

qbash

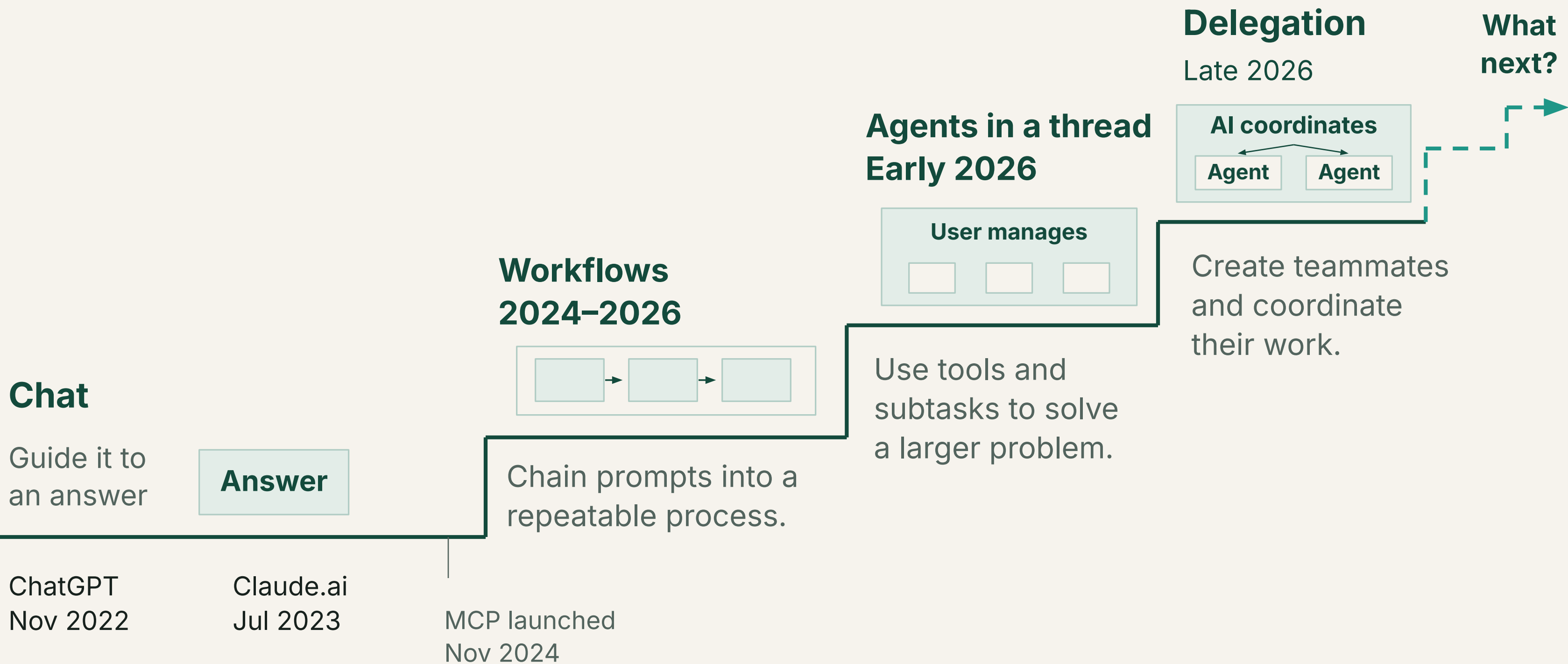
AI and business strategy

What we are seeing in the industry and what it means for the businesses we run

1. What is changing
2. What does that mean?

Capabilities keep making dramatic jumps

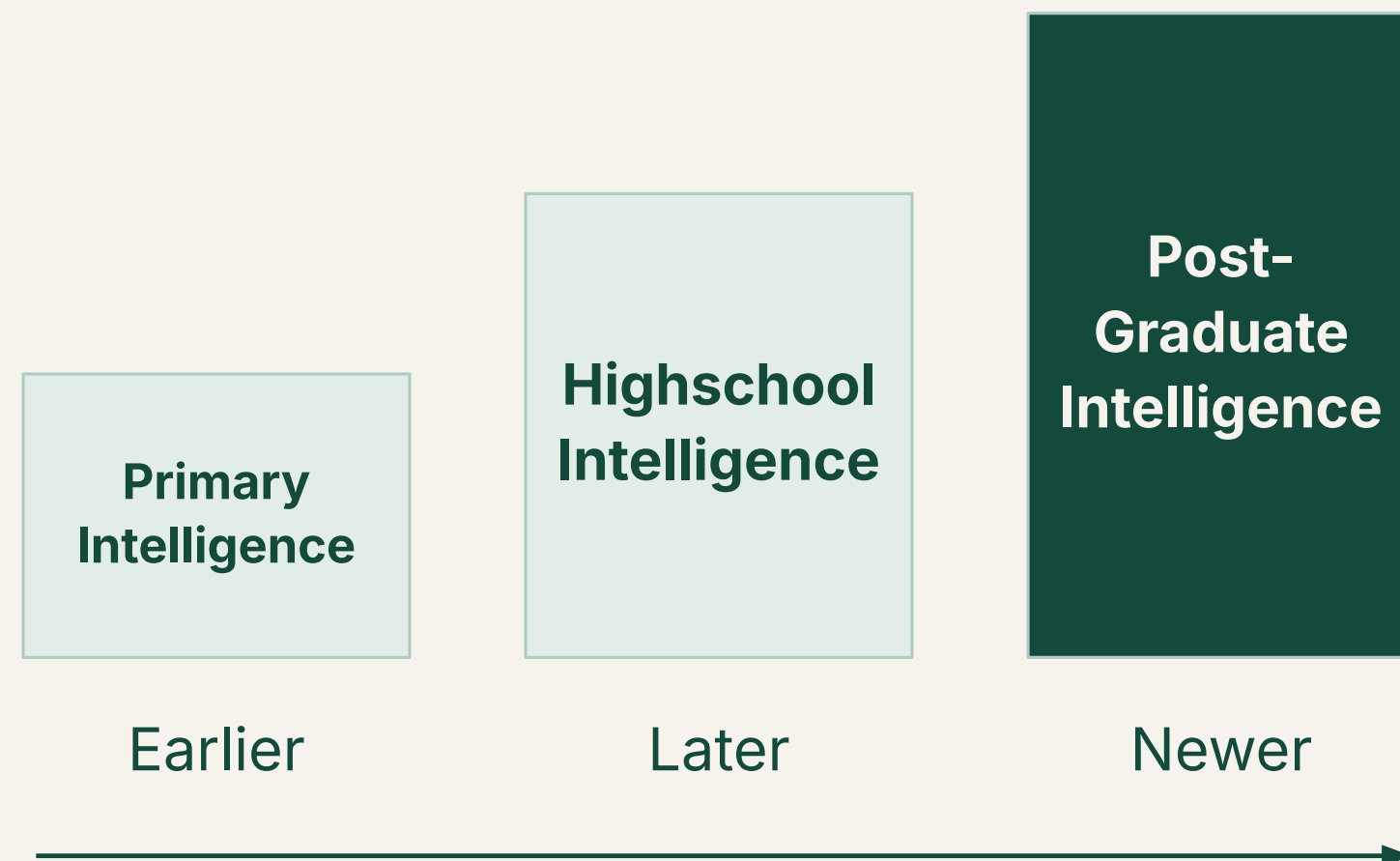
I have never seen capabilities accelerate this fast. Work that took days or weeks keeps collapsing into minutes. The scope of what we can ask for keeps expanding.



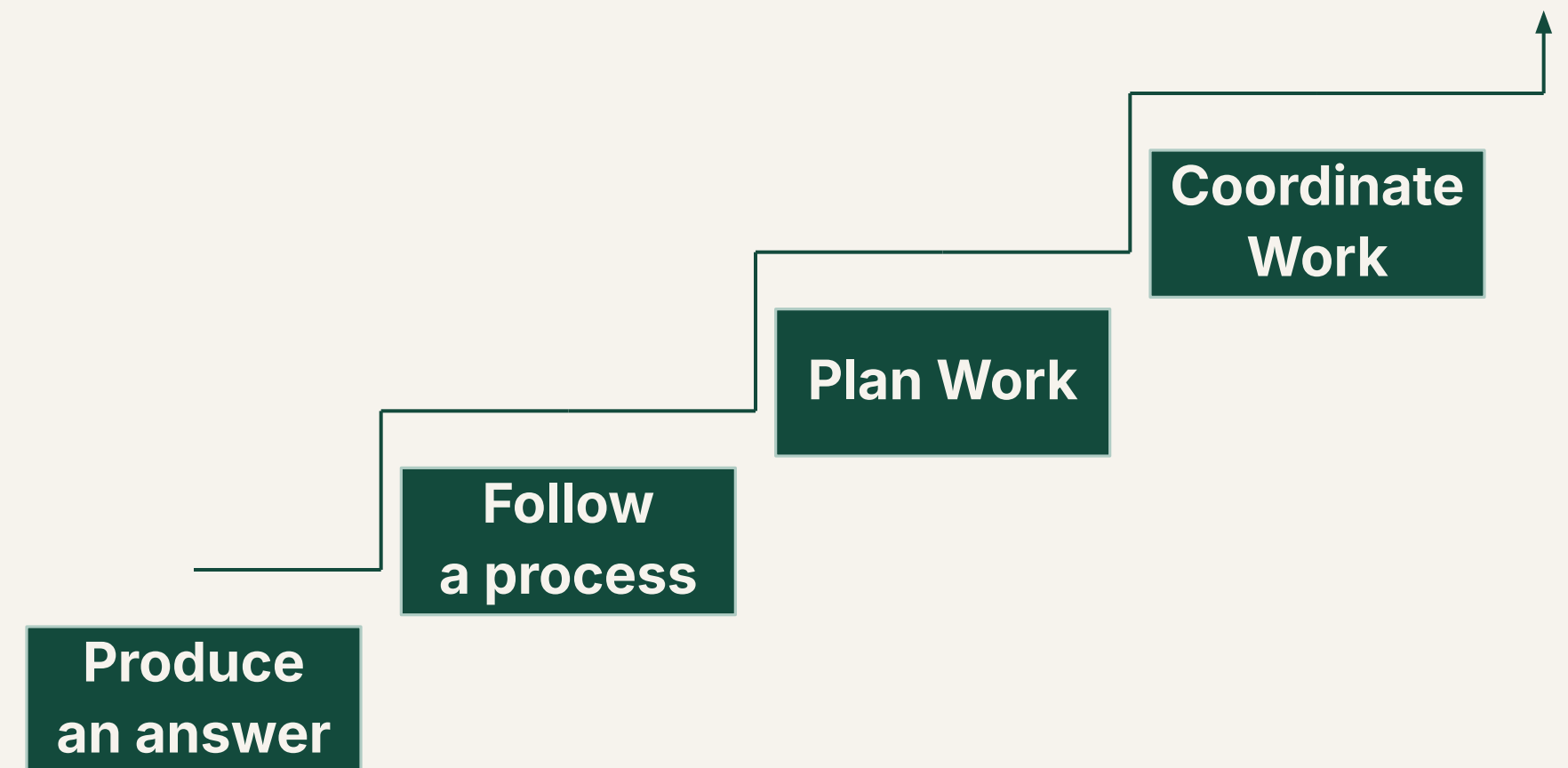
Depth vs breadth

It is important to understand that while models continue to get smarter, many companies overlook the increasing scope of work and the layers of a business that AI can absorb.

Depth: more capable models



Breadth: more of the work



Adobe. Office. Outlook.

As AI tooling autopilots with delegation more pillars fall. Tools like REA can clone complex software in days, reverse engineer old binaries, PlayStation Games, Eleven Labs, etc.

This same process can replicate some service offerings, and it is just a *function of time*.

2-3 OCT

7 Adobe-style apps

Photoshop, Illustrator, Premiere and four more.

7-8 OCT

Word. Excel. PowerPoint.

WordCraft, GridCraft and DeckCraft released.

12 OCT

Outlook-style MailCraft

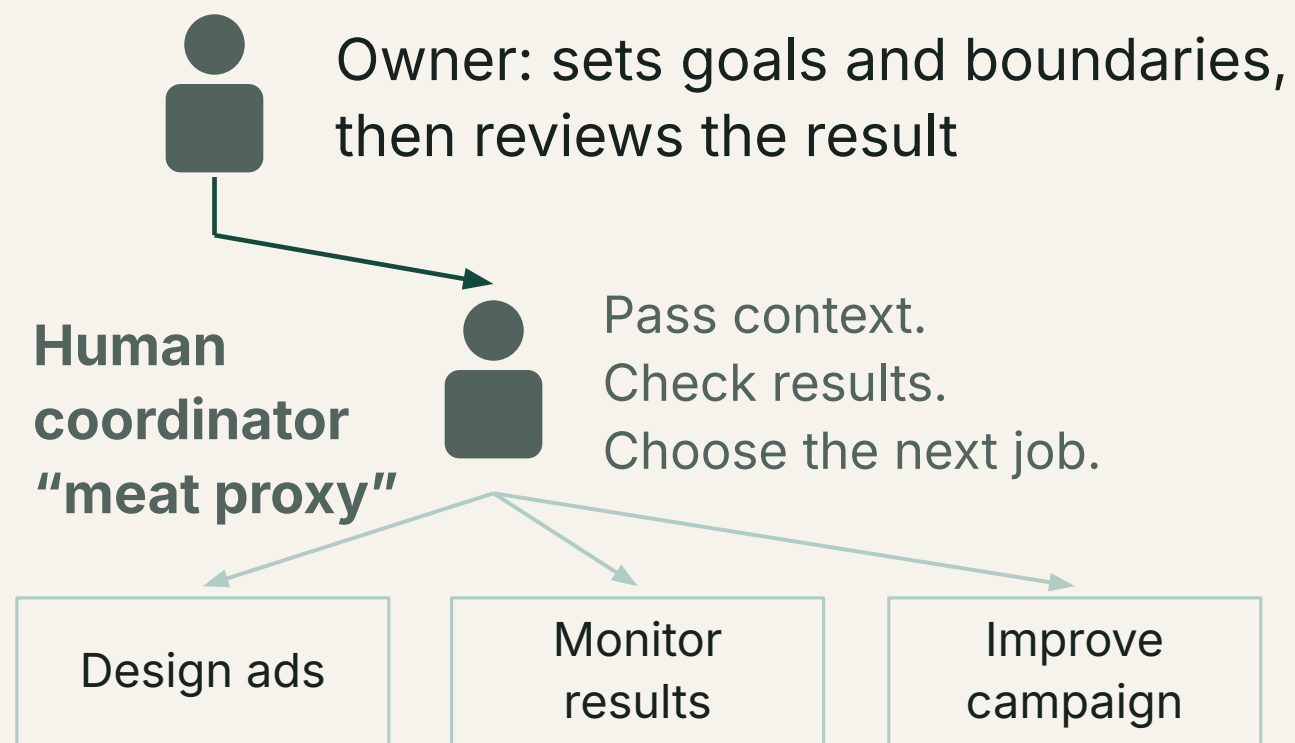
Mail, calendar and contacts.

Next stepwise increase: delegation

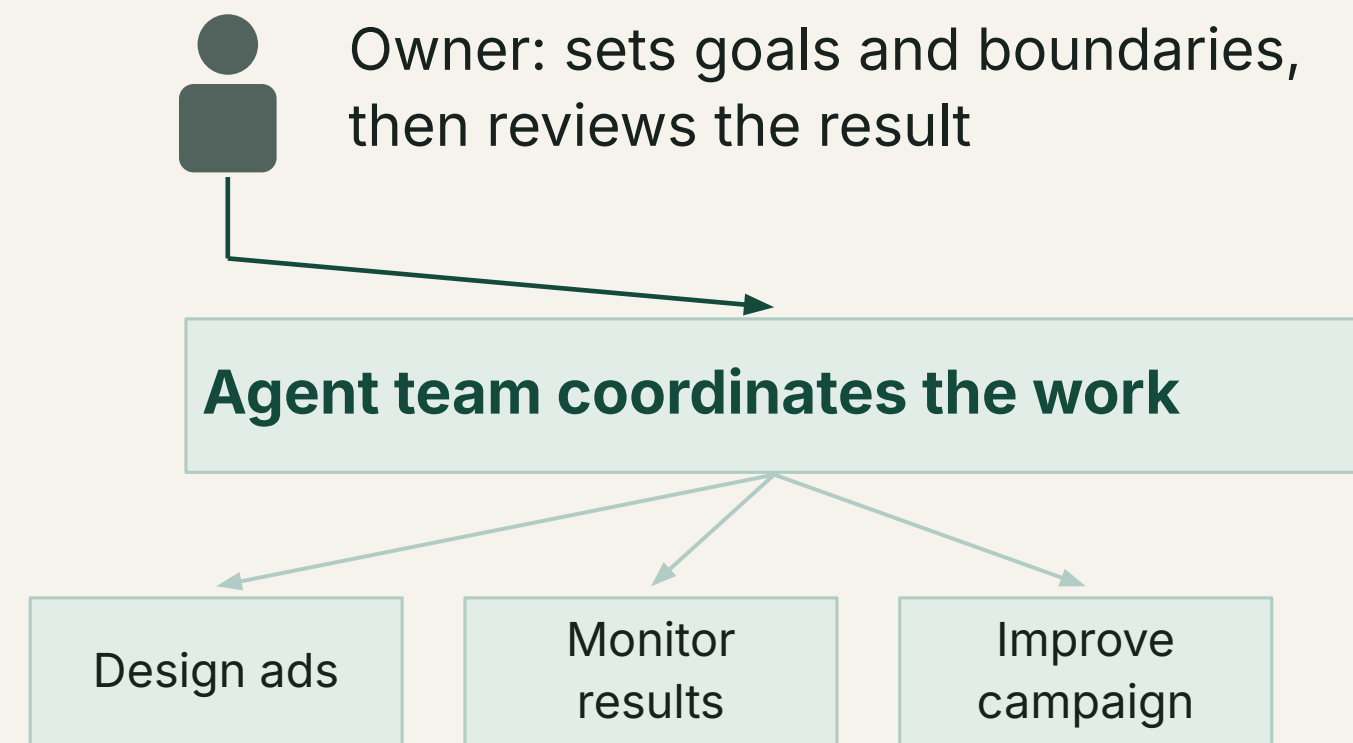
AI driven delegation is very near consumer-ready. AI moves up a layer: from completing work to taking on the coordination of work a person used to provide: creating teammates, passing context and responding to their results.

Automation moves from individual jobs to *whole functions*. Design an ad for this keyword becomes develop and monitor campaigns.

Human coordinates



AI coordinates



What does that mean?

"Software is eating the world" reaches further into the work a business organises and delivers.

Inside the business

Design reusable operations. Build a team that can keep extending them.

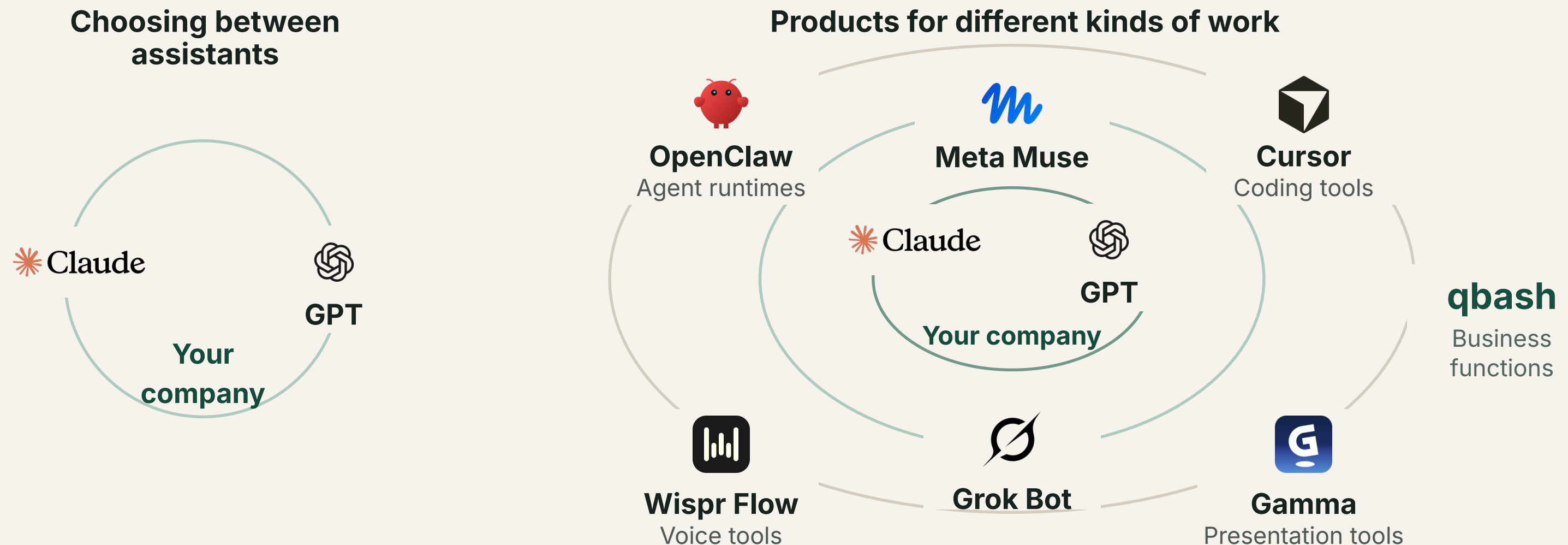
In the market

More of what customers buy can be reproduced. That changes who pays, what they pay and who competes.

In the market: Customers are scattered

Companies once largely chose between Claude and ChatGPT. They now use specialised products for different jobs. Delegation accelerates those differences, so reaching customers keeps changing.

An agent the customer never interacts with may choose your tools and integrations. Your work can contribute to the result without the customer knowing you exist. API or MCP access makes you callable. It does not guarantee visibility or distribution.



Delegation expands the companies AI absorbs

Businesses sell more than individual outputs. They sell the ability to organise and deliver the work behind them. Delegation makes more of that operation reproducible too.

Companies that rely on simple ancillary services to drive average prices are under particular threat - i.e. accounting firms with bookkeeping fees, filing fees.

**AI manages
the process**

Bookkeeping

Collect records, reconcile and flag exceptions

Research

Investigate sources and deliver a brief

Knowledge work

Combine evidence and prepare a report

Form filing

Gather information and complete a filing

**Accounting firms with
subservices**

Legal firms

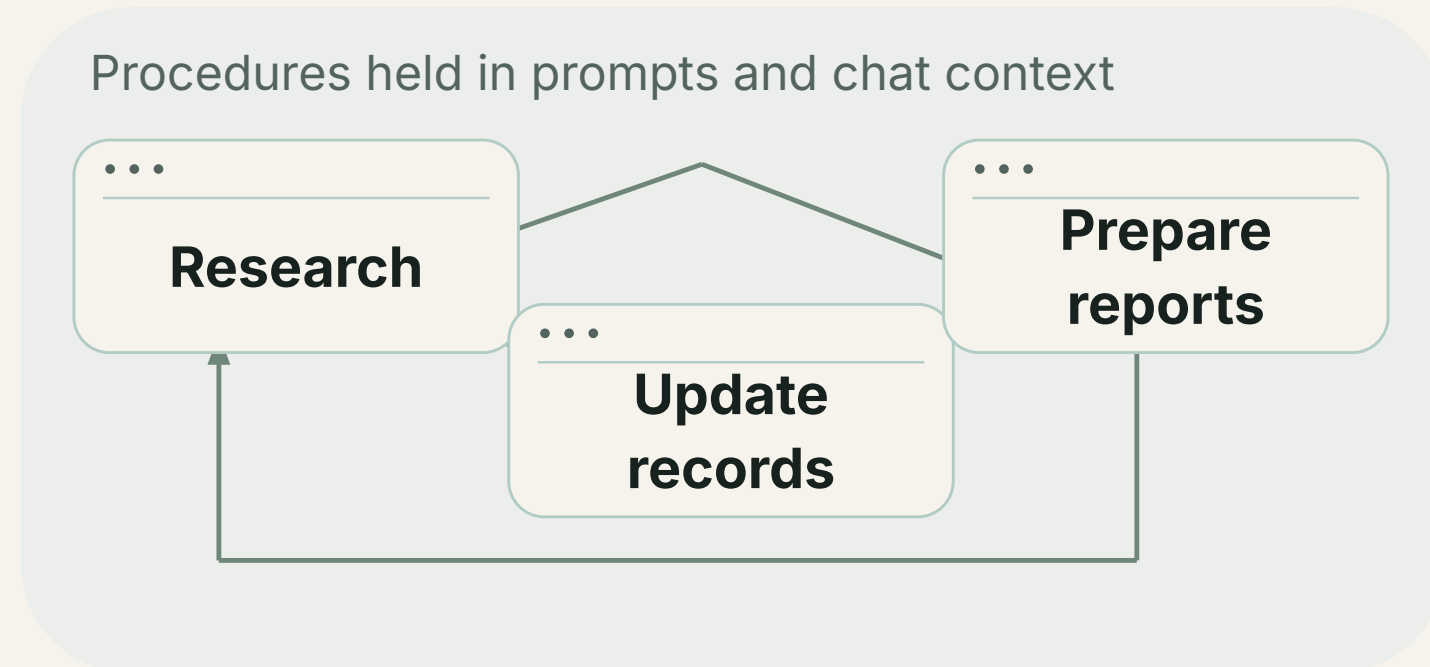
Software developers

Research firms

Functional Business design is being rewarded

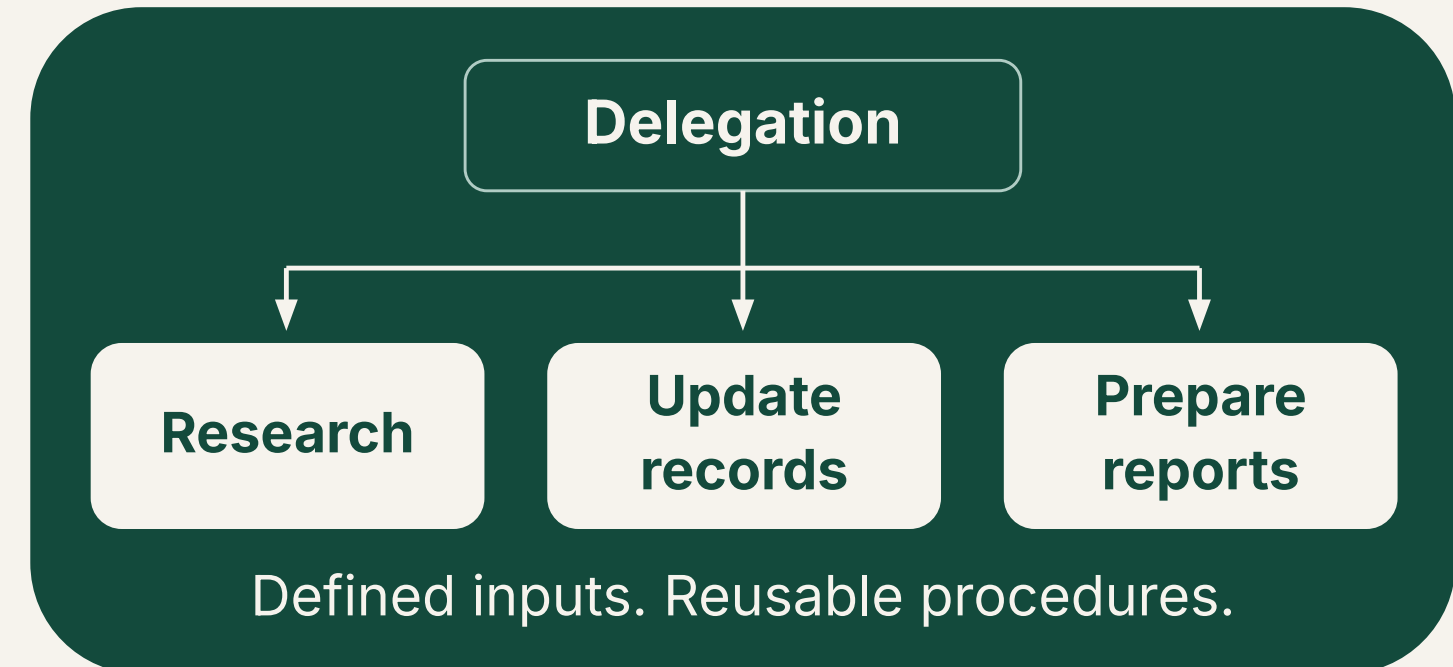
Companies that have deliberately designed their business as a set of reusable functions are better equipped for the next capability increase and continue to compound their work. Companies that have work married-in chat tools and Agents are not able to take the same advantage

Disconnected chat automations



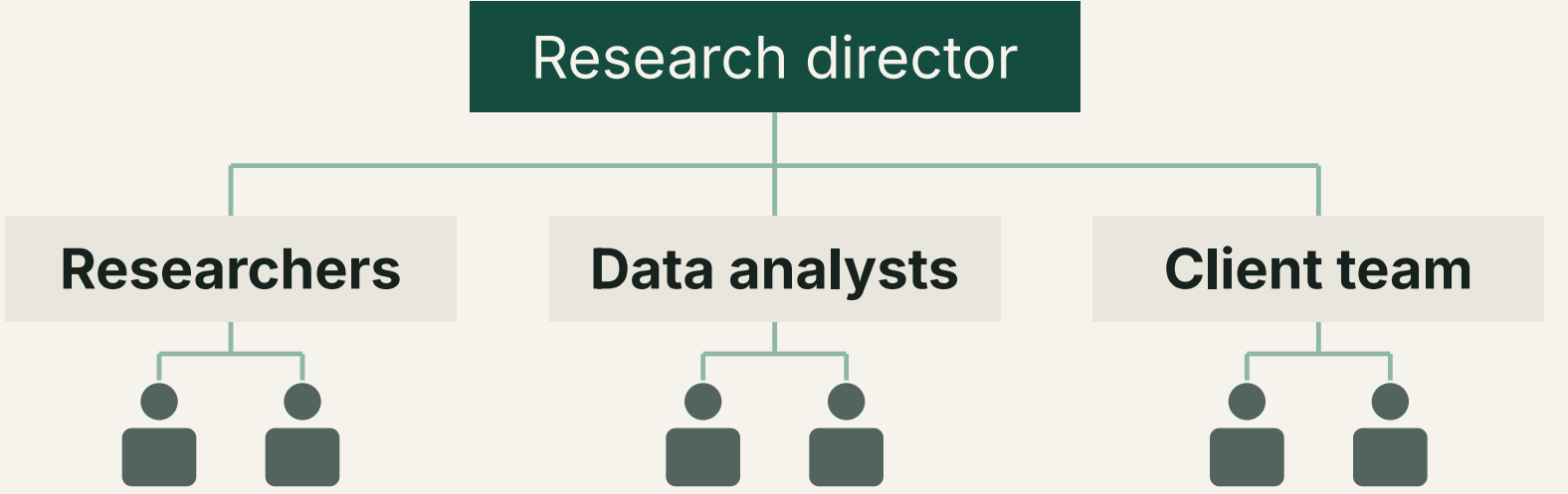
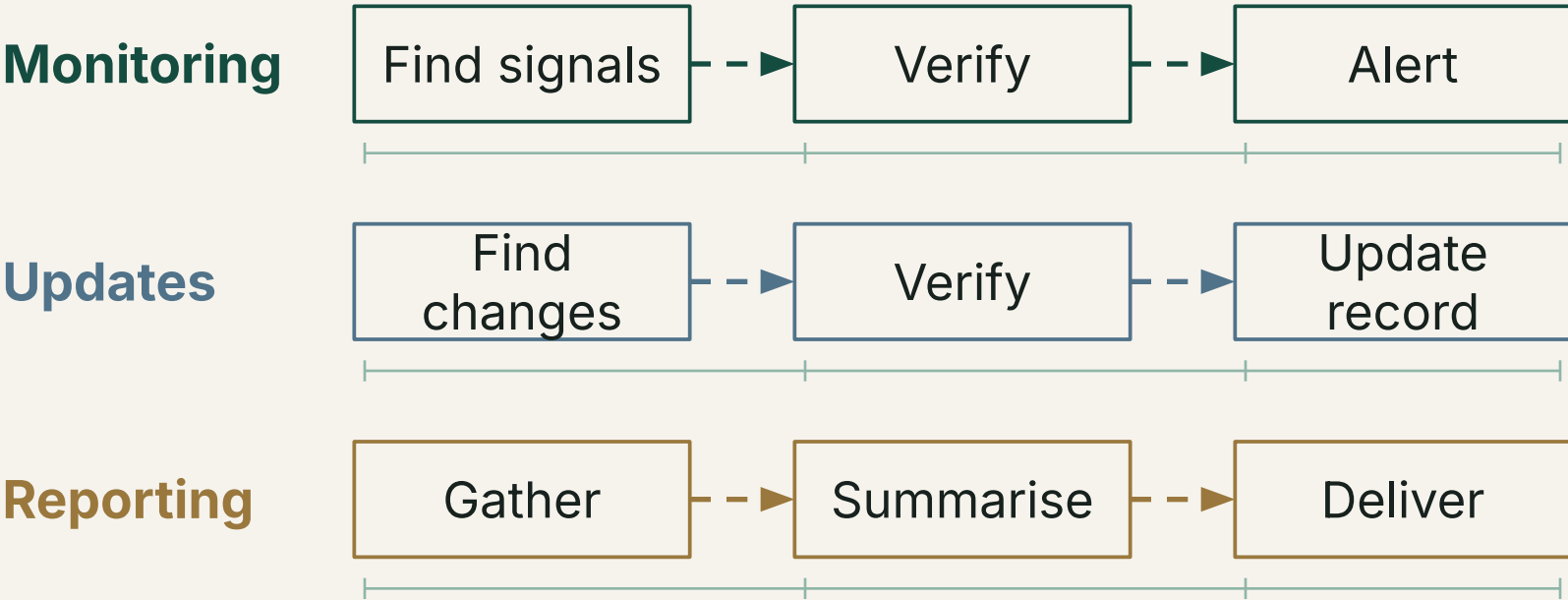
Less deliberate designs buckle under their own weight, needing constant refactoring.

A business built as functions

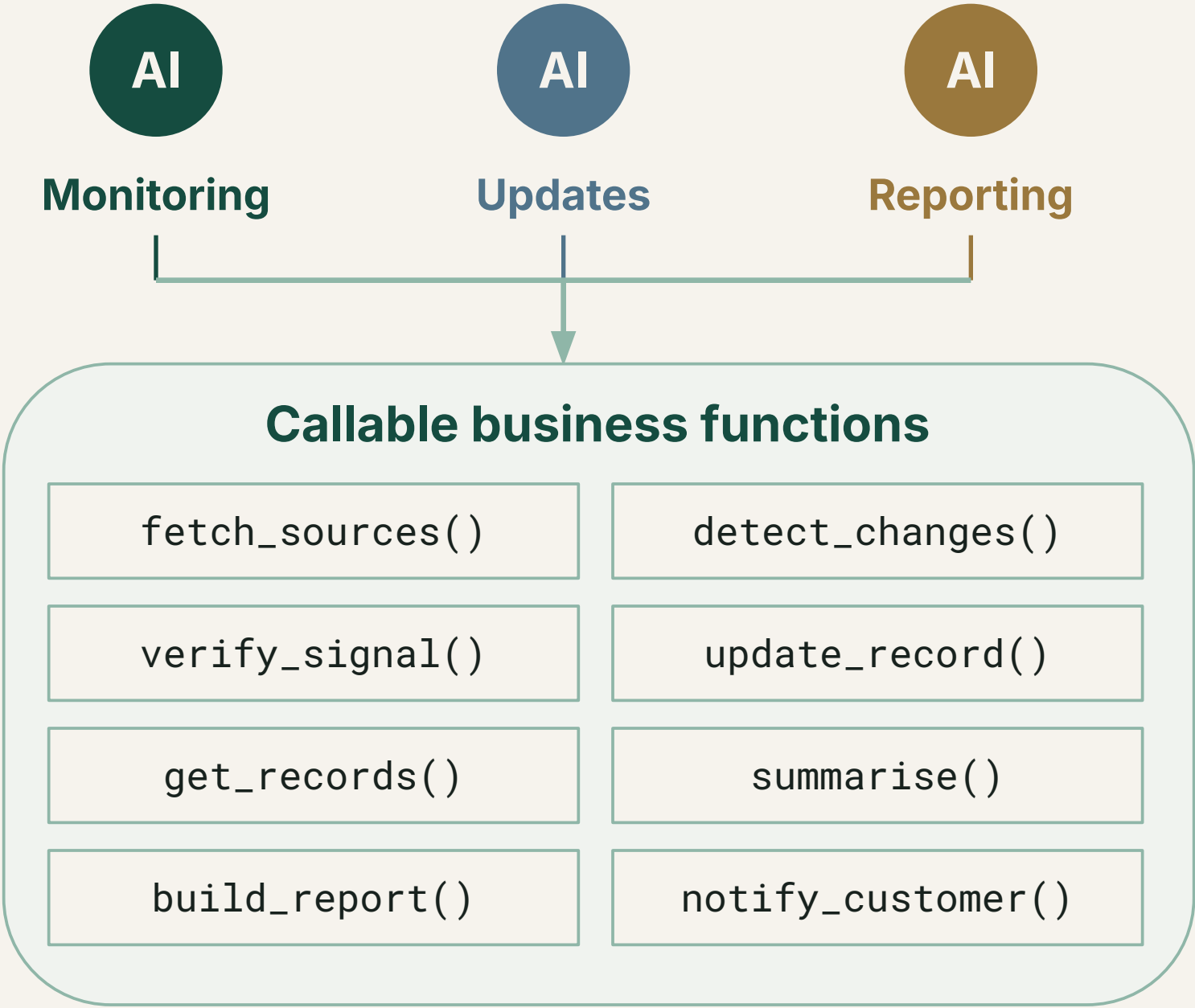


Delegation coordinates the same operations to run whole functions at greater volume.

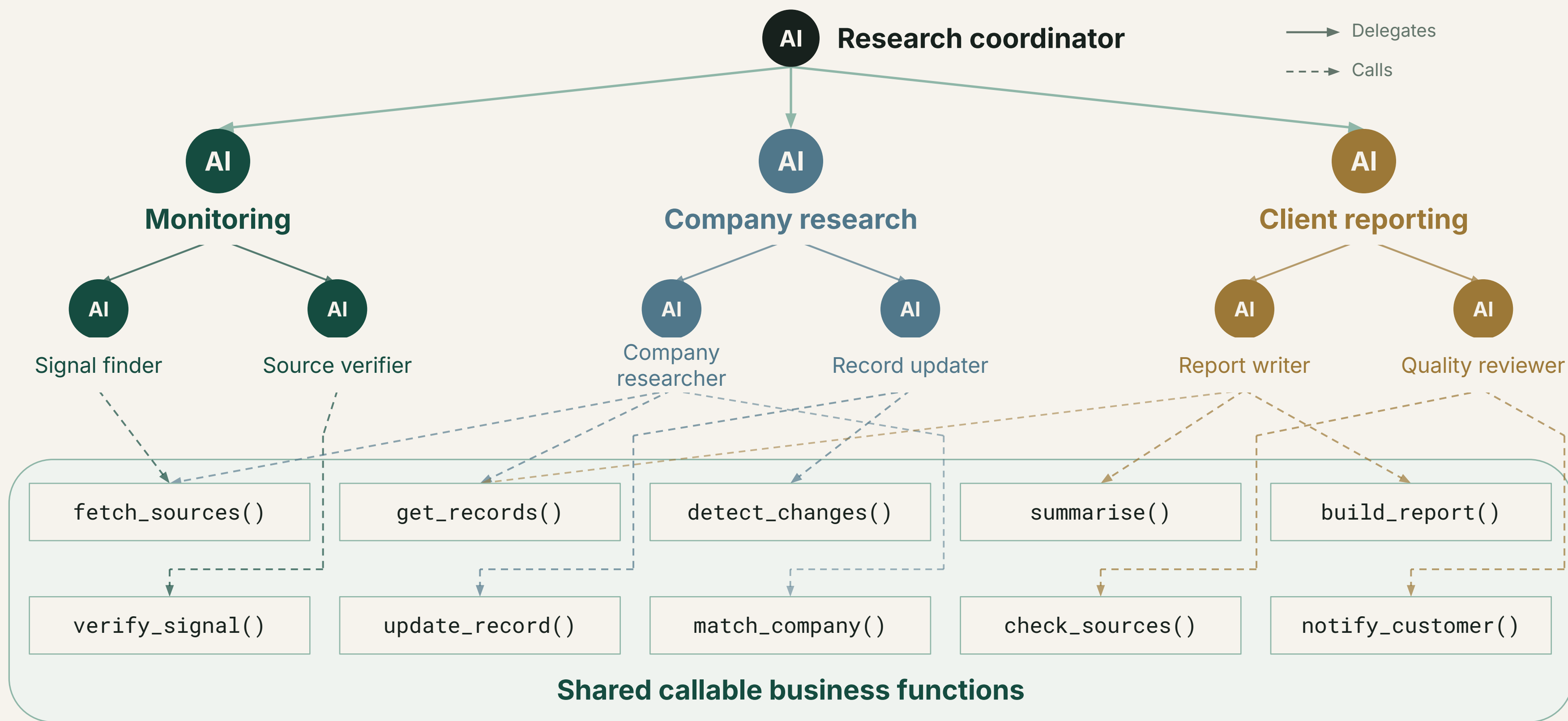
**Legacy:
Human Org chart, roles and processes**



**New World:
Agents + business functions**



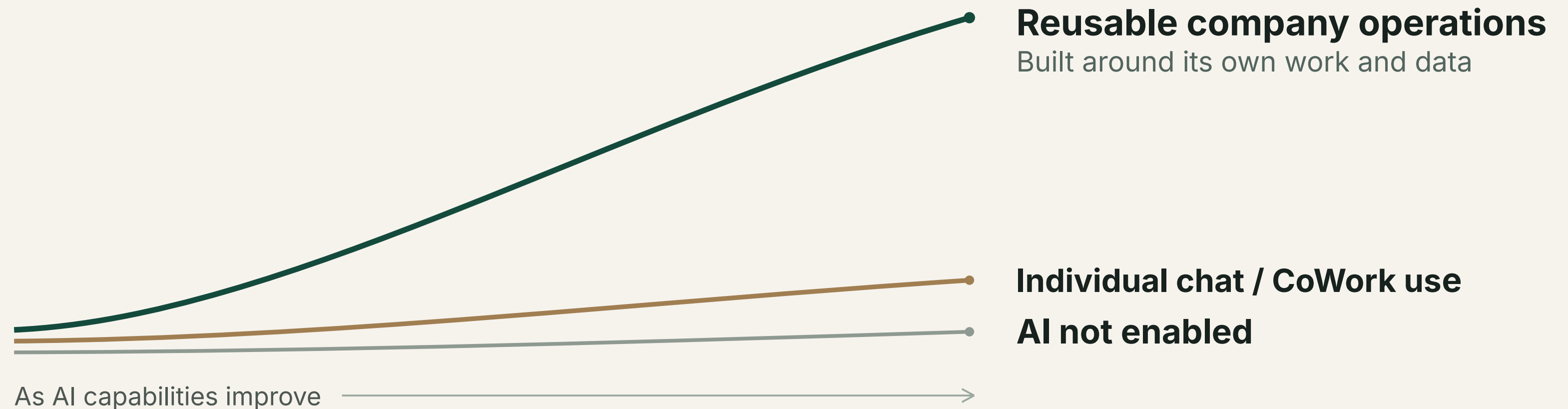
Competing with an AI-native company



Gulf between companies is growing

Companies that build reliable, reusable operations **carry them into each capability jump**. New agents can run the same operations at greater volume. Reinvesting in the people who build them makes the advantage harder to catch up with.

They are building a different cost base, infrastructure around their own work and data, and the internal ability to keep improving it.

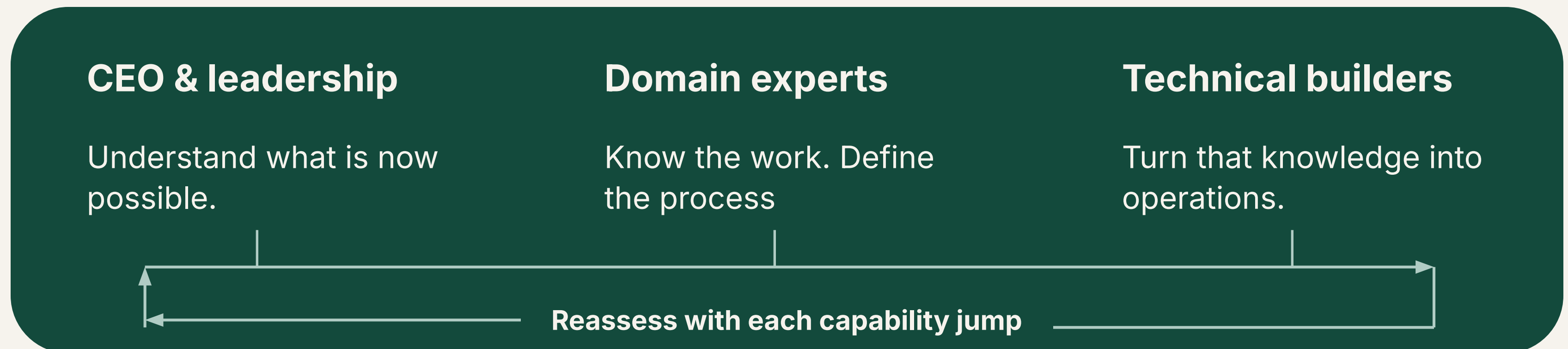


Leadership matters more than ever.

If nobody in the team can build the next version of the business, every capability jump leaves you further behind.

The people best equipped to run today's company may not be equipped for what it needs to become. Leaders need to prepare to view their business as a set of functions.

Buying established software used to be the obvious choice. For some companies we know, it is now cheaper to build and maintain the functionality they need themselves.



Commercial value is shifting (again)

Businesses sell more than individual outputs. They sell the ability to organise and deliver the work behind them. Delegation makes more of that operation reproducible too.

Fewer external buyers

Some work moves inside.

Less pricing power

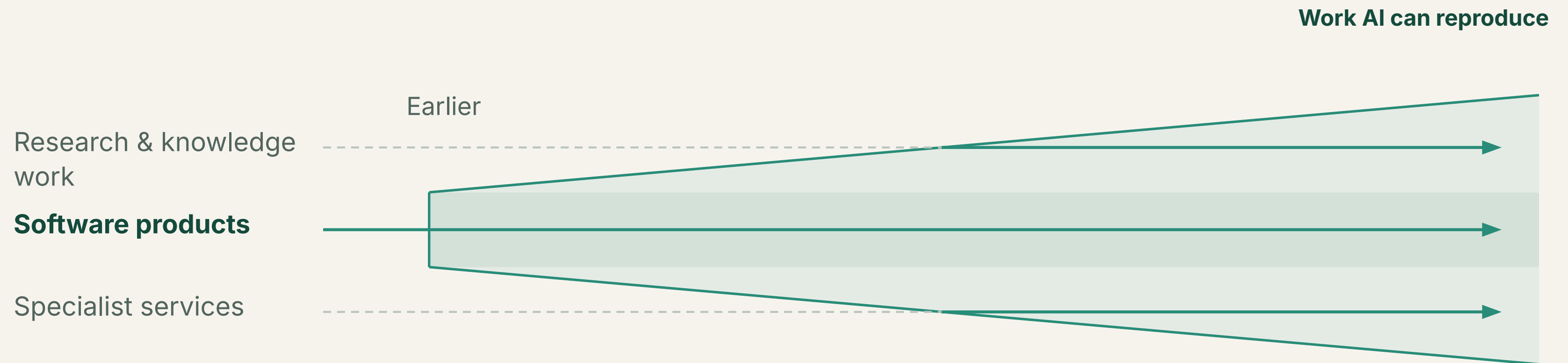
Lower fees or more work for the same fee.

Customers distributed

Customers spread in more tools and places

More competition

Smaller teams can deliver.



Threading the needle is getting harder

More of the business is exposed, with less time to respond.

Costs

Employee productivity improves, but a native competitor runs the operation at a different cost and scale.

Product

Customers can reproduce more of what they used to buy.

Distribution

You invest in one platform while customers move and agents choose other tools

Investment

The next capability increase makes your IP, infrastructure or even the agent processes you just built redundant.

Making this challenge even harder, Is a very easy environment to work yourself into a blockbuster position where you have made positive moves but are left with no response to the internet and streaming.

“Not yet” is a bet on time

Not every company will rebuild Outlook or bring bookkeeping in-house today.

Accepting that capabilities keep improving makes “not yet” a bet only on timelines. That is risky in a period of such rapid capability expansion.

The bets timeline may compress but certainly will not expand.

What we expect next

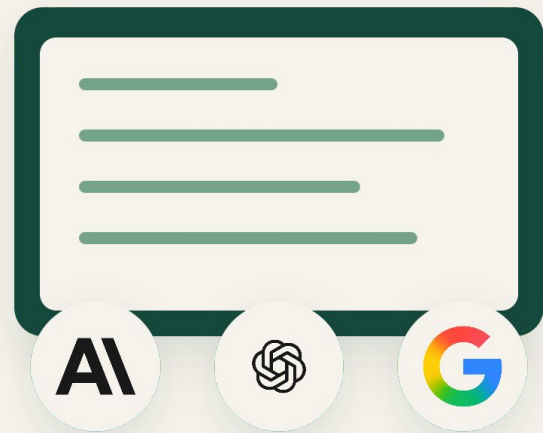
Delegation is here. We expect its consequences to keep escalating.

- 1 The work AI absorbs is moving upstream.** More of the coordination and delivery a business sells becomes automatable, alongside its individual outputs.
- 2 Competition over delegation will be intense.** Consumer products will compete to take on this work. We expect providers to favour their own services and shape who reaches customers.
- 3 Legacy chat layers are largely redundant.** Model access, tool connections and basic coordination are increasingly built into the major products. Generic multi-model portals look transitional to us.
- 4 Customers will use an increasingly wider mix of products.** Specialised tools still have reasons to exist. We expect more points of use and distribution even as the underlying models become commodities.

qbash and Tasks

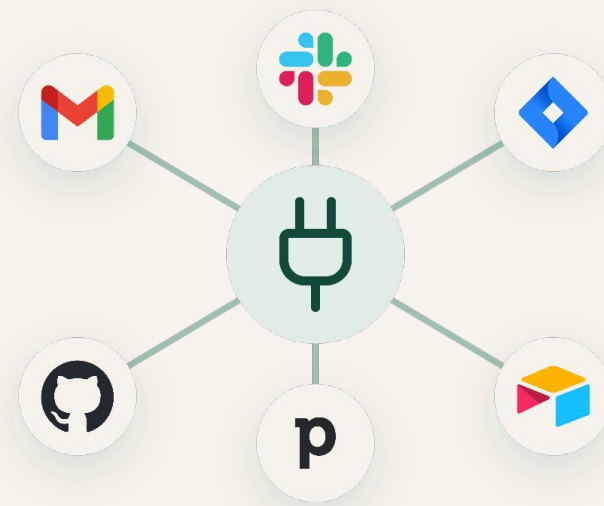
The qbash toolkit

Build an operation from Tasks, connectors and memories. MCP servers make it available to people and agents in the tools they use.



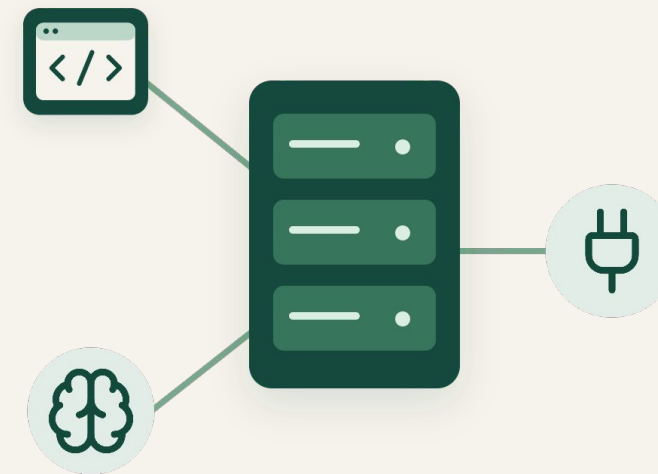
Tasks

The application logic.
Python for known steps,
AI for judgment.



Connectors

Authenticated access to
the systems your
applications read and
update.



MCP servers

Publish selected Tasks,
connector tools and
memories for people
and agents.



Memories

Keep documents and
findings that your
applications can search
and reuse.

Tasks: AI-enabled applications

Tasks are Python applications that qbash runs in the cloud.

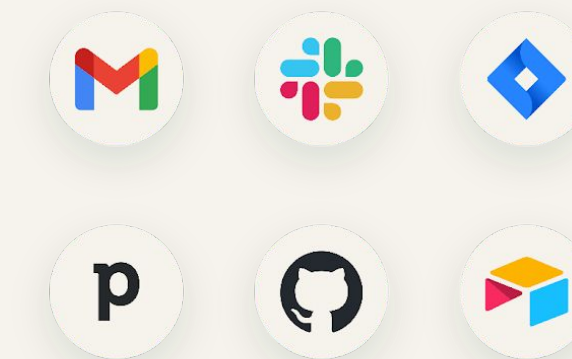
Build a repeatable process in code, using AI where judgment is needed. A Task can also wrap a service as a controlled tool for people or agents.

The SDK provides model calls, agentic runtimes, memory and connectors, with versions and logs.

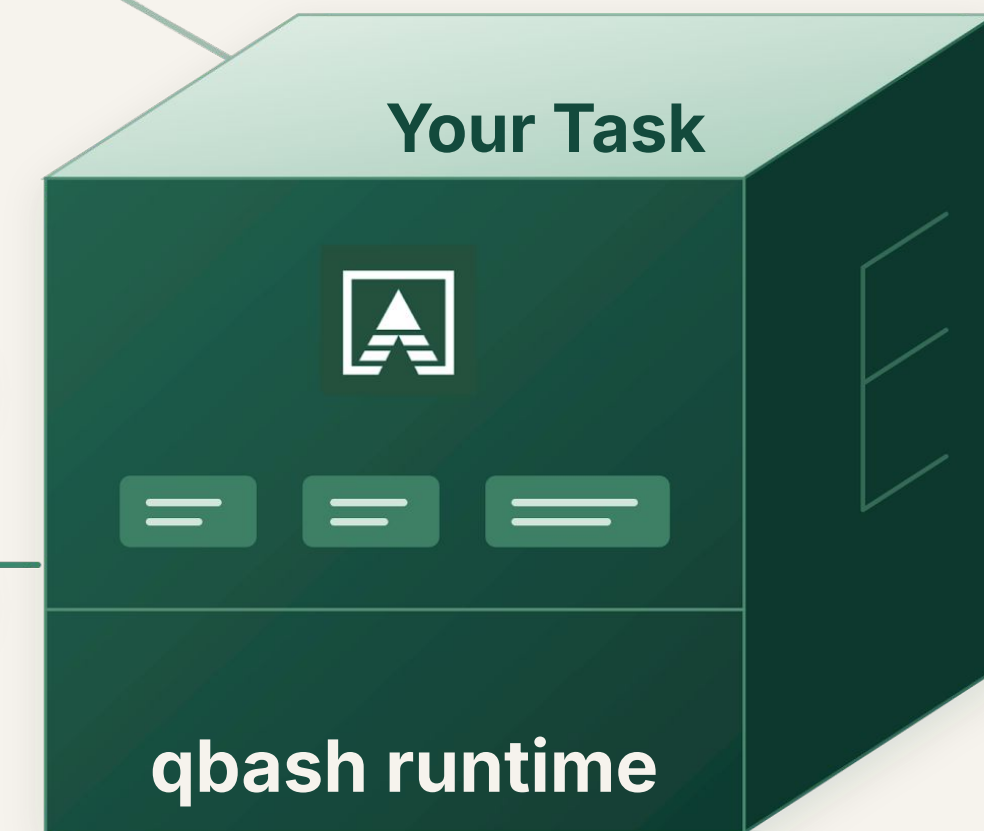
Prompts & models



Cloud hosted



Connectors



Shared memory

The same job, written two ways

```
jira = qbash.integrations.jira
for key in open_unassigned_issue_keys():
    issue = jira.get_ticket(ticket_id=key) ["ticket"]
    triage = qbash.ai.run_prompt(
        slug="triage-issue",
        variables={"issue": issue},
    )

    if triage["duplicate_of"]:
        link_duplicate(key, triage["duplicate_of"])
    elif triage["needs_review"]:
        send_to_backlog(key, triage["reason"])
    else:
        assign_team_and_priority(key, triage)

    if triage["is_outage"]:
        mark_urgent_and_notify(key)
    record_result(key, triage)
```

■ the same job, written for an agent

Go through our open Jira issues that have no team assigned. For each one, read it, decide which of our twelve teams should own it and how urgent it is, then update the issue. Keep going until everything is triaged.

Oh, and if an issue looks like a duplicate, link it to the original instead of assigning a team. If you cannot tell which team owns it, drop it in the triage backlog with a short note on why. Skip anything already assigned, do not touch closed issues, and if something looks like an outage, mark it urgent and post a heads-up in the incidents channel.

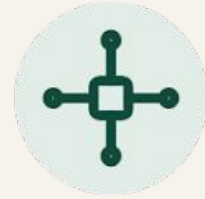
AI judges what the issue means. Once the operating rules are known, code can apply them without reasoning through the procedure again.

Benefits of Tasks



Deterministic where it matters

Program known steps.
Use AI for the judgments.



Distributable

One Task, available
through chat, APIs,
schedules and agents.



Your IP, abstracted

Keep the process in code
as models change.



Multi-model and upgradeable

Keep prompts versioned in
the built-in library. Change
models without rewriting the
process.



More predictable spend

Call models where
needed; cache and reuse
results.



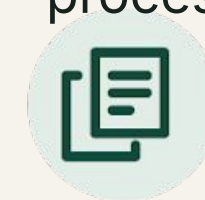
Direct API calls

Your code decides which
operation runs and when.



Privacy & access

Expose selected actions
and data without sharing
credentials.



Versions, tests & logs

Review changes, test
behavior and trace every
run.

Tasks work across chat, systems and agents

Build and maintain the application once. Make it available in the tools your team uses.



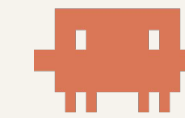
Chat tools

Use a Task from the chat tool you prefer.

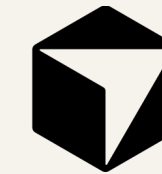


Triggers & schedules

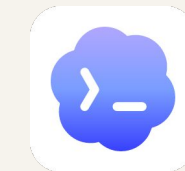
Run on a schedule or when another system triggers it.



Claude Code



Cursor



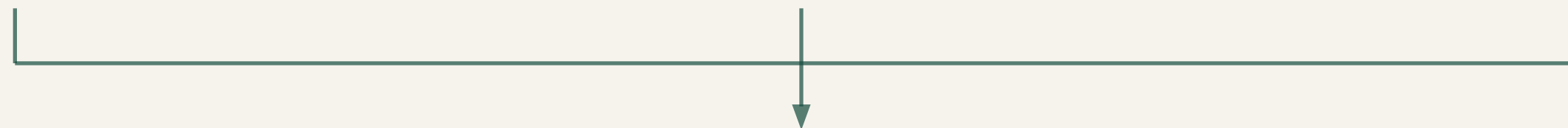
Codex



Grok Bot

Agents & agent teams

Give agents reliable tools they can call and combine.

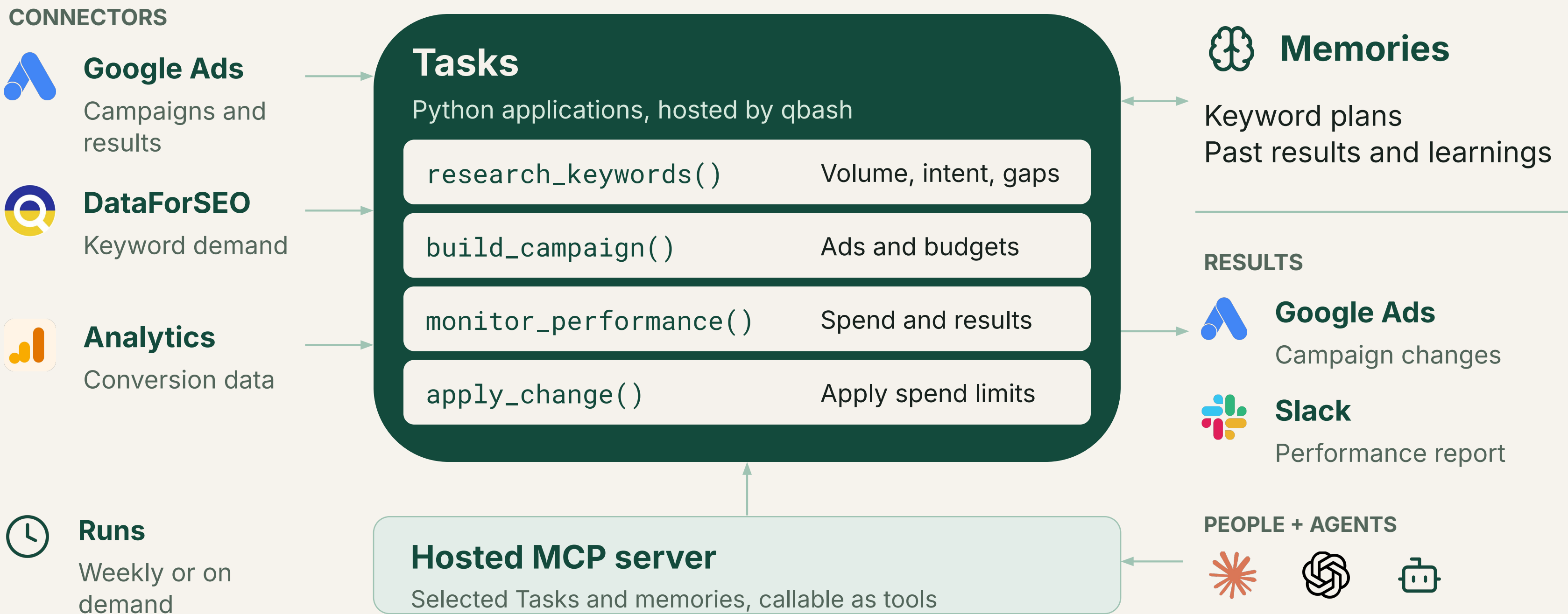


Your Task

Python application

Example: An ad manager built from Tasks

Running today on qbash's own ad account. Keyword research, campaign builds and ongoing optimisation, with rules in code and AI used for judgment.



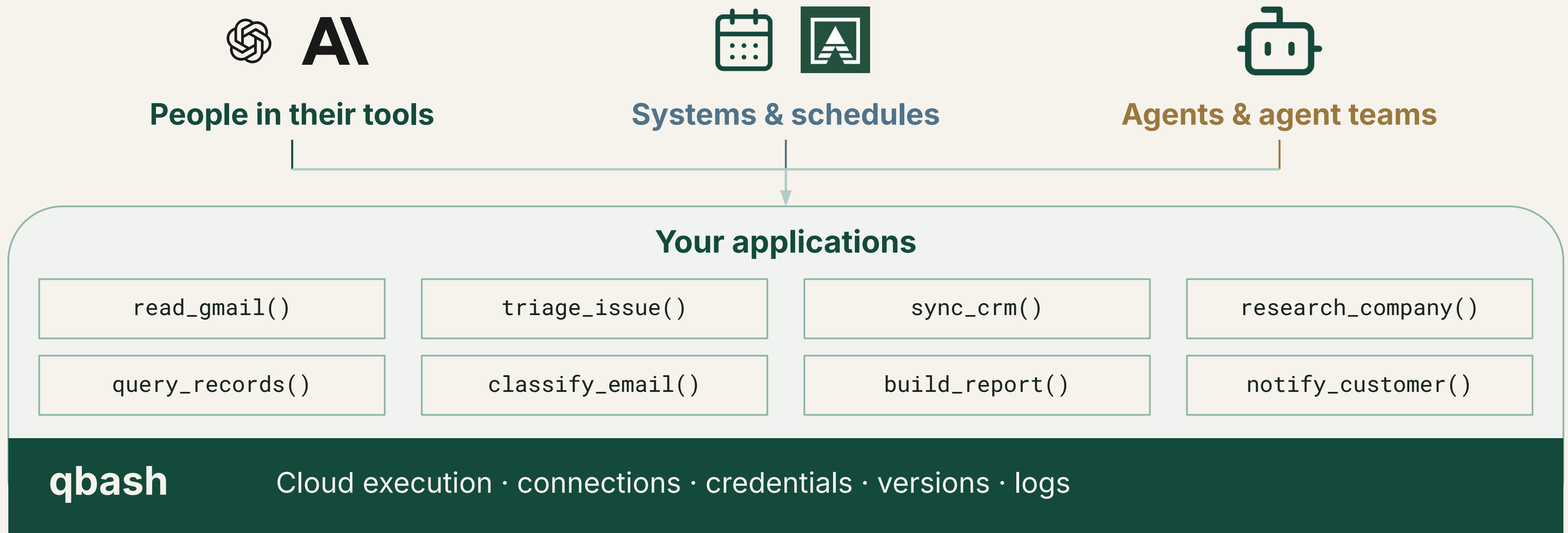
Tasks vs Alternatives

Use an agent when the next step is uncertain. Use a Task when the rules are known and it should run repeatedly.

	Agent or skill	Workflow canvas	Host it yourself	qbash Task
The process	Instructions guide decisions at runtime. It can call code and tools.	A defined graph of steps and branches.	Code and AI calls that you own.	Code, with AI calls where the work needs judgment.
Changing it	Edit instructions and evaluate the resulting behaviour.	Edit nodes, branches and configuration.	Review and test code, then deploy it.	Review versioned code, test it and publish the change.
Running it	Depends on the client or agent runtime.	The workflow service handles execution.	You own hosting, credentials, deployment and monitoring.	qbash handles task execution, connectors, revisions and run logs.
Cost and reasoning	Depends on the path, model use and runtime limits.	Depends on the service, steps and volume.	Infrastructure and model costs, plus engineering time.	Code handles known steps. Model use stays where the task calls for it.
A good fit	Open-ended work with an uncertain next step.	A process that fits the canvas and connectors.	Requirements that justify owning the infrastructure.	Recurring applications with managed infrastructure.

Tasks power people, systems and agents

More applications mean more to deploy, connect and maintain. qbash manages that shared layer, independently of the chats, systems and agents using it.



Program known steps once. Use models and agents where judgment is needed.

Tasks retain value as the tools change

After 18 months building AI automations with customers, we've seen a certain type of application retain its value: repeatable operations that keep working as customers change models, chat providers and tools.

They remove manual work today. As agents improve, the same operations become tools those agents can reuse.



Keep systems in sync

Apply agreed rules whenever a record changes.



Process incoming information

Identify, file and route documents or requests.



Research companies

Gather the same evidence and return it in a known form.



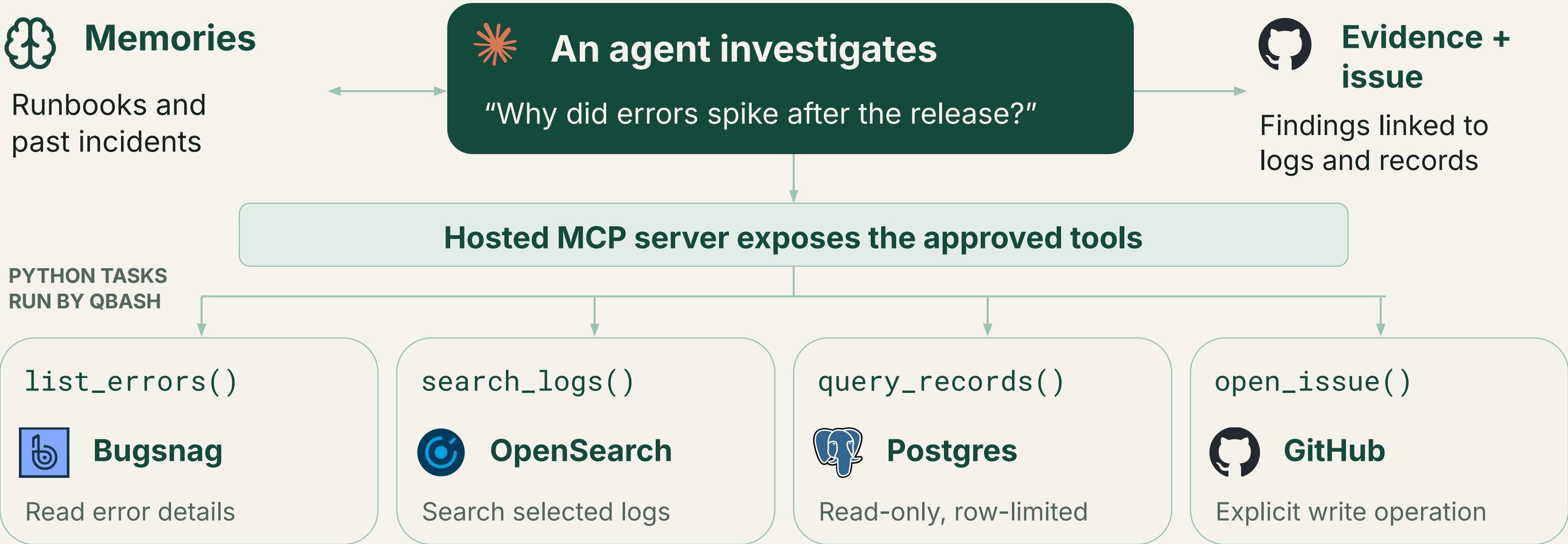
Produce regular outputs

Generate reports, updates or records on demand.

More Task Examples

Bug investigation through scoped tools

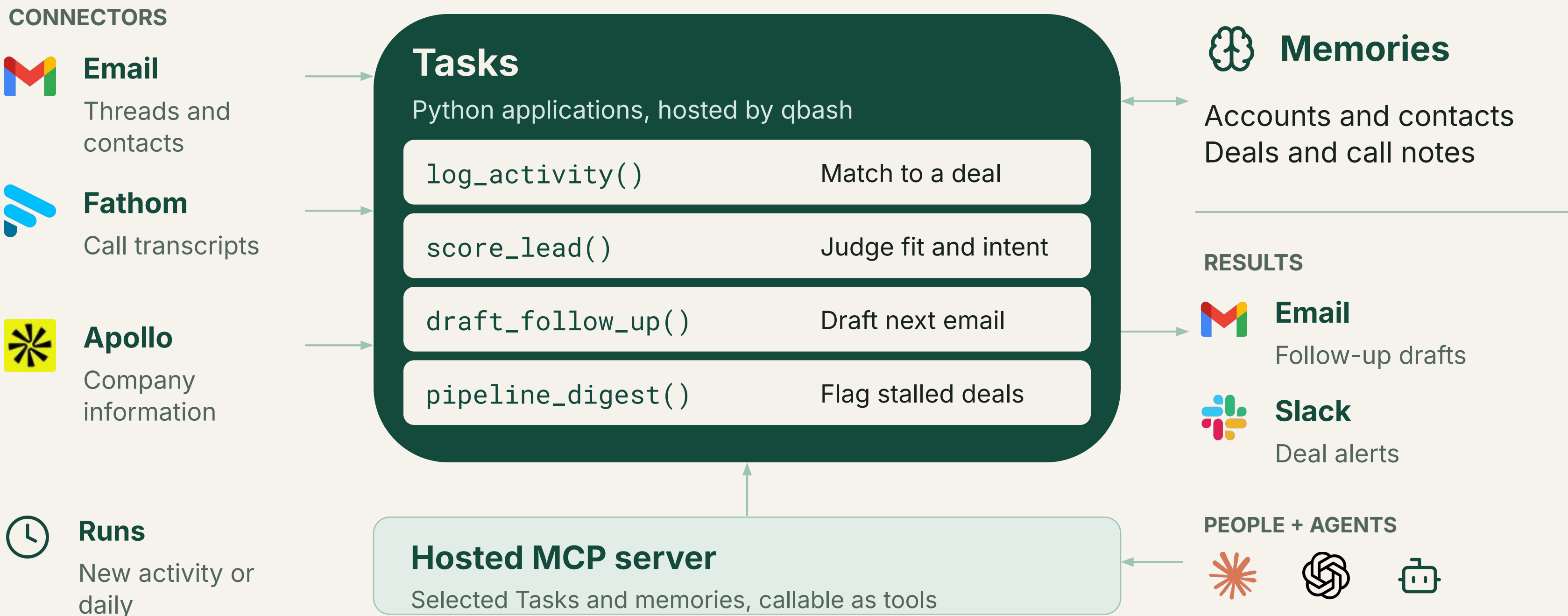
Our engineers debug using this Task



Credentials stay in qbash. Each tool exposes only the operations and data you choose.

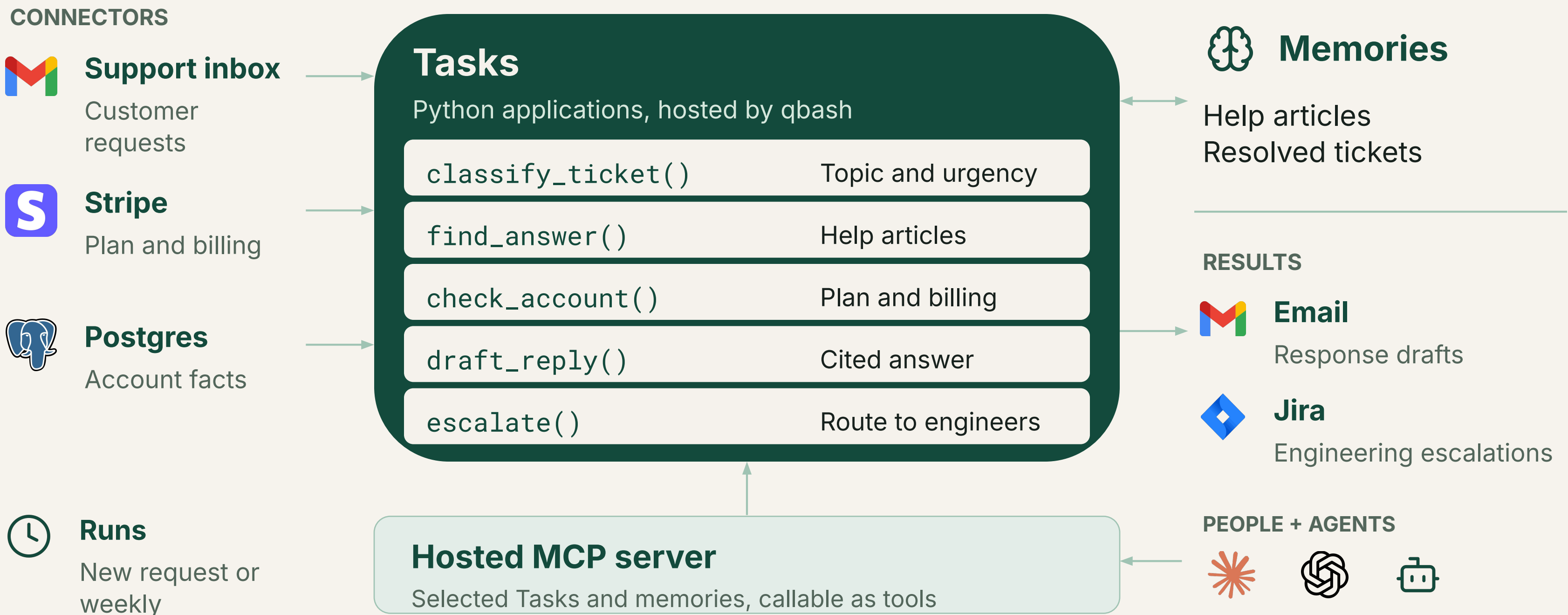
A sales operation built from Tasks

Capture activity, rank leads and draft follow-ups using the email and call data your team already has.



A support workflow built from Tasks

Incoming requests become grounded drafts or engineering escalations, with account facts and documentation attached.



Recurring reporting built from Tasks

Keep metric definitions in one place, run reports on a schedule and investigate changes against the underlying data.

