# What the leaked Claude Code codeba tells healthcare builders about designing agentic health tech

APR 02, 2026



## Table of Contents



## Abstract

On March 31, 2026, a 59.8 MB JavaScript source map file was accidentally bundl into a public npm package release of Claude Code, Anthropic's flagship agentic coding CLI, exposing its entire ~512,000-line TypeScript codebase. A researcher Solayer Labs spotted it within hours. By nightfall the repo had thousands of fork Anthropic confirmed no customer data was involved and attributed the incident packaging error. That said, what got exposed is a rare unobstructed view into ho most commercially successful AI agent in production actually works under the h

Key findings relevant to healthcare builders and investors:

- Three-layer skeptical memory architecture directly applicable to clinical AI co management

- Coordinator mode (multi-agent orchestration) is a production-validated patter parallel clinical workflows like prior auth, coding, and documentation

- AutoDream background consolidation offers a model for persistent, contradict resolving clinical knowledge stores

- KAIROS proactive daemon mode is the architectural predecessor to ambient cl AI that surfaces insights before the clinician asks

- Permission and risk classification system is a template for HIPAA-compliant ag tool governance

- Compile-time feature gating via dead code elimination is a viable pattern for st rollouts in regulated environments

- The unreleased feature roadmap (1M context windows, task budgets, effort cor signals where health tech AI investment should be concentrating over the next 1 months

# How This Leak Happened and Why It Matters Beyond the Drama

The mechanics of the leak are almost embarrassingly simple. When you build a JavaScript or TypeScript project, the toolchain typically generates source map fil with the .map extension. These files exist purely to help developers debug produ code – they map compressed, minified output back to the original readable sour The structure embeds the actual raw source code as strings inside a JSON file. Normally you strip these before shipping to production. In this case, nobody did .map file went out with the npm package, and anyone who pulled the package co grab 512,000 lines of unobfuscated TypeScript from Anthropic's own R2 cloud s bucket. The ironic twist that every developer on X immediately noted: the leake source contains an entire subsystem called Undercover Mode, built specifically t prevent internal Anthropic information from accidentally leaking into public re They built a whole concealment architecture for the AI, then shipped the source of the entire thing in a JSON file. Probably via Claude.

For healthcare builders, the drama of the leak itself is mostly noise. What matte that this is one of the only detailed, verified looks at how a production-grade AI actually works at scale in a commercially mature product. Anthropic is reported running around $19B in annualized revenue as of early 2026, with enterprise cor accounting for roughly 80% of that. This is not a research prototype. The patter this codebase are the patterns of a system that works at scale, survives enterpris procurement, and retains users. That makes it a genuinely useful reference architecture for anyone building agentic software in healthcare, which is a secto desperately needs proven patterns for how to manage agent memory, orchestrate parallel tasks, handle permissions, and roll out features safely in regulated environments.

There are approximately 40 tools exposed in the leaked source, covering everyth from bash execution to file operations to web fetching to sub-agent spawning. T query engine alone is ~46,000 lines. The base tool definition is ~29,000 lines. Th not wrapper code. And the healthcare sector, which is simultaneously drowning workflow complexity and terrified of AI liability, has a lot to learn from how this architecture was assembled.

# The Memory Architecture Problem: Context Entropy and What It Means for Clinical AI

The single most important technical insight in the leaked source for healthcare builders is how context entropy was solved. Context entropy is the tendency of A agents to become progressively confused, hallucinatory, or inconsistent as sessio grow long and complex. It is the core unsolved problem for any AI agent that ne operate across extended workflows – and in healthcare, virtually every meaningf workflow is extended. Prior auth spans days. Chronic disease management spans years. Complex coding reviews touch dozens of data points across multiple syste

The leaked architecture addresses this through what VentureBeat described as a three-layer memory system that moves away from the store-everything retrieval approach. The foundational insight is that the agent is explicitly instructed to tr own memory as a hint, not a fact. Before acting on something it believes it know agent verifies against the actual source material. This is a skeptical memory architecture, and it is a genuinely important design philosophy for clinical AI. In clinical settings, the consequences of acting on stale or contradicted memory are just wrong answers in a coding session – they can be adverse patient outcomes, b fraud exposure, or regulatory violations.

The autoDream consolidation engine, which runs as a forked background subag executes a four-phase memory pass: it orients itself by reviewing existing memor structure, gathers recent signal from logs and transcripts, consolidates by writin updating memory files while converting relative timestamps to absolute ones an deleting contradicted facts, and then prunes the memory index to stay under def size limits. The three-gate trigger (24 hours since last consolidation, at least 5 se since last run, acquisition of a consolidation lock to prevent concurrent passes) ensures the system neither over-consolidates nor lets memory drift stale for too l

For healthcare builders, this is a directly applicable pattern. Consider a prior authorization agent managing 50 concurrent cases. Each case has a history of cli

notes, payer criteria, submission attempts, and denial reasons. Storing all of that naively in context kills token budgets and introduces entropy. Running a backgr consolidation pass that strips contradictions, converts relative to absolute facts, maintains a pruned index under defined size thresholds is exactly how you keep agent performant over a weeks-long workflow. The dream system even runs read bash to verify facts against the actual project state before writing memory – tran that to an agent that verifies clinical facts against the EHR before committing th its working memory and you have a pattern that could genuinely reduce the hallucination rate in clinical documentation workflows. The healthcare AI comp that figure this out first will have a durable moat. The ones still using naive RAC no consolidation architecture will be embarrassed by the quality delta within 18 months.

## Multi-Agent Coordination: The Pattern Healthcare Has Been Waiting For

The coordinator mode in the leaked source is a full multi-agent orchestration sy activated via a single environment flag. When enabled, the tool transforms from single agent into a coordinator that manages multiple parallel worker agents. Th orchestration follows four phases: parallel research workers investigate the prob and gather data, the coordinator synthesizes findings and writes specs, worker a implement changes per spec, and verification workers test results. The system p for the coordinator mode explicitly teaches parallelism as a core design value, wi directive that workers are async and independent work should never be serialize when it can run simultaneously.

There is also a shared scratchpad directory for cross-worker durable knowledge sharing, and the coordinator is explicitly prohibited from lazy delegation – it rea worker findings directly and specifies exact next actions, rather than passing the ambiguity downstream. Worker communication happens via structured XML m passing with typed task notification schemas.

The healthcare workflow implications here are significant enough to spend some time on. Take the prior authorization workflow as a concrete example because it canonical case where the industry keeps trying and failing to automate at scale. A coordinator agent receives a prior auth request. It simultaneously spawns one wo to pull and summarize the relevant clinical notes from the EHR, another to retri and parse the payer's clinical criteria for the requested procedure, a third to che any prior submissions or denials on the same patient and case, and a fourth to ve member eligibility and benefit limits. All four run in parallel. The coordinator synthesizes findings into a structured submission spec. An implementation work drafts the prior auth submission. A verification worker checks it against payer fo requirements and flags issues before submission. This is not science fiction – thi directly the architecture in the leaked source, applied to a healthcare workflow.

The same pattern maps to concurrent clinical coding review (parallel workers checking ICD codes, CPT codes, modifier applicability, and payer-specific edits) ambient documentation (workers pulling from different encounter data sources simultaneously while a coordinator synthesizes into a note), and population heal monitoring (parallel workers checking patient panels against multiple protocol criteria while a coordinator surfaces actionable gaps). The companies building th coordination layer for specific healthcare verticals are the ones worth betting on now. The leaked source is essentially a validated reference implementation that removes years of trial-and-error from the architecture decision.

## KAIROS and the Shift from Reactive to Proactive AI in Care Settings

KAIROS is referenced over 150 times in the leaked source and represents the mo forward-looking design pattern in the entire codebase for healthcare application is a persistent, always-running daemon mode – an agent that does not wait to be prompted. It watches activity, writes observations to append-only daily logs, and receives periodic tick prompts that allow it to decide whether to surface someth proactively or stay quiet. It has a 15-second blocking budget for proactive action meaning it self-limits how much it will interrupt the user's workflow before defe

The name comes from ancient Greek. Chronos is clock time. Kairos is the right moment – the opportune instant when action is meaningful. That framing is deliberate. This is not an agent that floods you with notifications. It is an agent designed to intervene at the moment when intervention has the highest expected value.

Healthcare builders should pay close attention to this architecture because it is t technical predecessor to something the industry has been trying to describe but quite build: AI that surfaces clinical insights before the clinician asks. The close current analog is ambient clinical documentation, where AI listens to an encoun passively and drafts notes without requiring a prompt. But ambient documentat still fundamentally reactive – it waits for the encounter to end and then generate output. KAIROS-style architecture goes further: a persistent background agent t monitors patient data streams, flags emerging deterioration patterns, surfaces dr interaction risks when a new order is placed, or alerts a care manager when a hig risk patient's activity data shows anomalous patterns – all without waiting for a

The 15-second blocking budget is actually a really smart design constraint for cl settings. A persistent agent that fires high-priority alerts constantly would indu alert fatigue, which is already one of the most documented patient safety proble hospital medicine – studies have shown that more than 90% of clinical alerts in s hospital systems get overridden by clinicians who have been desensitized by volu A proactive agent with a built-in self-limiting behavioral constraint that defers l confidence or low-urgency interventions is a pattern that could genuinely reduce fatigue rather than exacerbate it. For health tech investors, companies that build proactive-but-self-limiting architecture into their clinical AI products are solvir problem that pure reactive AI cannot solve. The market for clinical decision sup that actually gets used (as opposed to clicked through and ignored) is enormous. architecture to build it is now documented in a public GitHub repo with thousa forks.

## AutoDream: Persistent Memory as a Clinical Infrastructure Problem

The autoDream system deserves its own section separate from the broader mem architecture discussion because of what it reveals about how to think about pers memory as an infrastructure problem rather than a feature. The leaked source tr memory consolidation as a background service with defined triggering condition explicit phase structure, size constraints, and a locking mechanism to prevent ra conditions. This is not bolted-on memory. It is a first-class infrastructure compo with the same design rigor as any stateful backend service.

The four phases (orient, gather signal, consolidate, prune and index) map remark well to how clinical knowledge management should work in a longitudinal patie care context. Orient means the agent reads its current memory structure before anything, establishing baseline understanding of what it already knows and how knowledge is organized. Gather signal means it identifies what has changed sinc last consolidation pass – new labs, new notes, new orders, new encounter record Consolidate means it updates its durable knowledge store, specifically convertin relative temporal references (three days ago, last visit) to absolute ones (March 2 2026, encounter 447291), which is critical for clinical reasoning that depends on accurate timelines. Prune and index means it removes stale information, resolves contradictions, and keeps the index size manageable for future session efficiency

The contradiction resolution step is particularly important for healthcare. Clini records are full of contradictions – a patient who is listed as a non-smoker in the problem list but has documented tobacco use in encounter notes, a medication l that includes a drug the patient reported stopping six months ago, an allergy list conflicts with a prescribed medication. An agent that just stores everything naiv will eventually try to act on contradictory information and produce unreliable outputs. An agent with active contradiction resolution in its consolidation cycle surfaces these conflicts explicitly rather than silently working around them, whi the behavior you need in a HIPAA-regulated environment where auditability of reasoning is increasingly a compliance requirement.

The size constraints in the autoDream implementation (memory index under 20 and approximately 25KB) also offer a useful design forcing function. Healthcare builders tend to assume that more context is always better, but long context win

do not solve context entropy – they just defer it. The discipline of maintaining a pruned, well-organized memory index under defined size constraints forces the a to make explicit decisions about what information is durable versus transient, w is exactly the kind of structured knowledge management clinical AI needs.

## Permission Architecture and Why Healthcare Builders Should Steal This Pattern

The permission system in the leaked source is probably the most directly transfe component for healthcare builders, and also the most underappreciated in most general tech coverage of the leak. The system classifies every tool action as low, medium, or high risk. It gates protected files from automatic modification. It inc path traversal prevention that handles URL-encoded attacks, Unicode normaliza exploits, backslash injection, and case-insensitive path manipulation. It has four distinct permission modes: default (interactive user prompts), auto (ML-based au approval via a transcript classifier), bypass (skip checks), and a mode called yolo ironically denies everything.

There is also a permission explainer component that generates a natural languag explanation of what a tool action will do and why it carries risk, before the user approves it. That explanation is itself generated by the model – meaning the AI i explaining its own actions in plain language as a precondition for execution. For healthcare, this is an extremely relevant pattern. The core HIPAA and emerging governance requirement for clinical AI is explainability – the ability to show wh agent did, why it did it, and what access it exercised in the process. An architect where every high-risk action generates a human-readable explanation before execution, and where that explanation is logged, is a compliance architecture as as a product architecture.

The specific protected file list in the source (git configuration, shell profiles, MC configuration files) maps directly to a concept healthcare builders should formal protected data objects. In clinical AI, that list would include patient records, co

flags, medication orders, and anything touching billing or coding. The principle same – certain objects are too consequential to allow automatic modification, regardless of how confident the agent is. The leaked codebase draws that line explicitly in code. Healthcare builders should draw equivalent lines explicitly in tool governance frameworks, not leave it to the model's judgment.

The YOLO classifier, despite the irreverent name, is a genuinely interesting component: it is a fast ML-based permission decision system that uses session transcripts to decide automatically whether a pending action should be approve without interrupting the user. In healthcare settings, the equivalent would be an that auto-approves routine, low-risk documentation actions while escalating any touching orders, prescriptions, or billing to human review. This is a much more sophisticated approach than blanket human-in-the-loop requirements, which ad friction without always adding safety. The leaked source shows that this kind of permission system is buildable with current ML capabilities and that the major developing the most widely-used coding agent in production has shipped it.

## Feature Gating, Dead Code Elimination and Staged Rollouts for Regulated Environments

One of the less-discussed but highly practical architectural decisions in the leak source is the compile-time feature gating system. Features are controlled via con time flags that the Bun bundler constant-folds and then dead-code-eliminates in external builds. Branches behind inactive feature flags are not just inactive at ru – they are physically absent from the compiled output. This means the external production build has a smaller attack surface, no accidental exposure of internal feature surfaces, and no runtime flag-checking overhead. The internal build has full feature set. The two builds share source but produce fundamentally differen artifacts.

For healthcare software teams, this is a compelling pattern for managing staged rollouts in a regulated environment. The FDA's guidance on AI/ML-based softwa

a medical device (SaMD), and the associated predetermined change control plan requirements, create real compliance headaches for teams trying to iterate quick AI features. A compile-time gating system that produces provably different build different deployment contexts (say, a research use only build versus a clinical dec support build versus a diagnostic aid build) offers a cleaner story for validation a regulatory submission than a runtime flag system where features could theoretic be toggled in a production environment. Regulators like hard boundaries. Dead elimination creates hard boundaries. This is worth stealing.

The GrowthBook-based runtime gating layer running alongside the compile-tim system also reveals a mature dual-track approach: compile-time elimination for structural features that differ across deployment types, runtime flags for increm behavioral tuning within a deployment type. Healthcare builders operating in m site deployment environments (which is most of them at any meaningful scale) sh think carefully about which of their feature variations belong to each layer. The leaked source shows that even a company with the engineering depth of the organization that built this system found both layers necessary for production operation.

## What the Roadmap Signals for Health Tech Investment Theses

The leaked source contains a list of undisclosed API beta headers representing features not yet public, and for investors, this list is essentially a forward-lookin product roadmap for the most widely deployed AI agent platform. A few are particularly relevant for healthcare investment theses.

The context-1m beta header points to a 1M token context window, dated August in the source. For healthcare, this is significant because the workflows that most AI assistance – complex case management, multi-encounter longitudinal analysis comprehensive chart review – are exactly the ones where current context limits f fragmentation, which introduces errors. A 1M context window changes the architecture of what is possible for longitudinal clinical AI. The companies build

on the assumption that 200K context is the ceiling may need to revisit their architecture assumptions sooner than they think.

Task budgets (task-budgets-2026-03-13) and effort control (effort-2025-11-24) ar in the unreleased beta headers. For healthcare, these translate directly to cost governance and workflow time management, both of which are top-tier procurer objections for clinical AI. An agent that can be given an explicit task budget – sp no more than $X in compute to complete this prior auth review – and an explicit effort level – do a quick check versus an exhaustive review – is far more deployal a healthcare operations context than an agent with unbounded resource consum These are the kinds of controls that make AI palatable to CFOs and CMOs simultaneously.

Redacted thinking (redact-thinking-2026-02-12) is interesting from a liability an compliance angle. The ability to expose or suppress the agent's reasoning chain l on deployment context matters a lot for healthcare companies navigating the du pressures of explainability requirements (show your work) and IP protection (bu all of your work). An AI that drafts prior auth appeals might be required to expl clinical reasoning to the payer, but a health system might not want to expose its internal decision logic. Configurable reasoning chain visibility is a feature that resolves a genuine compliance tension.

AFK mode (afk-mode-2026-01-31) and the associated transcript classifier for aut approval map directly to the unattended agent use cases that are increasingly the target of healthcare AI investment – autonomous claim scrubbing, overnight cod review, background eligibility verification. These are tasks where human-in-the- approval at the action level is neither practical nor necessary, but where you still audit trails and anomaly detection. The auto-approval classifier architecture in t leaked source is a validated approach to that problem.

## The So-What: Practical Takeaways for Builders and Investors

The leaked source is not a gift to competitors in the narrow sense that they can c Anthropic's UI or business model. It is more valuable than that – it is a validated reference architecture for building production-grade AI agents, written by a tea had to solve the same problems healthcare builders are fighting with right now. Context entropy. Parallel workflow orchestration. Persistent memory. Tiered permissions. Proactive versus reactive UX modes. Staged rollouts in regulated environments.

Healthcare has a tendency to try to reinvent every pattern from scratch because t domain is specialized enough that practitioners distrust general solutions. Some that instinct is right. HIPAA is real. Clinical liability is real. The regulatory environment for AI in clinical settings is genuinely more complex than general enterprise software. But the core agent architecture problems – how do you man context over long workflows, how do you parallelize task execution safely, how d build permissions that balance automation with human oversight, how do you rc features in a validated way – are not healthcare-specific problems. They are softv engineering problems. And the leaked source shows how a very good engineerin team solved them.

For investors evaluating health tech AI companies right now, the leaked source i useful benchmark. Any company pitching a clinical AI agent should be able to articulate how it handles context entropy over multi-session workflows. If the ar is basically just storing everything in a long context window and hoping the mod handles it, that is a red flag. Any company building autonomous clinical workflo agents should be able to describe its permission and risk classification system. It answer is human approval on every action or conversely no governance layer at a both are architectural immaturity signals. Any company targeting the unattende agent market in healthcare should have a story for how its auto-approval logic w and how it maintains audit trails.

The companies that will win in clinical AI over the next three to five years are th treating agent infrastructure with the same rigor that the leaked source reveals - set of deeply considered, production-hardened engineering problems, not as a th wrapper around a foundation model API. The bar is now visible. It is high. And

March 31, 2026, anyone who wants to read the specs is about 1,900 GitHub fork
from having the full picture.

← Previous

## Discussion about this post

Comments Restacks

Write a comment...