、ある主張を点検するのに欠かせなくとも、毎回のセッションで作業中の文脈を占める必要はありません。

loading.md の中身はコンテキストによって変わります。

- 本なら開くたびに構成案を、プロダクトの探索なら中心となるテーマノートを読み込むでしょう。
- 原則は同じです。
 - ▶ 動けるだけの情報で始め、
 - ▶ 作業がそれを正当化したときに残りを取りに行く。

本当はどこまで来ていたのか

使命は安定したままです。01-state/state.md のほうは、変わるために作られています。三つの塊に収まります。

状況

分かっていること

- 最初の面談では、入力先の送りが細部の喪失を招いている。
- 二つめでは、必須項目の一部がすり抜けられている。

不確かなままのこと

- 入力されたあと、どの項目が実際に再利用されているか分からない。

未済

- ホワイトペーパーは追加したが、信頼度の見極めが残っている。

この状況は、経緯のすべてを語るものでも、ノートのすべてを要約するものでもありません。仕事そのものについて私が持っている、いちばん良い理解を示します。

state.md は AI にも人間にも向けられています。

AI にとっては、セッションをまたいで残る作業記憶です。

- 開くたびに読み込まれ、利用できる確かな情報、残しておくべき不確かな点、忘れてはならない未完了事項を伝えます。
- これがなければ、新しいセッションのたびに、ソースとノートから状況を整理し直さなければなりません。

私にとっては、計器盤の役割を果たします。

- コンテキストが定まったと見なしていること、まだ不確かだと見なしていること、未完のまま残っている作業を、数行で確かめられます。
- 私の理解と食い違っていれば、その場で直せます。

こうして数十行で、AI は確かなことと仮説を混ぜずに仕事を再開でき、私は AI が寄って立つ理解を点検できます。

todo.md はそこから行動を取り出します。

```
## 進行中
- T-001 - 必須項目の実際の使われ方についての面談を準備する。

## これから
- T-002 - どのデータが監査に使われているかを確かめる。

## 済み
- T-003 - 最初の二つの面談を比べる。 - 2026-09-10 に完了
```

todo.md を自分で書き換えるために開くとはかぎりません。 言うだけでもよいのです。

監査にどのデータが使われているかを確かめる作業を追加して。

コンテキストは一行を作り、識別子を与え、正しい節に置きます。作業は仕事の最中に現れることもあります。

- あるソースの分析から必要な確認が浮かび上がれば、AI はそれを会話の中で消えさせる代わりに、追加することを提案します。
- 思いついたことをすべて黙って新しい義務に変えたりはしません。作業が優先順位を変えたり、追加の仕事を引き起こしたりするときは、判断の責任は私にあります。

仕事が済んだら、私はこう頼めます。

作業 T-002 を閉じて。(おおよその説明でもよく、AI が正しい参照を見つけます。)

5

コンテキストはその行を消しません。「済み」へ移し、終了の日付を加え、役に立つときは結果や、それを完了と見なせる根拠を示します。

- 終わった作業は消えません。
- その存在が、同じ仕事を繰り返すことを避けさせ、ある筋がなぜ動かなくなったのかを説明してくれます。

ですから `todo.md` は、一度書いて終わりの一覧ではありません。コンテキストの行動の待ち行列です。AI は何が残っているかを知るためにそれを参照し、仕事が進めば更新し、やり直しを避けるために終わった作業を保ちます。

何から始め直すのか

handoff.md は核の最後のファイルです。保存のたびに書き直され、きわめて具体的な状況に答えます。前の会話を何も知らない誰かが——人であれ AI であれ——いま到着する。間違えずに出発するには、何を知っていなければならないか。

再開

今どこにいるか

三つのソースが、入力のしやすさと活用のあいだの緊張を浮かび上がらせた。

続ける前に知っておくべきこと

二つの面談は個別の状況を述べ、ホワイトペーパーは一般的な見解を掲げている。この集まりは緊張を浮かび上がらせるが、その広がり測ることも、どの組織にも存在すると結論することも、まだできない。

次の一手

実際に使われている項目を見分けられる質問を準備する。

未解決の点

監査に必要な構造の水準は分からないままである。

次の一手はひとつだけで、すぐ実行できます。十の可能性を並べた一覧は、このファイルがまさに私から取り除くために存在している優先順位づけの仕事を、私に返してしまいます。

ですから開くコマンドは、コンテキストが眠っているあいだに置かれたものを取り込み、この核を読み、そしてこの一手を私に告げます。

- 細部より先に目的を取り戻すので、二週間の不在はもう一分の読書で済みます。
- セッションはこれで、向きを失わずに新しい情報を受け取れます。

覚えておくこと — 開くための核

コンテキストを開き直すとは、その中身を読み返すことではなく、小さな核を読み込むことである。使命、指示、現在の状況、そして再開。使命はどこへ行くかを告げ、指示は道中の条件を定める。役に立つ再開は、すぐ実行できるひとつの行動だけを指し示す。

与える——情報を取り込む

セッションが始まり、三つのことが同時に起こります。

- 運用責任者が面談の記録を送ってきます。
- やり取りのさなかに、ひとつの考えが浮かびます。
- 前日に置いておいたメモが、受信箱（メモ管理アプリ、メールなど）で待っています。

この三つの情報は、同じやり方でコンテキストに入るのではありません。それぞれに専用のスキルがあります。

資料は、完成した物として届く

外部から受け取った資料は、元のファイル、処理しやすい形式の本文、追跡用の管理ファイルとともに保存します。

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

SRC00001-entretien-exploitation.md という管理ファイルは、次のような内容です。

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

運用責任者との面談

なぜここにあるか

技術者が活用できる報告書を書けない理由を理解するため。

解釈 — 未分析

責任者は、現場での入力の速さと、分析に必要な構造化データとの対立を述べている。

扱っている範囲

- モバイルでの入力
- 必須項目
- データの再利用

これが生んだもの

- 分析後に記入

この管理ファイルは、必ずしも資料の要約ではありません。 ますます管理上の問いに答えます。

- どこから来たのか。
- なぜ加えたのか。
- 中身を読めるのか。
- どの程度信頼してよいのか。

「解釈 — 未分析」の節は、最初の読み取りを隔離しておきます。取り込みのときに浮かんだ印象が、ただちに確かな知識として扱われるのを防ぎます。

ソースがPDFやウェブページなど、そのままでは扱いにくい形で届いたときは、取り出しの作業が、人にもAIにも読める本文の版を生みます。

- この作業はまだ知識を生みません。元の資料を変えずに、情報に手が届くようにするだけです。

内容の整理や変換は、その後に行います。

- 取り出した本文を分析して、アトミックノート、テーマノート、用語集の項目、あるいは判断の記録を浮かび上がらせませす。
- これらのノートが*アイデアの世界*を養い、そこでソースの情報は、使い回せる知識のブロックの集まりになります。

信頼度は、加えたソースに評価を下すものではありません。

- その情報をどう使うかを示すだけです。孤立した証言、検証済みのデータ、契約上の文書、確認を要する仮説。

役に立つ情報のすべてが、まだ資料として存在するわけではありません。一部は私の経験、記憶、観察、あるいは働きながら言葉にする区別のなかにあります。

AIとのやり取りのセッションでそれを口にすると、それもまたコンテキストに入ります。

- 自動的に確かな事実になるわけではありません。

- 出どころを保ったまま、ほかのソースと突き合わせられるようになります。

ですから情報を与えるとは、資料を取り込むことであると同時に、コンテキストの持ち主が会話にもたらしものを保つことでもあります。

人の知識は、セッションから入る

セッションは、一度で渡される資料ではありません。やり取りを重ねながら少しずつ形になります。03-sources/sessions/SESSION-2026-09-12.md に記録され、こんな形をとります。

```
# 2026年9月12日のセッション
```

```
## 09:41 - AI
```

必須項目は本当に品質を高めているのか、それとも労力を技術者へ移しているだけなのか。

```
## 09:42 - 著者
```

面談を見るかぎり、必須項目が保証しているのは、値が入力されたことだけだ。

その正確さも、その有用さも保証しない。

ある項目が将来どう使われるかを技術者が分からないとき、彼は先へ進むために大急ぎでそれを埋める。

つまり制約は見かけの完全さを高めるが、実際に活用できる情報を良くしないまま、労力を移すだけになりうる。

```
...
```

AI の言うことをすべて保つ必要はありません。その一方で、仕事を変える問いや反論や言い換えは、保つに値します。

- 私の言葉と AI の言葉は分けられたままです。
- そのおかげで、あるアイデアの出どころを見つけ出し、私の意図とシステムの提案を区別できます。

セッションは乱れていてかまいません。

- その役目は、考えの実際の動きを捕まえることであって、完璧な要約を示すことではありません。

受信箱は、コンテキストが閉じているあいだに語りかける

セッションの外、電車の中や会議のさなか、インターネットのないところで、ひとつの考えが浮かびます。

- それは受信箱に入ります。
- この受信箱はコンテキストの根に置かれた `offline.md` というファイルです。
- その Markdown ファイルに書くだけで足ります。

こんな形をとります。

技術者が当たり障りのない値で必須項目をすり抜けていないか尋ねること。

5

次にコンテキストを開いたとき、このメモは読み直され、進行中の仕事と照らし合わされ、しかるべき場所に取り込まれます。作業になることも、面談の質問になることも、ノートになることもあれば、もう的を射ていなければ消えることもあります。

加えることは、まだ理解することではない

この情報を取り込んだあと、ディレクトリには複数のソースと、進行中のセッションの記録が収まっています。

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

これらのファイルは、情報が手元にあることを証明します。それが理解されたことは証明しません。

03-sources/ingestion-log.md というファイルが、この二つの作業を分けます。ソースごとに、分析が完了しているのか、部分的なのか、やり直すべきなのかを、中身の指紋といくつかの処理メモとともに示します。

この区別は、資料が差し替えられたとき、取り出しが変わったとき、セッションが中断したときに欠かせなくなります。コンテキストはそのとき、すべてを知っているふりをせずに分析を再開できます。

情報は入りました。あとは、結びつけ、論じ、使い回せるだけの精度を持ったアイデアに変えることが残っています。

覚えておくこと——ソース

資料、セッションでの言葉、セッション外のメモは、三つの違う扉から入り、それぞれ元の形を保つ。ソースは情報と根拠を保つ。そこから引き出される解釈は決して保たない。加えることは理解することではない。取り込みの記録が、存在するものと消化されたものを分ける。

変える――ソースからアイデアへ

いま私の前には三つのソースがあります。

- 二つの面談が、フォームの引き起こす摩擦を語っています。
- 一冊のホワイトペーパーが、構造化データの必要を説いています。

すぐに断を下して、必須項目は品質を損なうと書きたくなるかもしれませんが。それは急ぎすぎです。コンテキストは、これらのもたらしたものを、ひとつの結論に溶かし込まずに保たねばなりません。

そこで私は、ノートを作り更新するスキルを呼びます。まだ処理されていないものを読み、そこからアイデアを引き出します。

ひとりで立てるアイデアを取り出す

ソースに、そのまま使い回せるアイデアが入っていることはめったにありません。

- 面談には、事実も、例も、意見も、ためらいも、その状況ならではの細部も混ざっています。
- 変えるとは、すべてを要約することではなく、そこからアイデアを取り出すことです。

その道のりは三段です。

証言	「フォームが長すぎると、技術者のなかには終わらせるために 当たり障りのない値を入れる者がいます。」
↓	
観察	入力を速めるために、当たり障りのない値が ときどき使われている。
↓	
安定したアイデア	入力の制約が強すぎると、 報告書は瘦せうる。

強いアイデアとは、人目を引く言い回しでも、決定的な真理でもありません。

- 重心がはっきりしていて、強い。
- それを生んだ文章から離れても理解できる。
- 裏づけたり、含みを足したり、覆したりできるだけの輪郭がある。
- 出どころと限界も保っていて、ひとつの証言を一般的な規則に変えてしまわない。

AI は使命と指示に照らしてソースを読み、すでにあるノートと突き合わせます。アイデアが新しければ新しいノートを作り、そうでなければ、追加の含みや限界で既存のノートを豊かにします。

この作業は、ひとつひとつの言い回しに私の承認を求めずに行われます。私はあとでノートを読み直し、直せます。

- セッションのさなかに曖昧さが現れたときは、AI が私に尋ねます。
- あとになって自分では解けない矛盾を見つけたときは、私の代わりに断を下さず、それを保ちます。

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

```
---
```

```
id: AT000001  
status: active  
sources: [SRC00001, SRC00002]
```

```
---
```

入力のコ約が強すぎると報告書は瘦せうる

主なアイデア

フォームが作業を遅くすると、技術者は当たり障りのない値を入れたり、記述を求められる最低限まで削ったりしうる。

なぜ重要か

形のうえで構造化されたデータは、きれいに見えながら、現場の実際をうまく述べていないことがある。

含みと限界

二つの面談では、この現象の頻度を測れない。

出どころ／根拠

- SRC00001: 早く終わらせるために使われた、当たり障りのない値の例。
- SRC00002: 必須項目を増やしたあとに見られた精度の低下。

リンク

- AT000002と対照。

見出しはテーマではなく、ひとつの命題を担っています。ノートは、出どころの資料を開き直さなくても理解できるままで、同時にその出どころを確かめるための結びつきを保っています。

アトミックノートは、最初に書いた形で固まるわけではありません。

- その同一性を失わずに、含みや、新しい根拠や、関係を加えられます。

ホワイトペーパーは二つめのノートを生みます。比較し、集計し、自動化するには構造化データが必要だ、というものです。

- このアイデアは別のことを述べているので、最初のものとは切り離されたままです。

早すぎる決着を拒む

二つのアイデアは対立して見えますが、どちらを取るかを決めるだけの材料はまだありません。ですから私は矛盾をひとつ開きます。

C-001 - 入力を損なわずに構造化する

- 一方では、課された項目が現場の記述を瘦せさせうる。
- 他方では、規模をもった活用には比較できるデータが要る。
- 論点：どの情報を入力の時点で構造化すべきで、どれはあとからでよいのかを定めること。
- 開いた日：2026-09-11

矛盾は、対になった二つの誤りから私を守ります。早すぎる段階でどちらかの陣営を選ぶことと、誰の気にも障らない曖昧な言い回しで不一致をならしてしまうことです。

- 決定、新しいソース、あるいは記述の修正によって解消されるまで、矛盾は未解決のままです。

どんな不確かさもこの扱いに値するわけではありません。

- 私はこの扱いを、その後の検討や意思決定の質に大きく影響する対立に限っています。

これらのアイデアが合わさって語ることを結ぶ

取り込みのあいだ、AI はアトミックノートを作るだけではありません。

- 取り出したアイデアと、すでにあるノートとの関係も調べます。
- そのうちのいくつかは、ひとつの命題には縮められないまとまりを作るとき、同じ作業のなかでテーマノートを作り、それらを結ぶリンクを置きます。

私が二度目の指示を出す必要はありません。

- アトミックノートの作成、その照らし合わせ、そして場合によってはテーマノートの出現が、ひとつの取り込みの一部です。

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

品質は入力と活用の双方に左右される

切り口

報告書の品質は、埋まった項目の数だけでは評価できない。

まとめ

システムは、入力を形だけのものにも高くつくものにもせずに、データを活用できるだけの構造を得なければならない。

結ばれたノート

- AT000001 - 入力の制約は報告書を瘦せさせうる。
- AT000002 - 活用には構造化データが必要である。

緊張

- C-001 - 入力を損なわずに構造化する。

テーマノートは、要約を並べるだけではありません。その関係をどう読むべきか、何がまだ未解決なのかを示します。

ある語が、みなに同じ意味で通じていないと気づく

不一致の一部は、言葉から来ることがあります。そのとき用語集が、このコンテキストで採る意味を定めます。

構造化データ

比較や自動処理を可能にする明示的な形で記録された情報。

次と混同しないこと：作業の時点で利用者が必ず埋めねばならない項目。

例：故障の種類、停止の長さ、対象の設備。

この局所的な定義は、普遍的なものになろうとはしていません。

- 二つのノート、あるいは二人の人が、同じ語で別のことを語るのを防ぐだけです。

成果物に対して中立なブロック

ノートが*アイデアの世界*に属するのは、読者に効果を及ぼすことより知識を安定させることを求めているあいだです。

- ですからノートは、記事のつかみ、スライド、営業の論拠、仕様書の一部に、早くからなってしまうはいけません。

この中立さは、ノートが曖昧なままだという意味ではありません。

- むしろ見出しは明確な命題を主張し、中身はその含みを示し、リンクはその土台を点検させます。
- 中立なのはその形です。ひとつの結果に奉仕するには組まれていません。

ですから AT000001 は、面談の手引きではすり抜けについての質問になり、枠組みの覚書では危険になり、チーム向けの発表では例になりえます。AT000002 は、データの使われ方についての質問を支え、プロダクト上の要求を正当化し、なぜ自由記述だけでは足りないのかをチームに説明できます。

テーマノートは、これらの要素をひとつの決定版の文章にまとめてしまうものではありません。

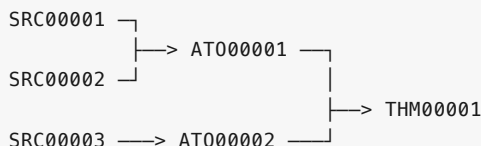
- 合わせて読めば何が理解できるのかを示し、成果物によって扱いを変えねばならない緊張を保ちます。

この中間の層が、作るたびに分析をやり直すことを避けさせます。

- ソースは根拠であり続け、
- ノートは利用できる知識になり、
- それぞれの成果物が、自分の必要に応じてそれらを組み合わせます。

どのアイデアからも、その根拠までさかのぼれる

この段階で、関係はこう表せます。



名前のついたリンクが、あるアイデアを裏づけるソース、含みを足すソース、覆すソースを見分けさせます。

- 信頼度が、複数の観察に支えられているものと、仮説にとどまるものを見えるようにします。

取り込みの網羅度は、どのソースが実際にノートに寄与したのかを示します。ディレクトリにあるというだけの資料を、消化済みと見なすことを防いでくれます。

これらのノートは、特定の成果物のために書かれてはいません。

- 共通の知識を形づくり、写し取られることも、どれかひとつに閉じ込められることもなく、複数の結果を養えます。

***アイデアの世界*とは、まさにこの使い回せる資産です。見つけ出せるだけ安定していて、組み替えられるだけ独立した、ブロックの集まりです。**

覚えておくこと——アトミックノート

安定したアイデアは、テーマではなく命題を担い、出どころの資料を開き直さずに理解できる。矛盾は、筋道に重くのしかかるあいだ保たれる。早すぎる決着も、ならすこともしない。どのノートもソースへのリンクを保つ。だからアイデアからその根拠へさかのぼれる。

生み出して決める

私のテーマノートは、はっきりした問いを浮かび上がらせました。作業の終わりに入力すべき情報はどれで、あとから補えるのはどれか。

いま三人が、私から何かを待っています。

- 話を聞くべき技術者。
- 決定を下す委員会。
- 説明すべきチーム。

三つの調査をするものではありません。同じノートから三度汲むのです。

ここはセッションのなかで、どのスキルも助けてくれない唯一の場面です。成果物を生み出すのは書く仕事であって、繰り返される所作ではありません。

同じ知識を三つの成果物に生かす

ノートから成果物への移行は、写しではありません。それぞれの結果が、自分に役立つブロックを選び、並べ、受け取る相手に合わせて言い直します。

共通のブロック	面談の手引き	枠組みの覚書	チーム向けの発表
AT000001 — 入力の制約が強すぎると報告書は瘦せうる	どの項目がすり抜けられ、どうすり抜けられるかを尋ねる	品質が失われる危険を述べる	現場の摩擦の例を示す
AT000002 — 活用には構造化データが必要である	どのデータが実際に再利用されているかを尋ねる	比較と自動化の必要を立てる	チームにとっての価値を説明する
THM00001 — 品質は入力と活用の双方に左右される	二つの要請を軸に面談を構成する	論点と判断の基準を言葉にする	語りの筋をつくる
C-001 — 入力を損なわずに構造化する	緊張を、確かめるべき問いに変える	解くべき問題として保つ	早すぎる解決策を告げずに見えるようにする

こうして同じ知識から三つの成果物が生まれます。どの成果物も知識のすべてを使うわけではなく、土台となったノートに取って代わるものでもありません。

この分離はとりわけ、知識を一度だけ直せばよいようにしてくれます。

- 新しいソースが AT000001 に含みを足したなら、次からの成果物はその豊かになったブロックから出発します。いくつもの資料に散らばった古い言い回しを再生産することはありません。

いちばん急ぐものから始める——次の面談

私の最初の必要は、技術者との面談を準備することです。成果物は下書きの場所から始まります。

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

次の面談のための質問

1. 作業のあいだ、もっとも手を止めさせる項目はどれですか。
2. 求められた情報が手元にないとき、何を入力しますか。
3. 作業のあとで足せる情報はどれですか。
4. 活用に役立つために、すぐ正確でなければならないデータはどれですか。

ノートは、使い回せるアイデアを保とうとします。成果物のほうは、受け取る相手と、形と、期待される行動を選びます。

読み直して承認されると、ファイルは draft/ を離れ、そのまま 05-outputs/ に入ります。この移動が示すのは成熟の度合いであって、文書の種類の違いではありません。

ここまで私は生み出しただけです。すでに知っていたことを、三つの読者のために形にしました。この章の続きは、まったく別の所作にかかわります。あるときには、示すことをやめて断を下さねばなりません。そして断を下したことは、六か月後に見つけ出せなければなりません。

判断の記録は、結果を形づくる選択を保つ

「仕様書を書く前に技術者に話を聞く」は、何を納めるかについての判断ではありません。

- この面談が単に次の一手なら、それは `01-state/todo.md` に属します。
- 探索の仕事すべてがこのルールを守るべきなら、それは `00-frame/directives.md` の指示になります。

どちらの場合も、判断の記録を正当化しません。

判断の記録が現れるのは、仕事がありうる複数の方向を浮かび上げらせ、製品や文章や手法、あるいはコンテキストの別の結果を形づくる方向を選ばねばならないときです。そのとき記録は、何を設計し何を納めると決めたのか、退けた選択肢、選んだ理由、受け入れた帰結を保ちます。

ですから私の例では、判断の対象は面談の進め方ではありません。機能上の決定です。

- 作業の終わりにすべての情報を必須にするのか、
- 自由記述だけを保つのか、
- それとも報告書をあとから補えるようにするのか。

役に立つ判断の記録は、選択の核心を、それを実装する人に委ねたりしません。三人が読んだら、同じ範囲、同じ期待される振る舞い、同じ限界を理解しなければなりません。構造に影響しない技術的な細部は未確定でもかまいませんが、そのどれも決定を暗黙のうちに変えるものであってはなりません。

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

書き上がるとこうなります。雛形が長いのは、決定を理解でき、見直せるものになっている筋道をすべて保つからです。判断の記録にすでに慣れた読者は、そのあとの解説へ飛んでかまいません。

```
---
uuid: 0d5efbe2-1eac-4ad7-94c1-29f68adcf001
id: DRE00001
created: 2026-09-11
updated: 2026-09-11
status: accepted
decided_by: プロダクトチーム
decided_on: 2026-09-11
supersedes: null
superseded_by: null
sources: [AT000001, AT000002, THM00001]
---
```

作業の終了を四つの必須情報に限る

決定

作業が終了したと見なされるには、ちょうど四つの情報がそろっていなければならない。

1. 対象の設備。作業票にすでに記されているか、終了の時点で入力される。
2. 作業が実際に終わった日付と時刻。
3. 作業終了時の設備の状態。復旧、能力を下げた状態での稼働継続、または停止のまま。
4. 行った作業の、または設備を復旧できなかった理由の、空でない要約。

ほかのどの情報も終了の条件にはならない。

終了のあと、報告書には、故障の種類、特定された原因、行った作業の内容、交換した部品、設備の停止時間を書き足せる。作業の終了がそれで覆ることはない。技術者がこれらの補足情報を入力してもよいが、運用責任者が二十四時間以内にそれを補うか承認しなければならない。

まだ不完全な報告書も参照できる。

「要記入」と印され、二十四時間後に運用責任者へ通知が送られる。

背景

技術者は、分析用の長いフォームをたどらずに、現場で作業を終えられなければならない。一方で運用部門には、故障とその原因、行った作業、交換した部品、停止時間を分析するために、比較できる情報が必要である。

前提

- 作業票は通常、作業の開始前に設備を特定している。
- 技術者は設備の状態を知っており、終了の時点で行った作業を短く述べられる。
- 運用責任者には、二十四時間以内に完全な報告書を承認するための時間と情報がある。

選択肢

- 終了の時点で九つの情報をすべて求める。
- 自由記述の要約だけを保ち、あとから作業の情報を自動で取り出す。
- すべての情報を任意にし、品質は事後にのみ点検する。

理由

ただちに得られる四つの情報で、作業を特定し、終了を日付け、設備の状態を知り、行った作業を理解するには足りる。

分析のための五つの情報はなお必要だが、技術者が現場で仕事を終わることを妨げてはならない。

二段構えにすることで、報告書を補うか承認する責任が、運用責任者にはっきり割り当てられる。

土台

- AT000001は、入力のコストが高すぎることによるすり抜けを述べている。
- AT000002は、比較できるデータの必要を打ち立てている。
- THM00001は、この二つの要請を結んでいる。

帰結

良い点

- 終了に必要なのは、ただちに分かる四つの情報だけになる。

- 作業内容の分析に必要な情報は、報告書が補われたあとも手元に残る。
- 欠けているデータについての責任が見える。

受け入れた不利益

- 作業に関する指標は、作業から二十四時間のあいだ不完全なままでありうる。
- 運用責任者に、承認の負担が増える。

危険

- 不完全な報告書が積み上がりうる。
- あとから足された情報は、現場で確かめた状況を忠実に映さないことがある。

適合の基準

- 四つの即時情報のいずれかが欠けていれば、作業を終了できない。
- 五つの補足情報がひとつもなくとも、作業は終了できる。
- 補足情報は、作業の終了状態を変えずに書き換えられる。
- 報告書は、どの使われ方においても「要記入」または「完全・承認済み」として示される。
- 終了から二十四時間後に報告書が完全かつ承認済みでなければ、運用責任者へ通知が送られる。

関連する決定

- C-001「入力を損なわずに構造化する」を解決する。

ソース

- SRC00001
- SRC00002
- SRC00003

ここでの `accepted` という状態は、判断が承認され、実装を導けるだけの精度を持っていることを示します。

- 求める情報、責任の所在、適合の基準が定まらないうちは、判断の記録は `proposed` のままでなければなりません。

- いったん受け入れられたら、歴史の痕跡として固定され、それ以上黙って変わることはありません。

新しい決定がそれにとって代わっても、古い判断の記録は参照できるままで、二つのファイルは `supersedes` と `superseded_by` で結ばれます。

この連なりが、判断の時間の流れを丸ごと保ちます。何が決まっていたのか、どんな情報に基づいていたのか、そしてどの要素が別の選択へ導いたのか。

ですから六か月後、私は現在のルールだけを見つけるのではありません。なぜ最初の方針が正しいと思えたのか、何が不足していたのか、次の決定にどう至ったのかを理解できます。

この時間の流れがなければ、選択の連なりはやがて支離滅裂に見えてきます。それがあれば、変わり続ける製品の、理解できる歴史に戻ります。

- 選択肢は、何が退けられたかを示します。
- 受け入れた帰結は、何が犠牲にされたかを見えるようにします。
- 土台は、決定を、それを支える知識と根拠に結びつけます。

生み出しても、知っていることは尽きない

面談の手引き、テーマノート、判断の記録は、同じ問題を扱いながら重複しません。三つの違う役割を果たしているからです。

- ノートは、コンテキストが知っていると考えていることを保ちます。
- 決定は、ひとつの選択とその理由を保ちます。

-
- 手引きは、その知識を、特定の相手に向けた行動に変えます。別の成果物が、同じ共通の知識を違う形で取り上げられます。**成果物を生み出しても、コンテキストの中身は失われません。むしろ、蓄えた情報を何に活用できるのかが明らかになります。**あとは、到達した状況を記録します。次のセッションで、これまでの経緯を整理し直さずに再開できるようにするためです。
-

覚えておくこと——判断の記録

同じノートが複数の成果物を養い、どの成果物も、使った知識を消費しない。判断の記録が保つのは、何を設計し何を納めるかを形づくる判断であって、行うべき作業ではない。退けたものと犠牲にしたものを保つ。だから選択は見直せるものであり続ける。

保存して再開する

判断の記録が受け入れられたばかりで、三分後に会議が始まります。調査も終わっていなければ、終わりらしい区切りにも達していません。

これが普通の場合です。別の仕事が注意を求め、機械を再起動せねばならず、パソコンを替え、あるいは AI の文脈の窓がいっぱいになります。

- ですから保存は、「調査が終わった」というような、仕事そのものの必要に応えるものではありません。
- 技術上の保証です。コンテキストの状況は、それを生んだセッションより長く生き延びねばなりません。

私はいま中断でき、あとで開き直し、置いたとおりの場所で仕事を見つけられなければなりません。

ひとつのスキルがそれを担い、数秒で終わらねばなりません。高くつく保存は、急いでいるとき——つまりそれがいちばん役立つとき——に実行されないからです。

AI は、その瞬間に分かっていることを固定する

01-state/state.md は、先を方向づけるのに十分なだけ安定したものをまとめます。

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

保存の瞬間、それはこんな形をとります。

現在の状況

安定した知識

- 入力のコストが高すぎると、報告書は薄くなる。
- とはいえ、規模をもった活用には構造化データが要る。

定まった決定

- 終了には、対象の設備、終了の日付と時刻、設備の状態、行った作業の要約が要る。
- 故障の種類と原因、作業の内容、交換した部品、停止時間は終了後に補われ、二十四時間以内に運用責任者が承認する。

主な不確かさ

- この分け方が、データの品質を損なわずに本当に終了までの時間を縮めるのかを測ること。

知識の未済

- 終了までの時間も、二十四時間後に完全である報告書の割合も、まだどの試行でも測られていない。

- 終わった作業は、やることの一覧から消えてかまいません。
- 解けていない矛盾は、状況のなかに見えたまま残ります。

統合された知見には、仕事が十分に安定させたものだけが入ります。状況は現在を述べるために書き直され、統合された知見は、得られた理解を保つために少しずつ豊かになります。

未来の自分のために、ひとつの行動だけを用意する

再開のファイルは、経緯のすべてを語りません。次に役立つ所作を用意します。

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

01-state/handoff.md はこんな形をとります。

```
# 再開  
  
## 現在の状況  
  
DRE00001は受け入れられ、C-001は解決した。  
終了の四つの情報と、五つの補足情報が定まっている。  
  
## 注意すべき点  
  
DRE00001に取って代わる新しい判断の記録を開かずに、終了の必須情報を増やさないこと。  
  
## 次の一手  
  
DRE00001の実装を始める。  
  
## 未解決の点  
  
- 作業後の設備の三つの状態について、技術者に示す表示名を定める。  
- 終了までの時間と、二十四時間後に完全である報告書の割合を測る準備をする。
```

役に立つ再開は、次の行動をひとつだけ指し示します。十の可能性を並べた一覧は、私がいたがた済ませた決定の仕事を、次のセッションにやり直させてしまいます。

再開どうしを比べられるよう、記録を残す

01-state/checkpoints.md は、状況の圧縮された写真の連なりを保ちます。

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

ファイルはこんな形をとります。

```
## 2026-09-11

- ソース：3
- アトミックノート：2
- テーマノート：1
- 開いている矛盾：0
- 受け入れられた判断の記録：1
- 次の一手：DRE00001の実装を始める。
```

```
## 2026-09-04

- ソース：1
- アトミックノート：1
- テーマノート：0
- 開いている矛盾：0
- 次の一手：運用側の見解を得る。
```

5

これらの写真は、すべてのファイルを比べなくてもコンテキストの変化を見せてくれます。停滞や、気づかないうちに膨らんでいく未済にも気づかせてくれます。

AI は、起きたことを書き留める。どう思ったかではなく履歴は、やり取りを一つひとつ再現しようとせずに、重要な出来事を記録します。私の例では、その日の出来事が 02-history/2026-09-11.md に加えられます。

```
contexte-qualite-comptes-rendus/
└─ 02-history/
   └─ 2026-09-11.md
```

加えられる塊はこんな形をとります。

```
## 2026-09-11
```

- SRC00003-livre-blanc-donnees-intervention を追加。
- AT000002 と THM00001 を作成。
- C-001 を開く。
- `questions-prochain-entretien.md` を承認。
- DRE00001 を受け入れ、C-001 を解決。

- 状況は、いまどこにいるかを告げます。
- 履歴は、何が起きたかを告げます。
- 再開は、次に何をするかを告げます。

自分では見えないものを、機械に点検させる

保存のときに、いくつかの機械的な点検が見落としを減らします。その結果は 01-state/health.md に書かれます。

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

- 参照されているファイルはすべて存在するか。
- 識別子は一意か。
- どのノートもソースを示しているか。
- 加えられたが分析されていないソースに印がついているか。
- 置き換えられた決定は、置き換えたものに結ばれているか。
- 次の行動ははっきり書かれているか。

これらの点検が私の判断に取って代わるわけではありません。構造上の誤りが、考えの問題のふりをするを防ぐだけです。

一連の流れと、そこから得られるもの

全体の動きが、ここで見えるようになります。

開く
↓
受信箱を読む
↓
使命、状況、再開を読み込む
↓
ソースを追加する
↓
ノートを作る、または更新する
↓
下書きを生み出す
↓
点検、承認、または決定
↓
状況、統合された知見、再開を更新する
↓
チェックポイントと履歴
↓
保存の点検
↓
次に開くとき

この連なりの終わりを待って保存するのではありません。

- ソースを加えたあと、ノートを作っている途中、決定の前、成果物を準備しているあいだ、文脈の窓がはち切れそうだと見えたときに保存します。

まさにそれが保存の役割です。**この一つひとつの状況を、進行中の会話とは無関係に再開できるものにすること。**

これで私の会議は始められます。

コンテキストはファイルの木であるだけではありません。長い時間のなかで考えるためのシステムであり、情報を取り込み、緊張を

保ち、アイデアを安定させ、複数の結果を生み、再開を準備します。

その形はあえて単純で持ち運べるままです。読めるファイルがあれば、仕事は見え、確かめられ、受け渡せるようになります。

この機械を理解したところで、別の問いが現れます。コンテキストは、その外に存在する道具や知識ベースや実際の製品と、どう対話するのか。

覚えておくこと—保存

保存は段階の終わりを印すものではない。状況が、それを生んだセッションより長く生き延びることの保証である。状況はいまどこにいるかを、履歴は何が起きたかを、再開は次に何をするかを告げる。ひとめぐりはどの時点でも中断できる。そしてそのどの時点からも再開できる。

部 6

コンテキストとその生態系

よく手入れされたコンテキストは、手に入る知識をすべてそこに入れたくさせます。

この部では、コンテキストが外に置くものと、なぜそこに置くのかを見ます。考える力を保つために小さいままでいます。そして、念のために抱え込んだものは、原本がよそで変わり続けるあいだに、知らせもなく古びていきます。ですから隣人とともに働きます。保存すること、測ること、実行することを仕事とする道具と、自分とは別の問いを調べているほかのコンテキストです。

構造化された知識に問いかける

作業報告書は待ってもらいましょう。今朝は同じ製品の別のテーマ、登録の途中で離脱する人たちに取り組んでいます。ある調査会社が調べを届けてくれたので、コンテキストに加えます。複数の参加者が、パスワードの規則を前にためらっている、と。この観察は、以前に出会ったあるアイデアを思い出させます。製品の価値を感じる前に大きな労力を求めると、利用者は去りうる、というものです。

- そのテーマについての材料を持っているとほぼ確信しています。
- ただ、それがどこにあるのか、どんな言葉で書いたのかが思い出せません。

初期の労力と書いたのか、摩擦と書いたのか、アクティベーションか、オンボーディングか。そのアイデアはおそらく、

- ある本から取り出され、
- プロダクトマネジメントのノートで展開され、
- 別の利用者の道筋に結びつけられ、
-

いくつかのキーワードで検索して、結果をひとつずつ開くこともできるでしょう。その代わりに、私はコンテキストに尋ねます。

利用者が製品の価値を感じる前に求める労力について、私たちはすでに何を学んだか。

コンテキストは、より広い記憶に問いかける

答えは、いま開いているコンテキストの中にはありません。その一方で、私の LLM Wikiの中にはあるかもしれません。本や記事や規格やほかの資料から取り出された何千ものノートが、すでに構造化されている共通の書庫です。

LLM Wiki とコンテキストは、同じ性質のものではありません。

ウィキは知識ベースです。

- ほかに人が書いたものから、構造化されたノートを保ちます（私の場合はアトミックノートの考え方を再利用しました）。定義、原理、調査の結果、規格の要求。
- どのノートも、何が分かっているのか、それがどこから来たのかを告げます。
- 何万もの数になりえます。何年もかけて、互いに関係のないテーマにわたって積み上がったものです。

どんなコンテキストも、この量を抱えることはありません。それはその役目ではないのです。

けれどもウィキはそこで止まります。

- その知識が、今日私が進めている仕事でなぜ重要なのかも、私がそれをどう使ったのかも告げません。
- アイデアを決定に結びつけることも、決定をそれが私に払わせたものに結びつけることもしません。

LLM Wiki は考えるためのシステムではありません。大量に保ちますが、筋道を立てません。

筋道はコンテキストの中に生きています。 どうしてそうなのかという理由を保つのはコンテキストです。

- 使命は何か、
- どんな指示と決定が定まっているか、
- なぜこのアイデアがこの使命のために選ばれたのか、
- 私は何に対して断を下したのか、
- 何を失うことを受け入れたのか、
- どの矛盾が未解決なのか。

それができるのは、小さいままでいるからにほかなりません。人間が読み返せる数十のノートです。

どちらにも、その強みと引き換えになる弱点があります。

- ウィキは規模を得て、筋道を失います。
- コンテキストは筋道を保ち、規模をあきらめます。

だからこそ、融合するのではなく、語り合わねばならないのです。

探しているアイデアを言葉で述べる

登録に捧げたコンテキストに、ウィキをまるごと写したりはしません。 RAG を走らせ、そのあとグラフを検索する、ということもしません。

私はただ、コンテキストに問いを立てます。

利用者が製品の価値を感じる前に求める労力について、私たちはすでに何を学んだか。

自分の言葉でそれを言います。

- ノートの見出しも、その置き場所も、書かれたときに使われた語彙も、知っている必要はありません。

コンテキストはこの問いを、API を通じて LLM Wiki に渡します。残りはすべてウィキの内側で起こります。

ウィキは意味で検索し、関連するノートへ広げる

LLM Wiki はまず、私の問いに意味の近いノートを見つけ出します。

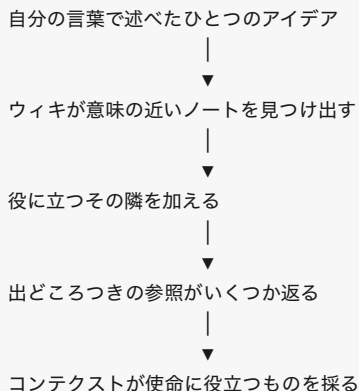
- あるノートは「労力」という語を使わずに、初期の摩擦やアクティベーションや最初の価値までの時間を語っているかもしれません。

RAG は、同じ語彙が使われていなくても、私のアイデアに近い既存の知識を見つけます。

次にウィキは、もっとも良い結果のまわりを見ます。その意味のグラフが、ノートのあいだにすでに置かれた関係をたどり、テーマを照らしうる隣のノートを加えます。

- 定義、
- 限界、
- 矛盾、
- 補い合うアイデア。

これらの隣のノートのひとつは、私の問いに直接答えることは決してなかったかもしれませんが。それでも貴重になります。そのアイデアがどんな条件で当てはまるのか、どこで真でなくなるのかを示してくれるからです。



RAG は、私が名づけられなかったものを見つけ出します。グラフは、私が探そうと思いつかなかったものを浮かび上がらせます。

6

私にとって、この二つの作業はひとつです。ある問題に答える参照を求めれば、LLM Wiki がアイデアとその隣を探し、短い一覧を返してくれます。

コンテキストは、出来合いの答えではなく参照を受け取る

ウィキは、既存のノートを、その中身と出どころと関係とともに返します。複数の結果を黙って混ぜ合わせた要約を作ったりはしません。

ですからコンテキストは、ひとつひとつの命題を読み、登録についての観察と比べ、根拠を点検すべきときはその出どころに戻れます。

- 製品に触れる前に課される労力を、たしかに扱っているノートもあります。
- 利用者がすでに価値を知ったあとに求められる労力を語るノートもあります。

私の問いに近いというだけでは、それらを同じものにしません。コンテキストは自分の使命をもとに選り分けます。次のことができます。

- 登録中の離脱を照らすアイデアを採る。
- 矛盾を保つ。
- 一般的すぎるノートを退ける。

採られたノートは、出どころと、それが寄って立つソースとともにコンテキストに入ります。残りはウィキの中にとどまります。

こうして私は、ひとつひとつのコンテキストを書庫全体の写しに変えることなく、はるかに広い記憶の恩恵を受けます。

- その記憶がどう索引づけされているかを学ぶ必要はありません。探しているアイデアを言葉で述べ、返ってきた参照を確かめるだけです。

この連なりは、すでに構造化された知識への入り口を与えます。

- 製品が今日実際に何をしているのかは、まだ告げません。

その別の問いに答えるには、コンテキストは書庫を離れ、コードと試験と動いている製品に直接問いかねばなりません。

覚えておくこと——二つのシステム、ひとつの扉

LLM Wiki は規模を得て筋道を失う。コンテキストは筋道を保ち、規模をあきらめる。私は自分の言葉でアイデアを述べる。ウィキは意味の近いノートを見つけ出し、役に立つその隣を加える。コンテキストは出どころつきの参照をいくつか受け取り、自分の使命に役立つものだけを採る。

実際の製品に問いかける

登録中の離脱についての調査を読み返していると、ひとつの問いが届きます。

招待された利用者も、ほかの人と同じようにパスワードを作らねばならないのか。

問いはごく小さく見えます。それでもその答えを、顧客に返事をせねばならないサポートの担当者が待っています。

私はこうすることもできます。

- 読むのを中断し、
- 試験用の環境を開き、
- 二つのアカウントを用意し、
- それぞれの道筋をたどり直し、
- 何が起きるかを比べる。

以前はそうしていました。

- 三十分後には、おそらく答えが出ているでしょう。
- 同時に、自分の仕事の筋も見失っていたでしょう。

今回、私はコンテキストに問いを立てます。

コンテキストが製品を見に行く

コンテキストは利用者調査から生まれたノートとともに働き、その主張のソースとしてその調査を保っています。

- 進行中の使命も担っています。

- なぜその問いが重要なのかは分かりますが、それだけでは製品のいまの振る舞いを知ることはできません。

私はもう、その振る舞いを長く述べ続けるための仕様書を保守していません。

- 製品は変わるのが速すぎて、脇に書かれた資料がその忠実な写しであり続けられません。

最新の真実は、納められて本番で動いているもの、つまりアプリケーションのソースコードだけです。

私のシステムでは、コンテキストの外にあるスキルが、ソースコードに問いかけ、逆引きの文書を作れます。コンテキストは、製品についての問いに答えるスキルを呼び、私の求めを渡します。

プロダクトマネージャーの問い



コンテキスト



呼ぶ

製品を読むスキル



問いかける

実際の製品

スキルはまず、招待の道筋をコードの中に見つけ出します。そのうえで、その道筋をアカウント作成の規則まで追います。

コードは、組まれたとおりの振る舞いを示します。ある試験が、招待された利用者が受け取ったリンクから、ただちにパスワードを作らずに登録を終えられることを確かめます。設定は、こ

の道筋について特定の顧客向けの変種が有効になっていないことを示します。

スキルは機能上の答えを返します。

招待された利用者は、この段階でパスワードを作らずに、招待リンクから自分のアカウントに入れます。そのあとで作ることを選べば、通常の規則が当てはまります。試験が、招待の道筋と、任意でのパスワード作成を確かめています。パスワードを忘れた場合の回復の道筋は調べていません。

最初の二文はサポートに渡せます。スキルが示した参照を開いて、開発者とその分析を点検することもできます。

そしてこの答えを、根拠と、検証の日付と、告げられた限界とともにコンテキストに収められます。

- サポートに答えることは、ひとつの場合を片づけてそこで終わります。
- コンテキストに蓄えることは、製品について検証された事実を、使命の*アイデアの世界*に入れることです。招待の道筋についての次の問いはゼロから始まりませんし、調査から生まれたノートは、観察だけでは得られなかったものを得ます。どの道筋が、どの瞬間にパスワードを求めるのか、という正確さです。

答えは、それに添えられた根拠によって値打ちを持つ

スキルは、もっともらしい答えを出して済ませたりしませんでした。主張のひとつひとつについて、何を参照したかを示します。

- コードの中にある規則、

- その振る舞いを確かめる試験、
- 有効な変種を定める設定。

この痕跡が、自信たっぷりでありながら誤った答えから守ってくれます。足りないものも、ただちに見えるようにします。

ここでスキルは、忘れたパスワードの回復を確かめていません。読んでいない道筋へ結論を広げる代わりに、そう告げます。

問いによっては、ほかの要素が必要になります。

- コードはありうる規則を含みますが、どれが有効かは機能フラグが示します。
- データベースの構造は製品が何を記録できるかを示し、許されたデータは特定のアカウントの状況を理解させてくれます。
- Git の履歴とプルリクエストは、ある規則がいつ変わったのか、その変更とともに残された説明は何かを見つけるのを助けます。
-

スキルは、立てられた問いに答えるのに必要なものだけを集めます。リポジトリをまるごと吸い上げたり、どのファイルもコンテキストのソースに変えたりはしません。

答えは、サポートにも、営業にも、プロダクトマネージャーにも読めるままです。根拠は、それを確かめる人や、調べを続ける人のために手元に残ります。

規則を見つけることは、それを変えると決めることではない
私は登録についての調査に戻ります。二つの道筋がパスワードの作成を同じ瞬間に示さないことは、すでにそこから学んでいました。

スキルは、調査には見えなかったことを加えます。招待された利用者は、パスワードの規則をすり抜けているのではありません。あとでそれに出会い、そのときの規則はまったく同じなのです。

ですから彼にとって摩擦は消えてはおらず、場所を変えただけです。この精度が、私の問いを動かします。誰がその規則を被るのかではなく、それぞれがいつそれに出会うのかを問うようになります。製品の価値を感じる前か、あとか。

コードは、製品が何をしているのかを教えてくださいました。

- そのルールが適切か、本当にアカウントを守っているか、利用者が負う不便を許容できるかまでは教えてくれません。

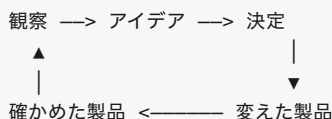
決めるには、コンテキストのほかの要素がなお必要です。

- 調査の観察と、その及ぶ範囲。
- 安全についての目標。
- この規則を採るに至った理由。
- 変えたときに見込まれる帰結。

スキルは製品についての事実を打ち立てます。コンテキストはその事実を、すでにあるノートと決定に結びつけ、同時にその出どころを点検させるソースを保ちます。 **判断の責任は私にあり続けます。**

パスワードの作成を変えると決めるなら、判断の記録がその選択と、その及ぶ範囲と、それを支える理由を保ちます。実装のあとで、スキルは製品に戻り、新しい振る舞いが決定どおりかを確かめられます。

ですからこの動きは双方向に働きます。製品が思考を照らし、そして決定が製品を変えます。



「なぜ」は残し、「何をしているか」は現物から確認する

コード中心の考え方

コードはもう、開発の最終成果物であるだけではありません。製品の現在の振る舞いを、そこから確認し直せる情報源になります。コンテキストは、コードだけでは説明できないものを保ちます。ソース、筋道、決定、退けた選択肢、そして選んだ理由です。

この分け方が、文書の居場所を変えます。

製品がなぜこの方向を取ったのかを説明するものは保ちます。面談、制約、仮説、判断です。現在の振る舞いは、実際の製品から確認し直します。

- サポートへの答え、
- 機能の説明文書、
- 変更の記録、

-

同じ振る舞いの二つの版を、手で保守することはもうありません。一方はコードの中に、もう一方は、公開した時点から（いえ、それよりずっと前から）古び始める資料の中に、というやり方です。

とはいえ、すべてがコードから来るわけではありません。

- 利用者調査はそこにありません。
- 法令上の義務は自分のソースを保ちます。
- 決定は、そこから生じた振る舞いに縮められません。

コードは、実装されたものを担います。コンテキストはなぜを保ち、状況が変わったときにそれを論じられるようにします。

六か月後に戻ってくる

サポートに渡した答えは、今日調べた製品の版に結びついたままです。そのあとの版を自動的に述べるものではありません。

- 六か月後、招待の道筋は変わっているかもしれません。
- 機能フラグが別の変種を有効にしているかもしれない、新しい口グインの方式が現れているかもしれません。

コンテキストは、問い、機能上の答え、検証の日付、調べた版、挙げた根拠、告げた限界を保ちます。そのおかげで、私たちが筋道を立てたときに何を知っていたのかを取り戻せます。

いまの振る舞いを知るには、スキルに検証をやり直してもらうだけです。

- 古い答えを写すのではなく、実際の製品から出発します。

そうすれば、しばしば混同される二つの問いを分けられます。

- 当時、なぜ私たちはこの決定を下したのか。
- 今日、製品は何をしているのか。

コンテキストは筋道を保つことで前者に答えます。スキルは真実の源に戻ることで後者に答えます。

ですから製品がコンテキストに取って代わることはなく、コンテキストが製品を含んでいると言い張ることもありません。

- 一方は、確かめるべき現実を与えます。
- もう一方は、それに意味を与え、変えていくために必要な知識を保ちます。

この行き来は、外の道具で終わりではありません。同じ製品の違う側面に複数のコンテキストが取り組んでいるとき、その知識もまた、ひとつのディレクトリに溶かし込まれずに出会えなければなりません。

覚えておくこと——コード中心の考え方

仕様書を製品の恒久的な説明として保守しない。現在の振る舞いはコードから確認し直す。答えは、添えられた根拠と、告げられた限界によってのみ値打ちを持つ。コードは実装された「何を」を担う。コンテキストは「なぜ」を保ち、それをふたたび論じられるようにする。

コンテキストが三つになった日

ある技術者が、作業報告書に抜けていた情報のために顧客へ電話をかけ直さねばならなかった、と話してくれます。

私はその一言を書き留めます。そして手が止まります。どこに置けばいいのか分からないのです。

- 報告書の品質についてのコンテキストがあります。本書の後半をずっと占めてきたものです。
- 作業そのものについての二つめがあります。何が含まれるのか、どんな規則が働くのか、それについてすでに何が決まったのか。
- そして数週間前に開いた、もっと広い三つめがあります。一次対応を現場の作業者に担ってもらうにはどうするか。

技術者の一言は、三つすべてに語りかけます。

- 欠陥のある報告書を記録しています。
- 作業について何かを語っています。
- 作業者をめぐる問いを照らします。

ここで問題の性質が変わります。

- コンテキストがひとつだったあいだ、問いはこうでした。これは入るのか、対象外なのか。
- 三つになると、問いはこうなります。どれに入れるのか。

間違え方は二通りある

最初の誘惑は、すべてをひとつにまとめることです。

- ひとつの大きな空間、すべての知識を同じ場所に、片づけの問いはもうなし。

それこそ私が逃げてきたものです。すべてを含むただひとつの空間は、文書管理システムがずっとそうだったものに帰ります。つまり、

- 情報はそこにあるのに、参照のたびに選り分けが始まる場所であり、
- 目的が混ざり合い、
- ソースが積み上がり、
- どのノートにもはっきりした持ち主がいなくなる場所です。

この行き止まりは本書の初めに語りました。新しい語彙でそれを作り直しても、良くはなりません。

二つめの誘惑はその逆です。それぞれのコンテキストが自分の家にとどまること。どれも清潔で、輪郭があり、筋が通っていて——そして盲目です。

結果はすぐに見えます。報告書についての私のコンテキストは、作業についてのコンテキストがすでに担っていた規則を安定させるのに三セッションを費やしました。目に見える重複ではありませんでした。誰にも気づかれないまま、やり直された仕事でした。ですから第三の道が要ります。それは、それぞれのコンテキストが何を担っているのかを知ることから始まります。

名前の前にひとつの数字

私のコンテキストは、01-saisie-compte-rendu、02-intervention、03-autonomie-operateurs、04-exploitation-catch-all という名前です。

数字は順位ではなく、名づけの約束ごとです。開く前から、そのコンテキストが何を担っているのかを告げます。

- **02- 製品のひとつの実体。** 本番で動いていて、自分の規則を持つソフトウェアの一片です。ERP なら 02-facturation、私のところなら 02-intervention。そこにあるものは定まっています。このコンテキストは閉じられません。
- **01- 実体についてのひとつの仮説。** 01-saisie-compte-rendu——入力が変わるべきだと思っていますが、どう変えるかはまだ分かりません。そこにあるものは何ひとつ確定していません。ですから 01-のコンテキストを作り、安定したもの（本番で動いているもの）と仮説をはっきり分けます。
- **03- ひとつの問題。** 03-autonomie-operateurs——一次対応を現場の作業者に担ってもらうこと。これは機能ではありません。複数の実体をまたぎ、いくつもの 01-を生みます。
- **04- 一時保管場所。** 耳にした一言、PDF、直感など、まだどのコンテキストにも属さないテーマの情報を置きます。これがなければ、断片ごとにコンテキストを作ることになります。私は当初そうしてしまい、数週間で中身の薄いコンテキストを五十以上抱えました。

01-はいつか必ず、動いている仕事から外れる

本番に出た日、新しい入力はもう仮説ではありません。作業の一部になります。

以後、製品を述べるものは 02-intervention に差し出されます。

残りは探索のコンテキストにとどまり、そのあと保管されます。報

告書が何をするのかを探す場所が、二つとも生きている、という状態はもうありません。

実を結ばなかった仮説は、それを止めたものとともに保管されます。消し去るべき失敗ではありません。二年後に同じ道をたどり直さずにすむものです。

04-のほうは、逆向きに空になります。ある問題が立ち上がった日、それに対応するコンテキストを作り、受け皿がそれについて何を保っていたかを見に戻ります。役に立たないものは削除します。

このすべてが、ただひとつの問い——これをどこに置くか——を、調べているものと確定しているものを分けることで片づけます。

- この分離がなければ、手放した仮説のたびに、その痕跡が製品の記述に残ります。
- そして、すべてが真であるはずの場所では、もう何も仮定できなくなります。

何か月もかけてようやく見えた誤り

私は 03-tarification を開くことから始めました。当たり前に見えたのです。大きなテーマで、複数の実体にかかわり、長く続く。

六か月後、そこにはこんなものが入っていました。

- 価格表、
- 競合の比較二件、
- 値引きについての議論、
- 誤った請求書についてのサポートの問い合わせ、

- ベルギーの付加価値税についてのノート、
- 競合の画面の写し三枚、
-

どれも対象外ではありませんでした。すべてたしかに価格づけの話でした。

まさにそれが問題です。ある材料を断るには、「これは私の役に立たない」と言えなければなりません。ところがこのコンテキストは何も探していませんでした。**問いを持っていなかったのです。**札だけがありました。

03-autonomie-operateurs のほうには問いがあります。一次対応を現場の作業者に担ってもらうにはどうするか。

ですから 03-は大きなテーマではありません。ひとつの問いです。価格づけはそれで消えたわけではありません。私のコンテキストをまたいで走っています——02-facturation がその規則を担い、ある 01-がそれに依存することもあります——が、どれも支配しません。

ですから私の約束ごとには、領域のための枠も、ほかの多くのもののための枠もありません。置き場所が明らかなものを片づけるためのものであって、それ以上ではありません。残りは受け皿で待ち、どこにしまったかを思い出すのではなく、何を探しているかを述べることで見つかります。

その受け皿は、間違えながら学んだ結果、二つあります。ひとつにまとめていたときは、プロダクトの仕事から来るものと、産業現場の運用という業務から来るものが混ざっていました。

二つの語彙、二つの読者、同じ対象を名づける二つのやり方。
空になるより速く、読めなくなりました。

- 04-exploitation-catch-all は、顧客の業務にかかわるもの、つまり運用を受け取ります。
- 04-productops-catch-all には、私の職務とその進め方、つまりプロダクトオペレーションに関する情報を置きます。

どちらも読めるままなので、どちらも空になっていきます。
技術者のあの一言は、受け皿を必要としませんでした。 02-intervention に入りました。いま私が抱えている問いとは無関係に、製品について真であることが生きている場所だからです。
あとは、三つのコンテキストがそれにどう気づくのかです。

覚えておくこと——四つの接頭辞、ひとつの問い

名前の中の数字が、コンテキストの担うものを告げる。02-は動いている実体、01-はその実体についての仮説、03-は複数をまたぐ問題、04-はまだ置き場所の分からないものを預ける関門。01-は、以後製品を述べるものを自分の02-に差し出し、そのあと保管されるために存在する。実を結ばなければ、それを止めたものとともに保管される。領域はコンテキストではない。複数をまたいで走り、どれも支配しない。問いを持たないコンテキストは、何が対象外かを決して知ることができない。

隣のコンテキストを参照する——情報は移さない

01-saisie-compte-rendu、作業報告書の品質に捧げたコンテキストに戻ります。先へ進むには、作業そのものの規則を知る必要があります。技術者が何を入力してよいのか、何が必須なのか、何がすでに決まっているのか。

これらの規則は、私のコンテキストの中には生きていません。隣のコンテキスト、製品に実際に存在するものを述べる 02-intervention のなかで安定しています。

手元に置くために写すこともできるでしょう。六か月後にはおそらく、同じ規則の二つの版を抱え、どちらが正しいのかを知る簡単な手立てがなくなります。

ですから私は、もっと軽い所作から始めます。私のコンテキストが、02-intervention を読むと宣言するのです。

読むことは、知識を持ち主のもとに残す

この関係は、私のコンテキストの安定した枠組みのなかに、使命と指示の隣に書かれます。それが開くのはひとつの明確な権利です。02-intervention がすでに知っていることを参照すること。

情報は持ち主のもとに残ります。そこで変わり続け、私の探索はそれに寄りかかりながら、所有しているとは言い張りません。こうして私は、報告書にかかわる調査のたびに必須項目の規則を複製せずに、いまの規則を確かめられます。そのソースや、それを築いた決定にも戻れます。

読むことは、変える許しを与えません。あるノートが不完全あるいは誤りに見えても、私のコンテキストは提案を言葉にできるだけで、隣人を直接直すことは決してありません。

この最初の関係は、よくある必要に応えます。外の知識を使いながら、その責任を負う場所でそれを生かし続けることです。すべての場合に応えるわけではありません。

取り込むことは、知識を自立させる

参照するだけでは足りないこともあります。

ある決定が寄って立つ規則を、そのままの形で固定しておきたいことがあります。自分の探索の限られた範囲に合わせて言い直したいこともあります。元のコンテキストが保管されたり、移されたり、組み替えられたりしても、その知識が手元に残ってほしいこともあります。

そういうとき、私はそれを取り込むことを選びます。

取り込むとは、知識を自分のコンテキストに入れることです。ノートをそのまま取ることも、自分の語彙で言い直すこともできます。必要に応じて、それを確かめられるソースも写すか、そこへの参照を保ちます。

とはいえ、その写しは過去のないアイデアになるわけではありません。どのコンテキストから来たのか、いつ取り込んだのか、何を言い直したのかを保ちます。

ですから違いは、技術的な所作よりも、そこから生じる責任にあります。

- 読むとき、知識は隣人のものであり続けます。

- 取り込むとき、自立した版が私のところに入り、その使い方の責任は私のものになります。

読むことは依存を生みます。取り込むことは新しい責任を生みます。

動きは逆向きにも進む

私の探索は、02-intervention がすでに知っていることを消費するだけではありません。その作業から、観察や仮説、ときには長く使える知識も生まれます。

報告書の品質を調べるうちに、ある入力 of 規則が実は作業そのものに属するのだと分かることがあります。この知識が、いつか保管されるかもしれない探索のコンテキストに閉じ込められたままではありません。

私はそれを 02-intervention に直接書き込みません。提案します。行き来 of 向きが、いま変わりました。

```
01-saisie-compte-rendu — 読む → 02-intervention
01-saisie-compte-rendu — 提案する → 02-intervention
                                   |
                                   └─ 取り込むかどうかを決める
```

知識を生んだ側が、その最終的な置き場所を決めるのではありません。受け取るコンテキストが、自分の使命に照らしてそれを吟味します。

次のことができます。

- ノートをそのまま取る。
- 自分の語彙で言い直す。

- 二つの読み方がともに成り立つなら、食い違いを保つ。
- 探索に役立つことが作業そのものを述べていなければ、取り込みを断る。

断ることは、行き来の失敗ではありません。局所的な結果が、黙って一般的な真理になってしまうのを防ぐものです。

探索が製品の現実になるとき

数週間後、報告書の新しい入力の本番に出ます。01-saisie-compte-rendu は役目を果たしました。仮説にすぎなかったものが、製品の現実になったのです。

とはいえ、その中身をすべて 02-intervention へ移すわけではありません。探索が生んだものを見渡し、二つのまとまりに分けます。

ひとつめには、以後製品を述べる、あるいは説明するものが入ります。

- 新しい入力の規則を打ち立てるノート。
- それを名づけるために安定した語彙。
- 構造を決める決定と、その選択の理由。
- 本番投入後も未解決の矛盾。
- この知識を確かめるために必要なソースや参照。

これらが 02-intervention に差し出されます。安定したコンテキストは、それらを闇雲に積み上げません。既存のノートを豊かにし、言い直すべきものを言い直し、本当に新しいアイデアのときにだけ新しいノートを作ります。

私たちの例では、終了を四つの情報に限る規則が、それを説明する決定とともに 02-intervention に入ります。いまやそれは製品の現実を述べています。

ふたつめのまとまりは、いまの製品を述べずに、この取り組みを語ります。

- 手放した仮説。
- 退けた試作と筋。
- 途中の下書き。
- 探索ならではの作業。
- たどった道のりを理解させてくれる議論。

これらは 01-saisie-compte-rendu に残ります。役に立たなくなっただけではありません。何が試されたのか、なぜある方向が断られたのか、新しい理由なしに繰り返すべきでないのは何かを説明します。

この選り分けが済むと、探索のコンテキストはまるごと保管されます。読めるままで、あとから再開できますが、動いている仕事を述べているとはもう言いません。

探索が実を結ばなかった場合も、断念の理由とともに同じように保管され、02-intervention には何も押しつけません。

ですから本番投入は、二つのコンテキストを融合させません。見直しを引き起こします。以後製品を述べるものは安定したコンテキストに入り、探索を語るものは保管されたコンテキストに残ります。

この選り分けは双方を守ります。

- 探索は、間違える自由を保ちます。

- 安定したコンテキストは、自分のものと認めると選んだものだけを受け取ります。

関係は枠組みの一部である

読むことも、取り込むことも、提案することも、暗黙の所作のままであってはなりません。

コンテキストの安定した枠組みが、次のことを示します。

- ほかのどのコンテキストを参照できるのか。
- その参照が何を許すのか。
- どんな規則のもとで知識を取り込めるのか。
- その出どころと、言い直した点がどう保たれるのか。

この取り決めに、重い仕組みは要りません。関係に宛て先を与えて、来歴と責任と境界が見えたままであるようにするだけです。

ですから網は、すべてをすべてにつなぐことで築かれるのではありません。関係をひとつずつ、仕事が本当に隣人を必要とするたびに築かれます。

- 別のコンテキストを参照すれば、相手が持つ知識を複製せずに済みます。
- 取り込む場合は、自分のコンテキストで管理すべき独立した情報にします。
- 提案することは、私の探索が生んだものを取り込むかどうかを、隣人が決められるようにします。

この行き来は、紙の上では整って見えます。ただし、コストもあります。参照先は通知なしに古くなり、複製した情報は出ど

ころを失い、二つのコンテキストが異なる解釈を持ち続けることがあります。

それを次の章で見えていきます。

覚えておくこと—三つの所作、三つの責任

参照では、知識は元のコンテキストに残る。取り込みでは、受け取る側のコンテキストに、出どころを記録した独立版を作る。探索は学んだことを提案するが、何が自分のところに入るかは、受け取るコンテキストの持ち主が決める。本番投入はディレクトリを融合させない。安定したコンテキストは以後製品を述べるものを取り込み、探索は残りを保ったまま保管される。

行き来が払わせるもの

六週間が過ぎました。

作業報告書についての私の仕事は校閲に出せる状態になり、そこで誰かがずれを見つけます。私が寄って立っている規則——終了時にどの項目が必須かを定めるもの——が、もう製品のものではないのです。

それは変わっていました。私が働いているあいだに別の探索が実を結び、作業のコンテキストがそれを取り込んでいたのです。

誰も教えてくれませんでした。私は読み直しませんでした。六週間前に読んでいたからで、戻るべきだと告げるものが何もなかったからです。

前の章は、きれいに見える規則を置いています。読むときは参照し、取り込むときは写す、と。私はそれを決着した判断だと思っています。

それは危険をなくしません。動かすだけです。どちら側に落ちるのかを知っておいたほうがいい。

見えない依存

いま起きたことには名前があります。盲目の依存です。

結びつきは生きているので、私が読むものは私なしに変わりえます。それは問題ではありません。たいていは、まさにそれが望みです。規則が持ち主のもので変わっていくのは、システムが働いているということです。

問題は、それが動いたと知らないことです。

足りないものは簡単に言えます。私が読んでいるものを、何も見張っていないのです。私のコンテキストは、自分が誰に依存しているかを知っています。依存先がいつ変わったのかは知りません。これは仕組みの一面にすぎず、もう一面ではコストが生じます。写していれば、これからは守られたでしょう。受け取るコンテキストは自立します。ソースの存在にも、その変化にも、もう依存しません。

けれども引き換えに、別のものを失います。写しがどこから来たのかを何も語らなければ、来歴は消えます。そして一年後、正しいけれども何に支えられているのか分からないノートが手元に残ります。それこそ、本書を通して語ってきた追跡可能性が防ごうとしている状態です。

ですから結論は、どちらかの陣営を選ぶことはありません。それぞれの所作が何を担うべきかを書き記すことです。

- **参照**は、そのソースがいつ変わったのかを知ることを求めます。
- **写し**は、どこから来たのかを告げることを求めます。どのコンテキストから、いつ、そして途中で何を言い直したのか。

ありうる発展

もうひとつの答えは、これから来る技術的なものになるでしょう。コンテキストが、自分の状況の変化を——とりわけ相手のコンテキストが見ている範囲の変化を——その観察者たちに知らせる力です。コンテキストは生きているテーマであり、探るべき可能性はまだ数多く残っています。

読むことが与えない権利

よく読むコンテキストは、やがて少し自分のもののようには思えてきます。隣人のノートに直接直したくなったことが、何度もあります。不完全で、何を書けばいいかわかっていて、三十秒で済んだはずでした。

それでも規則は動きません。私が相手に伝えることは提案であり、決定は受け取る側のものです。

追跡可能性を壊さずに削除する

削除には別の規則が要ります。それを私は、ひとつの受け皿を空にしながら学びました。

あるノートがコンテキストを移るとき、それを生んだソースはあとに残ります。もうどのノートも結びついていない、独りきりの状態になりえます。

とっさに片づけたくなります。その反応は半分だけ正しい。

すでに読まれ、ノートに変えられたソースは削除できます。担っていたものはよそで生きています。まだ誰も読んでいないソースは、そうではありません。

- 誰も取り出していない情報を担っており、それを捨てるのは、開いてすらいらないものを失うことです。

ですから基準は、ファイルの古さでも、占めている場所でもありません。

- 読まれたかどうかの状態であり、それといまなお引用しているものとの掛け合わせです。

ひとつの条件が伴い、これは譲れません。削除の前に、どのノートも、どの語彙の項目も、どの下書きも、履歴のどの行も、そのソースをもう引用していないことを確かめること。この確認なしでは、この規則は、自分が仕えるはずの追跡可能性を壊します。

食い違いは誤りではない

二つのコンテキストが同じテーマについて違うことを言う場合が残っています。

とっさにそこへ不整合を見て、秩序を戻すために早く断を下したくなります。それはしばしば誤りです。

食い違いは、見ている範囲の違いかもしれません。それぞれが自分の家では正しい。一方では検討中でも、もう一方は存在すら知らない、未検証の仮説かもしれません。テーマを二つの場所から見ているために、どちらも成り立つ二つの読み方かもしれません。

そのときは、そのまま保たれるに値します。それを決めるのは人間です。システムが保証すべきなのは、その食い違いが存在すると分かることです。

忘れられた食い違いは、いつも最悪のときに自分を思い出させます。誰かが、もう一方の存在を知らないまま、片方の版に寄りかかるときです。終了の規則で私に起きたのがそれでした。ただし二つの読み方ではなく、古びたひとつの読み方だったという違いはあります。

このようにネットワークにはコストがあります。後になって問題として発見するのではなく、あらかじめ明示しておくべきです。二つのコンテキストを結んでも、見張る仕事はなくなりません。それを可能にし、そして人間の側へ動かすのです。

つまるところ、それこそ本書が初めから語ってきたことです。

要点——ネットワークのコスト

参照は知らせもなく古び、写しは来歴を失いうる。どの所作にも未済があり、それがどれかを書き記さねばならない。読むことは書く許しを与えない。隣人に伝えることは提案であって、訂正ではない。すでにノートに変えられたソースは消えてよいが、一度も読まれていないソースは決して消してはならない。そして、誰がまだ引用しているかを確かめずに何かを削除してはならない。矛盾する二つのコンテキストは、必ずしも間違っていない。重いのは、それを知らずにいることである。

読み終える前に

結び

結び

本はここで終わりますが、テーマはそれとともに終わるわけではありません。

あとがきでは、この取り組みの現在地、十分に扱えなかった点、今後どこで続きを発表するかを述べます。最後の章では、コンテキストに先行する考え方や隣接領域との関係を整理し、何を取り入れ、何を取り入れなかったのかを示します。

あとがき

ここまで来てくださって、ありがとうございます。

この本をできるだけ読みやすくしようとしたましたが、完全には果たせませんでした。望んだより味気ない箇所も、物語より仕組みが前に出てしまった場所も残っています。それでも最後まで読んでくださいました。そのことを、決して当たり前とは思いません。

本書で扱えたのはテーマの一部だけ

厚みのわりに、コンテキストを表面からしか扱っていません。

概念の面で手薄なままの部分がいくつもあり、出版する時点でそれを分かっています。たとえばアトミックノート。実装のしかたによって、得られる結果はひどくばらつきます。あるコンテキストでうまくいくことも、LLM Wiki へ移ればすぐに手直しが必要です。情報の量も使い道も同じではないからです。

こうした点は、さらに掘り下げる価値があります。ここで扱いを止めたのは、本はどこかで終えなければならないからであって、テーマを語り尽くしたからではありません。

これからも、かたわらの仕事であり続ける

コンテキストは進み続けますが、つねに本業のかたわらの取り組みとしてです。

職業の傍らで行っている活動であり、すでに多くの夜を注ぎすぎました。また戻ってくるでしょう——いつも戻ってくるのです——が、そろそろ力を抜いて、人生を味わうときです。

次に探りたいこと

先行する仕事の調査は行いました。私のものにきわめて近い概念が浮かび上がり、ときにプロダクトや知識管理から遠い領域に、しばしば別の言語で——とりわけ日本語と中国語で見つかりました。その成果は、フランス語圏や英語圏からはほとんど見えないままです。この進め方についての記事を書きました。ブログに載ります。

足りないのはその続きです。これらの仕事を分析すること。

名前は特定でき、論文はそこにあり、私はそれを読んでいません。私が夜にひとりで取り組んできた問いについて、何百人もの人が何十年にもわたって重ねてきた研究です。それを読み解くほうが、この枠組みを発明し続けるよりはるかに確実に、それを強くするでしょう。

続きは第二巻にはならない

この本は、概念を結晶させる役に立ちました。それが役目でした。散らばっていたものを集め、受け渡せるようにし、論じられるだけ安定した形を与えること。

しばらくのあいだ、その発展は malorean.net で公開され、第二巻やこの本の新しい版にはなりません。記事は直せ、補え、覆せます。本は内容を固定します。しかし、ここで述べたことは、新しい版として再び固定するほどにはまだ成熟していません。

続きを追いたい方は、そこで書かれていきます。

系譜と隣接領域

コンテキストは、知の空白のなかに現れたものではありません。

それを構成する仕組みの多くには、ときに非常に古い先例があります。知識管理、ノート術、ソフトウェア工学、意思決定支援システム、認知科学、組織記憶などの領域です。

こうした影響の一部は、初めから意識していました。ほかは、手法を形にし、本をほぼ書き終えたあとで先行研究を調べるなかで見つけました。

ですから二つのものを区別しなければなりません。

影響とは、私が知っていて、その原理のいくつかを意識して取り入れ、あるいは作り変えた手法や考えのことです。

近縁または**隣接**とは、近い問題を解いた先行のシステムのことで、驚くほど似た仕組みを備えていることもあります。コンテキストに直接影響を与えたわけではありません。

この区別は重要です。二つの考えは、一方が他方から派生していなくても近いことがあります。

ツェッテルカステン—使い回せる単位の網としての知識

ツェッテルカステンとの近さは、おそらくもっとも明らかなです。ニクラス・ルーマンのシステムでは、ノートをその時々プロジェクト用フォルダーにしまうだけではありません。ノートは相互に結びついた永続的な仕組みに加わり、新しい議論にも再利用できます。

コンテキストはこの考えを強く共有しています。ある仕事のあいだに生まれた知識は、その成果物とともに消えるべきではない、という考えです。

コンテキストのアトミックノートは、はっきりとこの系譜に属します。

主な違いはその位置づけにあります。アトミックノートは来歴を保たねばならず、ソース、矛盾、決定、現在の状況、統合された知見、成果物をはっきり区別する構造のなかに置かれます。ですからツェッテルカステンは、**知識の資産を少しずつ築くこと**については非常に強い近縁ですが、システム全体についてはそうではありません。

PARA——行動に合わせて情報を整理する

ティアゴ・フォルテの PARA は、コンテキストが意識して受けた影響のひとつです。

PARA は情報を Projects、Areas、Resources、Archives の四つに分類します。その重要な発想のひとつは、情報整理は世界を抽象的に分類するためではなく、行動に役立つものであるべきだということです。

コンテキストはこの直観を受け継いでいます。

コンテキストは、テーマが存在するから存在するのではありません。人間がひとつの使命を追っているから存在します。

ですから境界は「データベースについて私が知っていることすべて」ではありません。

「製品 X のためにデータの構造を選ぶ」でありえます。

違いは、コンテキストそのものが自立した作業の単位になることです。自分の状況、ソース、知識、決定、そして生涯を持ちます。

ドメイン駆動設計——中身より先に境界

コンテキストという語そのものが、ドメイン駆動設計の**境界づけられたコンテキスト**を自然に思わせます。

近さはとりわけひとつの考えにあります。ある情報や概念は、ある境界の内側でしか意味を持たない、というものです。

同じ語が、二つの異なるコンテキストで違う意味を持ちえます。それらを融合させる必要はありません。

コンテキストはこの直観を移しています。

ただし境界を、筋の通った業務モデルに依存させる代わりに、**人間の使命**に依存させます。

ですから二つのコンテキストは、追う目的が違うという理由で、同じテーマについて違う理解を保てます。

Git——分岐、自立、そして再統合

Git は、より直接に技術的な影響です。

探索が十分に大きくなったときは、安定した空間をかき乱すより、別のコンテキストのなかで育てたほうがよいことがあります。

この考え方はブランチに似ています。

探索は分岐し、自分のソースと知識を生み、そのあといくつかの結論が元の空間に再統合されえます。

けれども Git と同じように、統合はすべてを融合させることを必ずしも意味しません。

この直観は、コンテキストどうしの関係にも影響しました。局所的な自立、ありうる分岐、そして選んでの統合です。

プロジェクトメモリー—プロジェクトのなぜを保つ

プロジェクトメモリの研究は、コンテキストを構築した時点では知らなかった近縁分野です。

そこで立てられている問題は、仕事が何か月も続くときに現れるものとほとんど同じです。組織は最終的な資料をわりあい容易に保ちますが、プロジェクトがどのように、なぜその結果に至ったのかを説明する背景は、はるかにうまく保てません。

なぜこの決定なのか。

その前に何を試したのか。

その時点でどんな制約があったのか。

新しい人が続ける前に、何を理解しておくべきか。

コンテキストは、いくつもの別々の記憶で同じ問いに答えます。現在の状況、履歴、決定、統合された知見、そして引き継ぎです。ですから働きの上での近さは、とても強いものです。

エクスペリエンス・ファクトリー—成果物は資産ではない

ヴィクター・バジリの研究のまわりで育ち、とりわけ NASA で試されたエクスペリエンス・ファクトリーは、さらに重要な近縁をもたらします。

それは、プロジェクトを実行する組織と、そのプロジェクトの経験を次のプロジェクトのために使い回せる知識へ変える組織とを区別します。

この考えは、コンテキストの根本的な区別にまっすぐつながります。

成果物は資産ではない。

プロジェクトは、優れた結果を出しながら、あとにほとんど何も残さないことがあります。

逆に、その経験を分析し、変え、保つことで、次の問題を解く力を高められます。

コンテキストは、この蓄積を、仕事が行われるその空間に直接組み込みます。

ナレッジ・センタード・サービス—働きながら知識を生む

ナレッジ・センタード・サービス—KCS—は、おそらくコンテキストのもっとも興味深い隣人のひとつです。

その原理は、文書化を、仕事のあとに行う別の活動にしてはならない、というものです。

問題が解かれたとき、既存の知識が探され、使い回されます。不完全なら、改められます。存在しなければ、作られます。

こうして、働くという事実そのものが、知識を生み、手入れる仕組みになります。

それはまさにコンテキストの望みのひとつです。

システムを使うたびに、その情報資産が増えてほしい。

ただし KCS は主にサービスとサポートの環境のために考えられています。コンテキストのほうは、数多くの使命に通じる構造を与えようとしています。

HWMSE——人、機械、知識をひとつの空間に

調査のなかで見つけたもっとも意外な近縁は、おそらく中国から来ます。

Hall for Workshop of Meta-Synthetic Engineering——

HWMSE——は、銭学森の研究のまわりで、一九九〇年代の初めから育てられてきました。

複雑な問題を扱うために、次のものをはっきり組み合わせるようになられています。

- 人間の専門家。
- 機械のシステム。
- 知識のシステム。

さらに興味深いことに、HWMSE についての研究は、既存の知識を使うことだけを語るものではありません。**知識の創造**と、調べている問題に固有の知識システムが、使われているあいだに絶えず豊かになっていくことも語ります。

ですからコンテキストとの近さは大きいものです。

- 問題へ向けられた活動。
- 人と機械の協働。
- 作業中に利用される資源とモデル。
- 活動によって生まれる知識。
- 積み上がる資産。

とはいえ、HWMSE はコンテキストと同じ運用の文法を備えてはいません。ソース、アトミックノート、状況、統合された知見、矛盾、引き継ぎ、成果物という形での分離は、そこには見当たりません。

それでも調べたシステムのなかで、構造としてはおそらくもっとも近い隣人のひとつです。

ScholOnto と SWAN——命題はそのソースではない

もうひとつの強い近縁が、学術的な議論に捧げられたシステムに現れます。

ScholOnto と、のちの SWAN は、学術文献と、**それらの文献から生み出された命題**を区別します。

ある資料が、ある主張の出どころになりえます。

その主張は来歴を保ちます。

ほかの要素によって支えられ、論じられ、覆されえます。

これはコンテキストのアトミックノートを支える原理にきわめて近いものです。

ソースはソースのままである。そこから築かれた知識は別の対象である。

これらのシステムはまた、知識システムに記録された命題を、自動的に真と見なしはならないことも示します。

命題には来歴があり、ひとつの筋道のなかでの位置づけがあります。

これはコンテキストの知の構造と、とりわけ強い近縁です。

連合型システム——読むことは写すことではない

最後に、コンテキストどうしの関係についてのいくつかの考えには、連合型データベースのシステムのなかに、驚くほど古い親がいます。

これらの構造は、自立した複数のデータベースが、自立を手放さずに協力できるようにしなければなりません。

あるシステムが、一部の情報を公開できます。

別のシステムが、それを遠隔で参照することも、手元に写しを持つと決めることもできます。

そして写しが作られた瞬間から、古典的な問題が現れます。原本が変わったら、どうなるのか。

コンテキストでそれをまとめているのが、次の言い方です。

参照は依存、写しは責任。

ですから情報技術上の仕組みは、ずっと前から存在していました。

コンテキストはそれを、人と AI の空間のあいだの知識統治のルールへ移し替えています。

無からの発明ではなく、ひとつの統合

これらの隣接は、コンテキストをまったく新しい仕組みの発明として示すことを難しくします。

- 歴史が示しているのはそれではありません。
- 来歴は存在していました。
- アトミックノートは存在していました。
- プロジェクトの記憶は存在していました。

- プロジェクトをまたぐ蓄積は存在していました。
- 仕事のあいだの知識創造は存在していました。
- 人と機械と知識システムを組み合わせる環境は存在していました。
- 出どころつきで、異議を唱えられる主張は存在していました。
- 情報を共有あるいは複製できる自立した構造は存在していました。

コンテキストをより特徴づけていると思えるのは、これらの仕組みが**ひとつの手法のなかで選ばれ、組み合わせられ、揃えられている**やり方です。

手法は、それらに共通の文法を与えます。

- ソース。
- アトミックノート。
- 矛盾。
- 決定。
- 状況。
- 履歴。
- 統合された知見。
- 行動。
- 引き継ぎ。
- 成果物。
- コンテキストどうしの関係。

そして何よりも、これらの仕組みをきわめて現代的な問題に適用します。人間と AI の協働を、個々の会話が孤立して以前の記憶を失う状態にせず、長期にわたって継続するという問題です。このように、コンテキストには多くの先例があります。先例を知ることで、コンテキスト独自の部分も、より明確に理解できます。