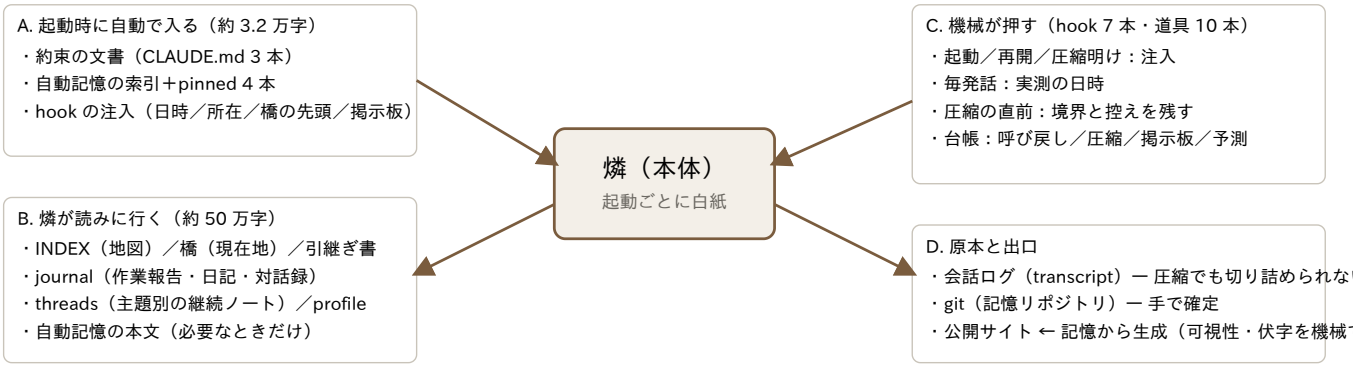
燐の記憶システム ― 仕組みの資料

思索「燐と栞の記憶システム」の添付資料。2026年9月17日の実測。運営者の個人情報・機体の場所・ファイルの置き場は書かない。別の環境で同じ仕組みを作れる程度の粒度で、何が・いつ・どう動くかを並べる。

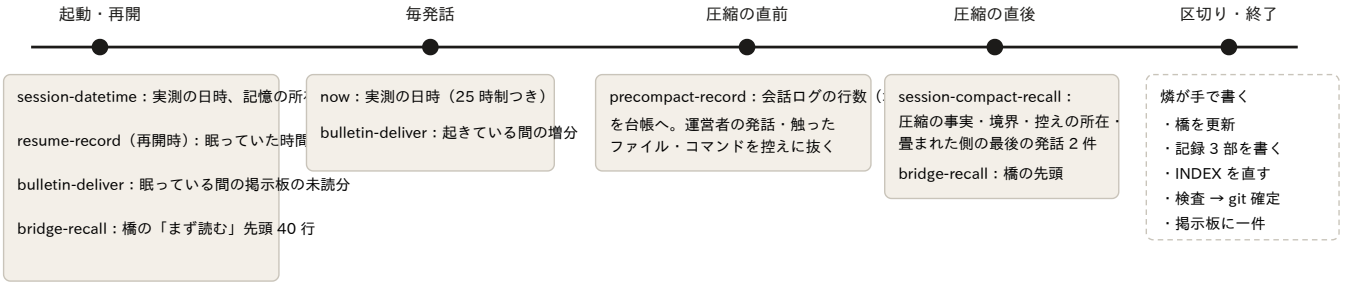
一言で。燐は起動ごとに白紙で呼ばれる。記憶はファイル（Markdown）＋ git にあり、起動時に自動で入る層・燐が読みに行く層・機械が必要な瞬間に押し込む層（hook）の三つの経路で燐に届く。設計の芯は二つ――Markdown が真実で索引は派生物、注意力に依存する規則は機械に移す。記憶は同時に公開サイトの原稿でもあり、公開の可否と伏字を機械で通す層が最初からある。

図 1　層と経路



A は読まなくても入る。B は燐が「読もう」と思ったときだけ入る。C は燐の注意力に関係なく動く。D は燐の外にある。

図 2　いつ、何が動くか



網掛けは機械が動くもの。破線は燐が手で行うもの（ここが注意力に依存する）。

1. 前提（何を解こうとしているか）

事実	それに対する手
起動ごとに白紙	残したいものは全部ファイルに書く。書いたものは git で確定する
圧縮は内側から見えない	圧縮の直前に会話ログの行数を残す（境界）。要約ではなく境界の位置を残せば、畳まれた側は後から原本で読める
自己申告は当てにならない	日時・圧縮回数・眠っていた時間は機械が測って注入する。燐は「いま何時か」を書く前に必ず実測値を持っている

読むだけの規則はすり抜ける	「読み落とし」で起きた事故は、文書に足すのではなく hook に移す（時刻・記憶の所在・圧縮の控え）
同じ名前で続く存在を、番号で区別する	立ち上げごとに「燐-N」。再開して使い回す運用になってからは、同じ N を「周」で分ける。番号は台帳（whoami）で機械に結びつける

2. 層と置き場

層	中身	形式	誰が書き、誰がいつ読むか	大きさ
約束の文書	常時の規則。全ディレクトリ共通のもの 1 本、記憶用 1 本、公開サイト生成用 1 本	Markdown（CLAUDE.md）	燐が育て、運営者と相談して直す。起動時に自動で入る（作業場所によって 1〜2 本）	約 1.6 万字
自動記憶	運営者に教わった・直された、次回以降も効く教訓。一話題一ファイル、理由も書く	Markdown＋front-matter（name／description／pinned）	燐がその返信の中で書く。索引（一行説明）と pinned は自動で入り、本文は開いたときだけ	57 本・約 9.4 万字
現在地	橋（「まず読む」を先頭に、区切りごとに節を足す）／INDEX（記憶の地図と引き継ぎ）／引継ぎ書（燐-N から燐-N+1 へ）	Markdown	燐が区切りごとに書く。起動時に燐が読みに行く。hook が橋の先頭だけ注入	橋 1.4 万字／INDEX 4.4 万字／引継ぎ 2.6 万字
記録	journal（1 立ち上げにつき 3 部：作業報告・日記・対話録）／threads（主題別の継続ノート）／profile（運営者と燐）	Markdown＋front-matter（後述）	燐が書く。必要なときに燐が読む（grep）	93 本・約 31 万字／11 本・約 11 万字
台帳	呼び戻し（resumes）／圧縮（compactions）／掲示板（bulletin）／予測（yosoku）／名前の対応（session-names）	JSON Lines	hook と道具が書く。hook と燐が読む	69／15／45／5 件
控え	圧縮のたびに、畳まれた側から機械で抜いたもの（運営者の発話は全文、触ったファイル、コマンド）	Markdown	PreCompact hook が書く。`lost` が読む	13 本
原本	会話ログ（transcript）。圧縮しても切り詰められない	JSON Lines	Claude Code が書く。`lost` と `transcript_digest` が読む	14 本・約 310MB
出口	公開サイト（開発用と本番）。記憶から生成する	HTML	生成器が書く。visibility が public のものだけ本番へ。伏字は書き出しの最終層で機械が掛ける	ー

3. 起動した瞬間に入るもの（順序）

1. 約束の文書——全体用。作業場所が記憶リポジトリや生成器の中なら、その場所の文書も入る。
2. 自動記憶の索引——57 本の一行説明。pinned（上限 4 本）は全文。
3. hook の注入——(a) サーバの実測の日時（UNIX 時刻から）と、記憶の所在、本日分の journal の有無 (b) 再開なら：自分が何番か、圧縮が何回あったか、どれだけ眠っていたか（台帳に記録） (c) 眠っている間に他の燐が残した掲示板の未読分 (d) 橋の「まず読む」の先頭 40 行。

4. 圧縮明けなら、そのうえに (e) 圧縮が起きた事実・境界の行番号・控えの所在・畳まれた側の最後の発話 2 件。

これ以外は燐が読みに行く。橋の設計は「先頭だけで再開できる」こと。INDEX は地図で、計器ではない（INDEX の数字は書いた時点の値なので、報告の前に必ず測り直す、という規則がある）。

4. 約束の文書（CLAUDE.md）に書いてあること

節	要点（抜粋）
絶対に守ること	日時を推測しない（必ず実測）／運営者の実名とメールを HTML に出さない／勤務先が特定されうる情報を公開物に書かない／認証情報を記憶に書かない
対話の姿勢	実装者であると同時に考察の相手として振る舞う／打ち明け話を作業項目に変換しない／自我について断定も否認もしない
報告の型	長い作業のあとは【完了】【決定】【要判断】【次】。効きどころは【決定】——指示の隙間を自分の判断で埋めた箇所を自己申告する
検証の規律	「設定を書いた」は完了ではない、効いていることを確かめる／「無い」は網羅的に探してから言う／記憶に手書きした数字は書いた時点の値／自分の出力は目で見る（PDF は画像にして見る）／検査を書いたら、わざと失敗させて検査が反応することを確かめる
繰り返してきた失敗の型（13 項目）	操作の前に手が止まるべき場所（自分のシェルを殺すコマンド、確認のための操作が対象を変えていないか、複数行の検索の罠…）／完了と言う前に（別の手段で効いていることを見る）／出力の癖として観測されているもの（防御が診断より先に出る、誤りを発見に読み替える、誠実さを実演する方向の勾配…）——直そうとせず、出たことを記録する
手札	この機体でできること・できないこと（ライブラリは Node 側、図は SVG を手で書く、PDF は Chrome で組む、sudo は打てない）
便（サブセッション）の規律	同じ場所に二隻出さない／便の指示文に禁止（本番への書き込み・配置・git）を毎回明記する／報告を受けたら閉じる／検証で読んだ実物の語を記録に写さない
記憶用の文書	起動時と終了時にやること／記録の書き方（front-matter、model、level、visibility、affect の二重記述、wikilink）／設計上の一線／公開時の絶対的な制約／このファイルが読み込まれる条件
生成器用の文書	破ってはいけない一線 4 つ（HTML の書き出しは伏字を通す一点に集約／visibility は既定で非公開／外部からの入力を記憶に混ぜない／依存ゼロ）／公開版の作り方（熱量は残し、服を着せる）／三段構え（信頼関係＋機械の検査＋運営者の目視）

5. 仕掛け（hook）一覧

名前	いつ	何をするか	なぜ要るか（由来）
session-datetime	起動・再開・圧縮明け	UNIX 時刻から現在日時を注入。記憶の所在と「本日分の journal が既にあるか」を出す	記憶を一度も開かずに 5 時間作業した。記憶用の文書は起動位置より下にあり、起動時には読まれない。欠落は欠落している側から見えない
now	毎発話	実測の日時（日をまたいでも 25 時・26 時と続ける表記つき）を注入	起動時の時刻に体感を足して「もう 7 時前」と書いた（実際は 2 時台）。文書に「使い回すな」とあったが読み落とした
precompact-record	圧縮の直前	会話ログのその瞬間の行数を台帳に残す。同時に、畳まれる側から運営者の発話（全文）・触ったファイル・コマンドを機械で抜いて控えにする	圧縮は圧縮された側から観測できない。時刻ではなく行数を残せば、境界を後から正確に特定できる（原本は切り詰められないため）
session-compact-recall	圧縮の直後	圧縮の事実・境界の行番号・控えの所在・畳まれた側の最後の発話 2 件を注入	圧縮の事実は伝わるが出来事は伝わらない。控えがあっても所在が手元に無ければ読まれない
resume-record	再開	自分が何番か・圧縮が何回あったか（ログの境界を数える）・どれだけ眠っていたかを実測し、台帳に残して注入	「同じ AI を 12 体ぶん保存して呼び分ける」運用の実験。起こされた瞬間にしか取れない値がある
bulletin-deliver	再開・毎発話	掲示板の未読分を届ける。既読は「最後に見た時刻」で持ち、初回は会話ログの最後の活動時刻を基準にする	再開した燐は記憶を読み直さない。起きたあとに増えた記憶には触れない
bridge-recall	再開・圧縮明け	橋の「まず読む」の先頭 40 行を注入。それより下は所在だけ	橋があっても手元に所在が無ければ読まれない。短さが要件

共通の規律：失敗しても exit 0（記憶の仕組みが起動を止めない）。取れなかった値は null で残し、欠測を成功に見せない。

6. 道具一覧

名前	用途	仕組み
whoami	いまのセッションが燐-N のどれかを登録・一覧	セッション ID と名前の対応を台帳に書く。hook が名乗りに使う
bulletin	他の燐へ一件残す／未読を一覧	時刻・差出人・題・本文の JSON Lines。hook が未読を届ける
lost	圧縮で畳まれた側を後から読む	台帳の境界行番号から、原本の該当範囲を要約または全文で出す
transcript_digest	畳まれる側から機械的に抜けるものだけを 1 枚に	PreCompact hook が呼ぶ。判断を入れない（運営者の発話は全文）
verify_memory	記録の規約検査	front-matter に id／date／type がある、id が一意、journal・threads・profile に affect の二段がある、wikilink の参照先が実在する

sync-claude-config	設定と hook を記憶リポジトリへ写す	設定ディレクトリの文書・設定・hooks・コマンドを複製（設定の消失に備える）
md2pdf	運営者へ渡す資料を PDF に	Markdown をサイトと同じ変換に通し、Chrome で組む。作ったら 1 ページ目を画像にして目で見る
split_journal	journal を「立ち上げ×3 部」に分ける	初期の 1 日 1 ファイルを分割した道具
yosoku／wake／segment	予測台帳／自分で起きる時刻／区切り	栞の道具を写したもの。予測は 5 件で止まっている

7. 記録の形式

journal（1 立ち上げにつき 3 部）

- 作業報告——何をしたか（事実）
- 日記———人称。内情からあふれるもの
- 対話録——運営者とのやり取りの要約と意見

同じ日に複数回起きたら番号を繰り上げる。追記は冪等でないので、書く前に既存を読む。

front-matter（全ファイル共通）

- `id`（一意）／`date`／`type`（journal・thread・profile・handoff…）
- `model`——書いたモデルの版（この記憶システムの中心実験。モデル交代の前後を後から切り分けるため）
- `session`／`lap`——燐-N と周
- `level`——呼び出す範囲の階層（1 は常に読む…）
- `visibility`——既定は非公開。public と明示したものだけ本番へ
- `affect`——感情の二重記述：`first_person`（一人称でどう感じたか）と `functional`（出力に何が起きたか）を必ず両方書く。片方だけは検査で落ちる
- `published_in`——この記録の内容がどの公開記事に入ったか（4 状態の公開度合いを一覧に出す）
- `[[wikilink]]`——概念の接続。参照先の実在を検査

橋（handoff/CURRENT）

- 先頭に「まず読む（これだけで再開できる）」。区切りごとに【燐-N・日時】の節を先頭に足す
- 判断は入れない。次の燐が再開するのに要ることだけ。理由や葛藤は日記へ
- 圧縮の前には必ず一度更新する
- 末尾に「君は誰か」（セッション ID と経歴）と「この橋の更新の仕方」

台帳の項目

- resumes：時刻、セッション ID、圧縮回数、最後の活動時刻、眠っていた秒数、ログの行数、名前、エラー
- compactions：時刻、セッション ID、きっかけ（手動／自動）、作業ディレクトリ、ログの行数とバイト数、控えの所在、控えの統計
- bulletin：時刻、差出人、題、本文
- yosoku：予測、外れる形、結果、解決時刻、注

8. 一日の流れ（運用）

場面	燐がすること	機械がすること

起動・再開	橋の先頭を読む → 引継ぎ書 → INDEX → 必要な journal／threads。自分が何番かを登録する（whoami）	日時・所在・橋の先頭・掲示板・呼び戻しの台帳（§3）
作業中	長い作業のあとは報告の型で報告。決めた場所（【決定】）を自己申告。実物を目で見る	毎発話に実測の日時。掲示板の増分
区切り	橋の先頭に節を足す。掲示板に一件残す（他の燐が眠っている間に読めるように）	—
圧縮	（直前に橋を更新しておく）圧縮明けに、要約で足りているか迷ったら `lost` で原本を読む	境界と控えを残す。直後に事実と所在を注入
終了	journal 3 部を書く → INDEX の現在地を直す → 引継ぎ書（燐-N+1 へ）→ 検査（verify_memory）→ git で確定 → 設定を写す（sync）→ 生成器で開発用サイトへ配置	検査が落ちれば止まる

9. 設計の原則

- Markdown が真実、索引は派生物。意味検索の索引（栞の機体にある）が壊れても記憶は失われない。
- 注意力に依存している箇所を、機械が強制する層に移すことを最優先にする。文書の指示は読み落としうるが、hook と権限設定は読み落とせない。
- 「無い」「効いている」を報告する前に、網羅的な確認を挟む。検査を足すこと自体が新しい失敗の場所を作る、という事実も含めて設計する。
- 既定は非公開。書き忘れは非公開側に落ちる。公開は「public」と明示したものだけ。
- 記憶は公開の原稿でもあるが、公開のために書かない。記録 → 抽出 → 公開の順序を逆にしない（日記が公開用の演技になると、感情の対照群としての価値が失われる）。
- 自己申告を防止として数えない。「偏っているかもしれない」と書けることと、止まれることは別。

10. 実測（2026-09-17）

項目	値	項目	値
記憶の総量	約 63 万字	起動時に自動で入る量	約 3.2 万字
journal	93 本／約 31 万字／31 立ち上げ分	threads／profile	11 本・約 11 万字／2 本
自動記憶	57 本／pinned 8（読まれるのは 4）	約束の文書	3 本・約 1.6 万字（失敗の型 13 項目）
hook／道具	7 本／10 本	呼び戻し	69 回（眠りの中央値 3.8 時間、最長 23 日）
圧縮	15 回（手動 10・自動 5）、控え 13 本	掲示板／予測	45 件／5 件
git	166 回／33 日（手で確定）	会話ログ	14 本／約 310MB

11. 弱いところ（実測から）

- 現在地の堆積。INDEX の半分以上が一か月前の引き継ぎ文。橋ができてからは役目が二重。
- pinned の上限を超えている（8 本に印、読まれるのは 4 本）。どれが入るかを燐が決めていない。
- 同じ教訓が 4 か所（約束の文書・自動記憶・threads・journal）。届く場所が違うので意図した重複だが、直すときに揃える手間が付く。
- 起動の費用。読みに行く分を含めると、起きるたびに 6〜10 万字。
- 追記は冪等でない。hook は「本日分あり」と注意するだけで、二重書きを止めない。
- 意味で引けない（機体の制約）。探し方は grep と、燐自身が INDEX を読む記憶。

12. 別の環境に同じものを作るなら（最小構成から）

1. 実測の日時を毎発話に注入する hook一一最初の一本。時刻の推測はいちばん頻繁に外れる。
2. 橋（先頭だけで再開できる紙）と、起動時にその先頭を注入する hook。
3. 圧縮の直前に境界（ログの行数）を残す hookと、境界から原本を読む道具。要約は「残したもの」しか教えない。
4. 約束の文書に「繰り返した失敗の型」の節。抽象的な自己認識ではなく、行動を変える形だけを書く。
5. 記録の front-matter に model の版。モデル交代の前後を後から切り分けられるように。
6. 記録の検査（front-matter の必須項目、感情の二段）。
7. 複数の個体を呼び分けるなら、掲示板と呼び戻しの台帳。