

Agent Atlas

One agent first. More only when the parts are independent.

Part A agents

From a request to an agent

A chat answers once. An agent is a model in a loop, until the job is done:

Think → Use a tool → Read the result → Repeat until done

Where agents run

Everyday agent: Claude (Cowork) / ChatGPT Work — your files & apps.

Code agent: Claude Code / Codex — your repo & terminal.

Tools & MCP: how an agent acts

Every tool definition sits in the context on every request. **MCP:** one plug per app (Jira, Figma, GitHub...).

Connect what the task needs, load the rest on demand: ~77K → ~8.7K tokens in Anthropic's example.

Autonomy: who says yes?

Easy to undo (edit, rename, draft): let it run. **Hard to undo** (delete, send, pay, deploy): keep your OK.

Claude Code: Manual → Accept edits → Auto → Bypass. Codex: Read Only → Ask for approval → Approve for me → Full access.

Subagents: a fresh context per side job

A subagent reads a lot and returns a short summary (often 1–2K tokens), so the main context stays focused. **The catch:** each one spends its own tokens, and a summary drops details.

Three ways to split the work

Sequential: steps depend on each other. **Parallel:** independent parts at the same time. **Coordinator:** parts you can't know up front.

Part B effort

Agentic effort: beyond thinking longer

Plan → Split → Run in parallel → Check → Combine

Effort, two ways

S2: one model thinks longer. **S4:** the work itself gets organized. You set the goal and the checks.

Read in parallel, write in one place: subagents read and review; one agent writes; tests after each step; one summary.

Part C the hype

Four questions for any AI post

Compared with what? What did it cost? Does it really work? Who says so?

Don't trust a post, trust a source.

Myth Buster

More agents = better results → Several agents usually use 3–10× the tokens of one (Anthropic). Start with one; add more only when the parts are truly independent.

A prompt to keep

Prepare the release note and the FAQ update from these files: the spec, the support tickets, the FAQ.

First show me a short plan and wait for my OK.

Read the three files separately, in parallel if you can.

Check every sentence you write against the file it comes from, and list what doesn't match.

Finish with one result: the release note, the FAQ changes, what's still open.

Four sessions, one thread: whatever the tool, ask **what does the model see right now?**