PEOPLE IN THE LOOP

The Library

AI terminology in plain words: agents, Claude Code, prompts, tools, context, governance, and the human in the loop.

People in the Loop

Nina Thomas Living edition

hirepitl.com

Contents

Every term here is explained the way you would explain it to a smart colleague who does not code: a plain-words definition, and an analogy where one helps. These are the same shelves published in the Library room, bound so you can keep them. The Library grows; the download is refreshed from the shelves every hour.

01

Governance & Guardrails

10 terms

AI governance

The rules for how an organization uses AI: who may use it, for what, with which data, and who is accountable when it goes wrong. Guardrails are how those rules get enforced.

Approval gate

A required human yes before an agent's action takes effect: sending, spending, publishing, deleting. The loop pauses until a person signs off.

Audit trail

The record of what an AI system did and why: every action, tool call, and decision logged so a person can reconstruct it later. For compliance purposes, if it isn't logged, it didn't happen.

Data sensitivity

Knowing which information must never reach an AI tool (regulated personal data, client confidences, secrets) and anonymizing or withholding it before use. The redact-before-upload habit.

Guardrails

The technical limits that stop an AI system from doing what policy forbids: permission rules, blocked actions, filtered inputs and outputs. Policy says what; guardrails make sure.

Think of it as: Bowling-lane bumpers: the ball can wander, it just can't leave the lane.

Hallucination

When a model states something false with full confidence, including invented citations, numbers, or names. It isn't lying; it's filling gaps convincingly. This is the reason verification and human review exist.

Human in the loop

A person stays close enough to an AI system's work to check it, correct it, and decide when it can run on its own versus when a person must step back in. It is a design choice about where human judgment sits in a process, not a slogan.

Think of it as: A student driver with an instructor who has their own brake pedal.

Human on the loop

The lighter version of human in the loop: the AI acts on its own and a person reviews afterward, sampling results and stepping in when something looks off. In the path versus watching the path.

Least privilege

Give an AI system the smallest access it needs and nothing more.

Think of it as: The intern gets a key to the supply closet, not the vault.

Red teaming

Deliberately attacking your own AI system: prompting it to misbehave, leak data, or break its rules, so you find the failure before a stranger does.

02

Agents & Orchestration

22 terms

Adaptive decomposition

Splitting open-ended work (research, investigation) where findings reshape what the next subtask should be.

Agent

An AI model running in a loop with tools: it thinks, takes an action, looks at the result, and goes again until the job is done.

Think of it as: A new employee with a phone and a to-do list, not just a brain in a jar.

Agent SDK

Anthropic's framework that runs the agent loop for you: manages subagents, ships built-in tools. The raw API makes you write the loop yourself.

Agentic loop

The cycle your code runs around a stateless model: gather context, take action, verify results, repeat.

Think of it as: Cooking from a recipe: read, do a step, taste, adjust, continue.

API (Application Programming Interface)

The doorway a program uses to talk to another program or service over the web: send a request, get a response. Claude itself is reached through one.

Think of it as: A restaurant counter: you order from the menu, the kitchen does the work, your food comes back. You never go into the kitchen.

Call-inspect-execute-append cycle

The full agent turn: call the API, inspect the stop_reason, execute the requested tool locally, append the result to the history, call again. Skip inspect and the agent goes silent; skip append and it repeats the same tool call forever.

Coordinator (orchestrator)

The manager agent that breaks a big job into pieces, hands them to subagents, and combines the

results.

end_turn

A stop_reason value meaning "I'm done." The loop ends here, and only here.

fork_session

Starts a new session as a copy of an existing session's context, so parallel work can branch from the same starting point.

Hook

A small script that runs automatically at a fixed moment in an agent's work. Rules that must ALWAYS happen belong in hooks. Prompts are requests; hooks are law.

Hub and spoke

The orchestration shape where workers talk only to the manager, never to each other. Keeps the system predictable and debuggable.

Think of it as: An air-traffic control tower: every plane talks to the tower, not to other planes.

PreToolUse / PostToolUse

The two classic hook moments: right before a tool runs (gate it) and right after (enforce follow-ups, like logging every change).

Refinement loop

A pass where an agent's work is checked for coverage and gaps before the final version ships.

SDK (Software Development Kit)

A code library that wraps an API so developers do not have to hand-build every request. Anthropic ships official Python and TypeScript SDKs for Claude.

Think of it as: The meal kit version of the restaurant: same food, but the fiddly prep is done for you.

Sequential decomposition

Splitting a job into steps where each needs the previous step's output. Use when order matters.

Stateless

The model keeps no memory between API calls. Every call starts blank; your code supplies the history each time.

Think of it as: A brilliant consultant with amnesia: brief them fully at the start of every meeting.

stop_reason

The machine-readable label on every API reply saying why the model stopped. Your loop should be driven by this label, never by reading the model's prose or counting turns.

Subagent

A worker agent spawned for one piece of a job. Returns a compact summary to the coordinator, which protects the coordinator's context budget.

Targeted retry

When one subagent fails, retry only that one. A blanket retry re-runs steps that already succeeded, re-firing their side effects (like duplicate database writes).

Task tool

How a coordinator in the Agent SDK/Claude Code actually spawns a subagent.

The four surfaces

The map of the Claude ecosystem: the Claude API (one stateless HTTP endpoint plus SDKs), the Agent SDK (runs the loop), Claude Code (a terminal agent with file access), and MCP (the standard plug for outside tools).

tool_use

A stop_reason value meaning "run this tool for me and send back the result." The loop continues.

03

Claude Code

14 terms

.claude folder

The project's Claude configuration home: rules/ (always-on instructions), commands/ (reusable /name prompts), skills/ (bigger packaged abilities).

.mcp.json vs ~/.claude.json

Where tool connections live: .mcp.json in the project is shared with the team; ~/.claude.json is personal. Same logic as project vs personal CLAUDE.md.

@claude (GitHub Actions)

Mentioning @claude on a GitHub issue or PR to trigger automated review or fixes through a GitHub Action.

@import

How a CLAUDE.md pulls other files into itself, so instructions can be split up but load together.

Claude Code

Anthropic's terminal agent. It works inside a folder or repository with real file access through tools like bash and grep.

CLAUDE.md

The instruction file Claude Code reads at session start. Three levels: user (~/.claude/, personal, never version-controlled), project (repo root, committed, team rules), and directory (subfolder-specific rules).

dangerously-skip-permissions

The flag that bypasses every permission prompt. Never use it, except in strictly controlled, low-risk environments like a locked-down CI job.

Headless mode (-p)

Runs Claude Code non-interactively with one prompt, for scripts and CI/CD. Pair with --output-format json for machine-readable results.

Permission rules (allow / ask / deny)

Per-tool access control for Claude Code: allow silently, ask first, or deny outright, for tools like bash, read, edit, and web fetch.

Plan mode

Makes Claude Code propose an approach for approval before touching anything. Right for big risky changes; overkill for one-liners.

Sandboxing

Restricting what an agent's shell can touch on the host machine (tools like bubblewrap and socat). Containment beats trust.

Session commands (resume, fork, compact, clear, rewind)

Managing a session's life: re-enter an old one (resume), branch its history (fork), shrink token use (compact), wipe it (clear), roll back to an earlier turn (rewind).

SKILL.md frontmatter

The header block of a skill file. Can set context: fork, allowed-tools, and argument-hint.

Slash command

A reusable prompt saved as a markdown file in .claude/commands, triggered by typing /name. Its YAML frontmatter can restrict tools and hint at arguments.

04

Prompts & Structured Output

8 terms

Confidence calibration (field-level)

In extraction pipelines, scoring confidence per field (99% on the date, 40% on the vendor) so low-confidence fields route to human review. Different from escalating a conversation on the model's mood, which is always wrong.

Few-shot examples

Showing 2 to 4 examples of the output you want inside the prompt. Beats describing the format in words.

JSON schema (as a tool)

Defining the exact shape of data you need as a tool definition, so the model fills in a form instead of writing an essay you have to parse.

Mechanically checkable rules

Replacing vague adjectives ("be careful") with categorical rules a checker could verify. Vague adjectives produce false positives.

Message Batches API

Submit thousands of independent prompts as an offline batch: results within 24 hours at 50% of the cost. Never for anything a person is waiting on.

Think of it as: Dropping laundry at the wash-and-fold overnight instead of standing at the machine.

Nullable fields

Marking schema fields optional so "not present" is a legal answer. A required field with missing source data gets fabricated. Required + missing = made up.

tool_choice

Per-request control of tool use: auto (model decides), any (must use some tool), a named tool (must use that one), or none.

Validation-retry loop

Check structured output against the schema; if invalid, send the error back for a corrected retry (usually succeeds), then fail loudly. Never silently accept bad data.

05

Tools & MCP

7 terms

MCP (Model Context Protocol)

The open standard that lets outside services (databases, browsers, CRMs) offer themselves to any compliant model as tools, with no custom glue code.

Think of it as: The USB standard for AI tools: one plug shape, everything connects.

MCP server / client

The server offers the tools; the client (like Claude Code or claude.ai) uses them.

Remote transport (SSE / streamable HTTP)

The MCP connection for a server on a different host, with authentication. Older material says SSE; the spec has been moving toward streamable HTTP. The judgment tested: local = stdio, remote = network transport.

Resources vs Tools (MCP)

Resources are read-only things a server exposes; Tools take actions. Knowing which is which decides what an integration can and can't do.

stdio transport

The MCP connection to use when server and client are on the same machine. Using a network transport on localhost is a classic wrong answer.

Structured errors (isError, errorCategory, isRetryable)

When a tool fails, return machine-readable error fields so the agent can plan recovery. Retry a timeout; never retry access-denied. "Something went wrong" tells the agent nothing.

Tool description

The name and description are how the model decides when to use a tool. Write them like API documentation, including a disambiguation rule for when to use THIS tool instead of a similar one.

06

Context & Reliability

9 terms

Context budget

Treating context like money: spend it on what matters, don't dump raw tool output into it, make subagents return summaries.

Context window

Everything the model can see at once: the conversation, documents, tool results. Finite, and managing it is a real skill.

Escalation triggers

Objective, checkable conditions for handing off to a human: the customer asked, the request is outside policy, or no progress is being made. Never the model's detected sentiment or self-reported confidence.

Fact block / case block

A pinned, structured block of durable facts (customer, order number, issue) re-anchored at the end of the context every turn, instead of hoping the model re-finds them on page 40.

LLM (Large Language Model)

The engine under Claude, Copilot, and their peers: a model trained on enormous amounts of text that predicts language, which turns out to be enough to draft, summarize, reason, and converse.

Lost in the middle

Models pay the most attention to the start and end of a long context. Critical facts buried in the middle get missed.

Prompt caching

Marking static sections (system prompt, few-shot examples) with cache breakpoints so repeat calls reuse them, cutting cost up to ~90% on the cached portion.

Scratchpad

Having a long-running agent write intermediate notes to a file instead of holding everything in context.

Token

The unit models actually read and write: chunks of a few characters each. Context windows, pricing, and limits are all measured in tokens, roughly three quarters of a word apiece.

Think of it as: Lego bricks of language: the model builds and bills by the brick, not by the sentence.

07

Certification

5 terms

CCAR-F / CCDV-F / CCAO-F / CCAR-P

The exam codes for Anthropic's four certifications: Architect Foundations, Developer Foundations, Associate Foundations, and Architect Professional.

Claude Partner Network (CPN)

Anthropic's partner program. Tiers (Select and up) require active certified individuals; Architect and Developer certs count, Associate does not.

Minimally qualified candidate (MQC)

The person the exam is calibrated to: someone who just barely deserves to pass. The standard-setting anchor behind the cut score.

Pearson VUE / OnVUE

The proctored testing provider: test centers, or OnVUE for taking it from home with a webcam proctor.

Scaled score

Exam results reported on a 100 to 1,000 scale; 720 passes. Roughly 43 to 45 of 60 questions, so about 17 wrong answers still pass.