Imagine you're on-call

Generated by Gemini

LinkedIn

CAPT

Contextual Agent Playbooks and Tools

Context Engineering at LinkedIn

Ajay Prakash

Senior Staff Engineer · LinkedIn

LinkedIn

Coding agents

Pre-2024

Smart autocomplete

Finish-the-line. Finish-the-function.

Late 2024

"Agent mode" arrives

Multi-file edits. Run commands. Make changes autonomously.

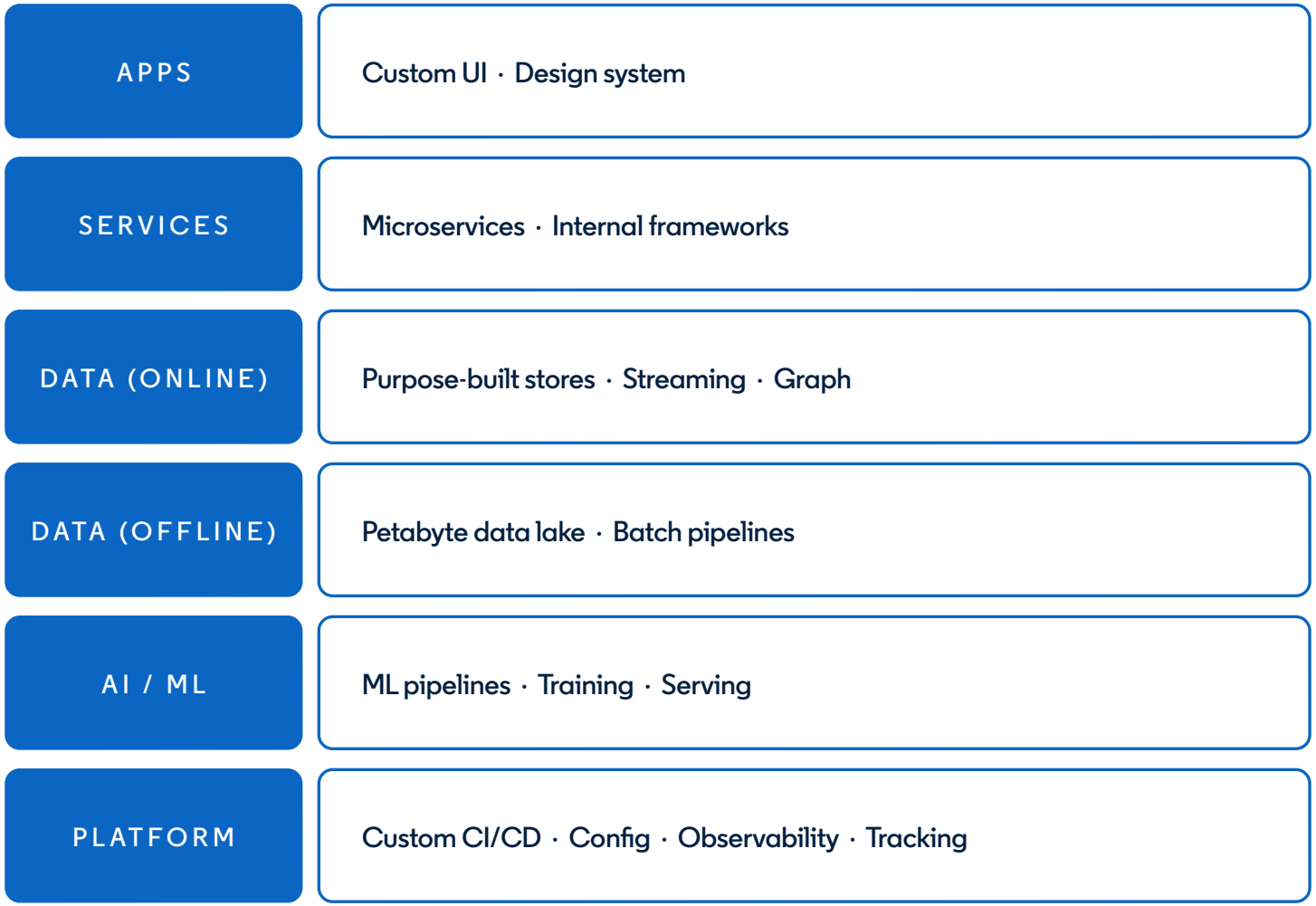
Early 2025

"Vibe coding"

Karpathy coined the term. Top-down push to adopt at every company.

Coding agents didn't work on mature codebases

LinkedIn's stack: what the model doesn't know



1,000s of repos.

Internal frameworks at every layer.

Custom-built infra for our scale.

LinkedIn

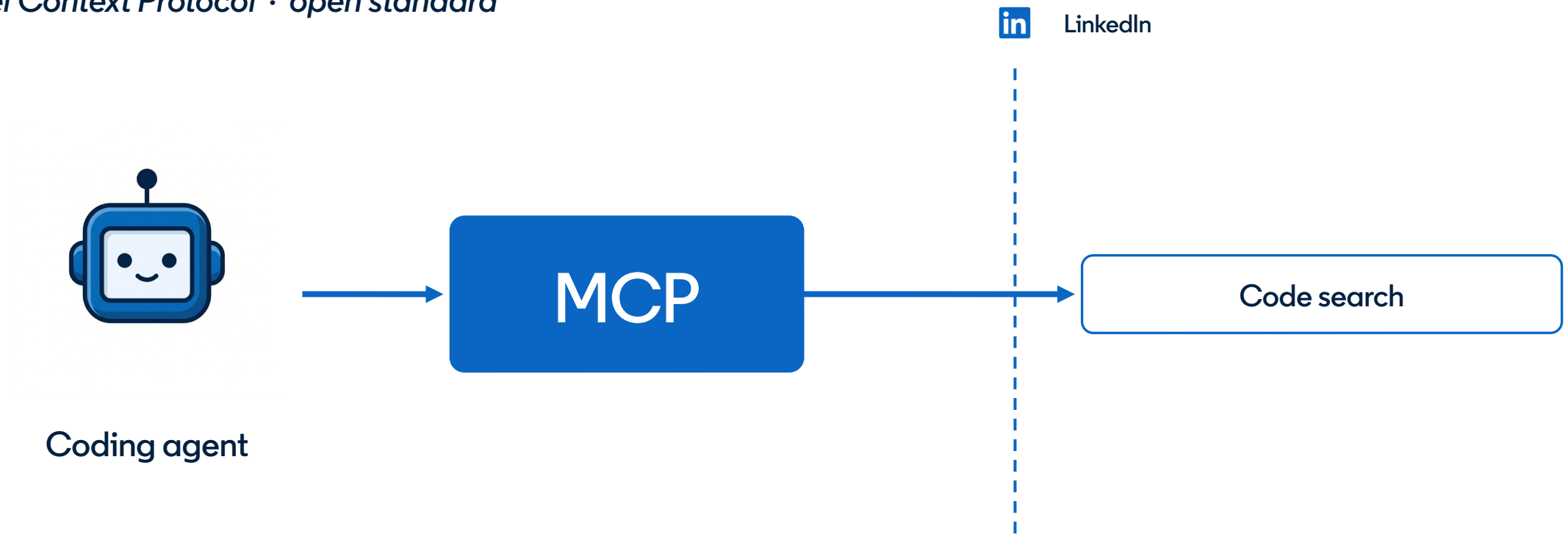
How do we make any coding agent
understand LinkedIn well enough

to ship code our engineers can trust?

LinkedIn

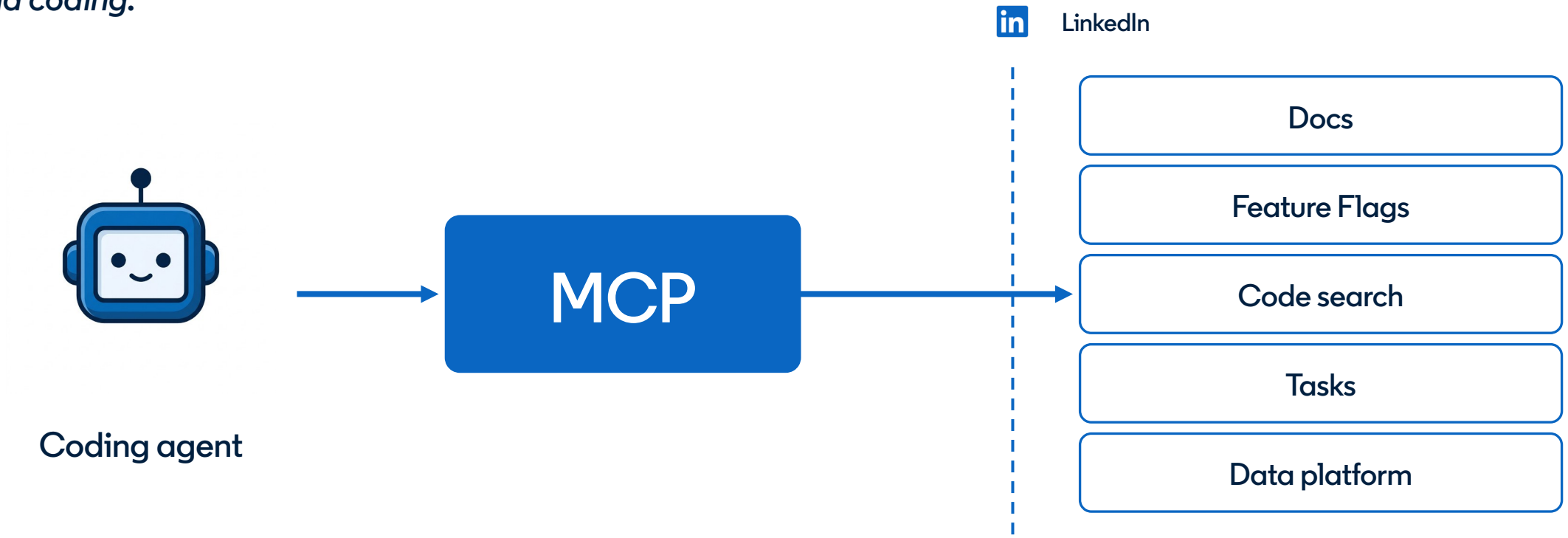
MCP: one protocol, any agent

Model Context Protocol · open standard



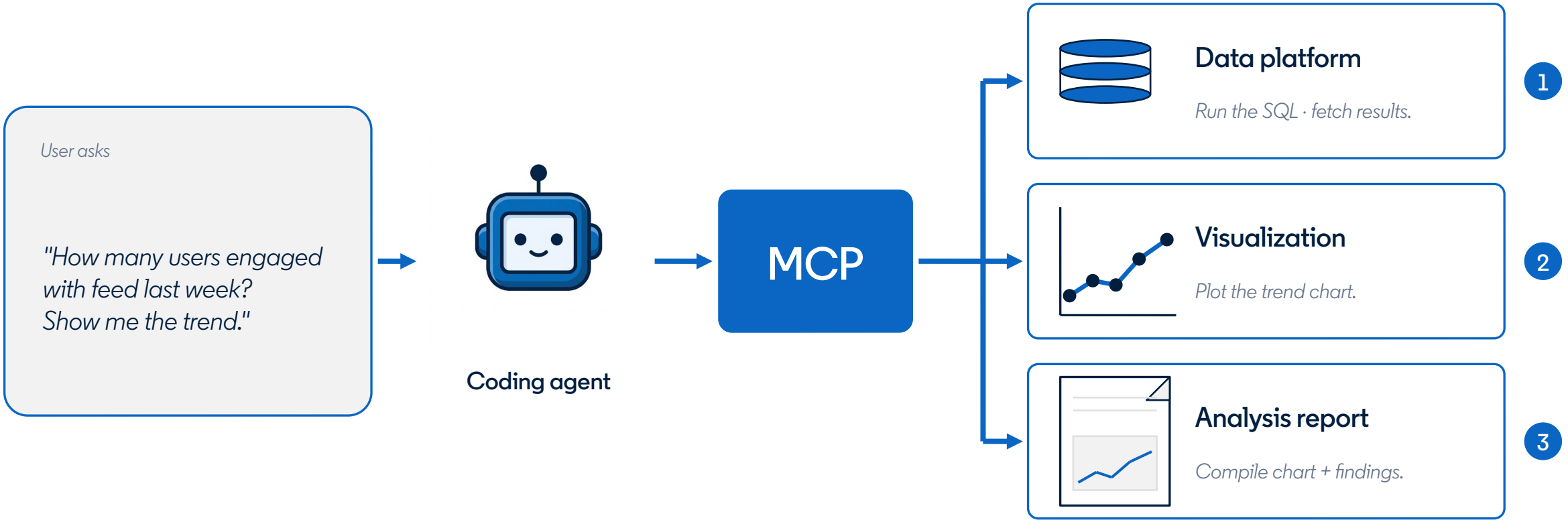
Every tool compounded the last

Beyond coding.



Now anyone can ask the data

PMs, designers, EMs: anyone can self-serve from a question.



The context gaps

Tools alone aren't enough. Long-running tasks need the right context.

Tribal knowledge

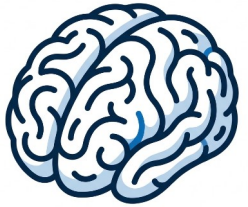
Scattered across docs, wikis, Slack.

Agents hit **dead ends or hallucinate**.

Context overload

More tools, more searching.

Window fills, agent **loses what it found**.



No durable memory

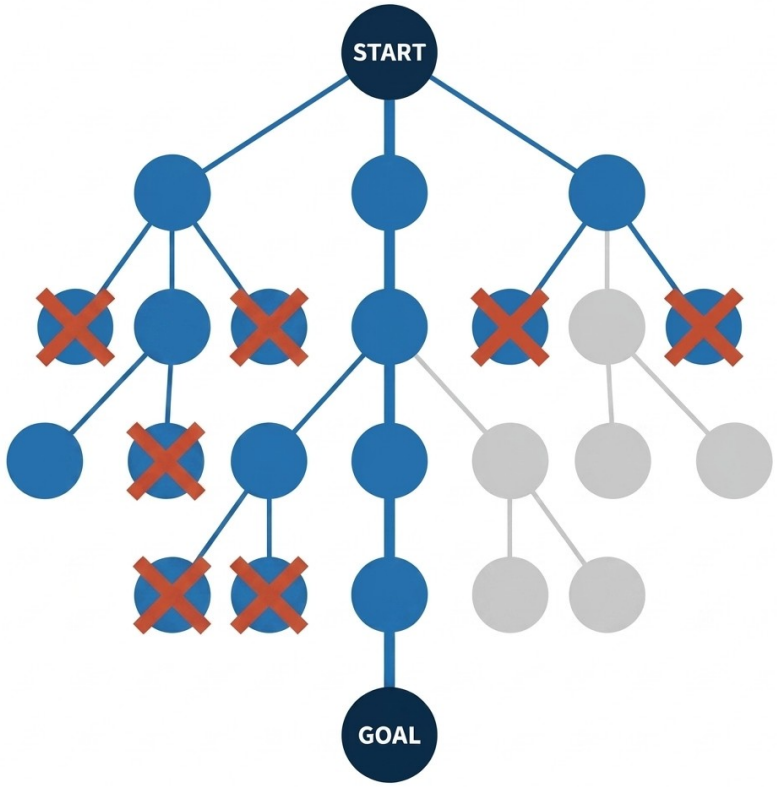
Every session starts from zero.

The path is **rediscovered every time**.

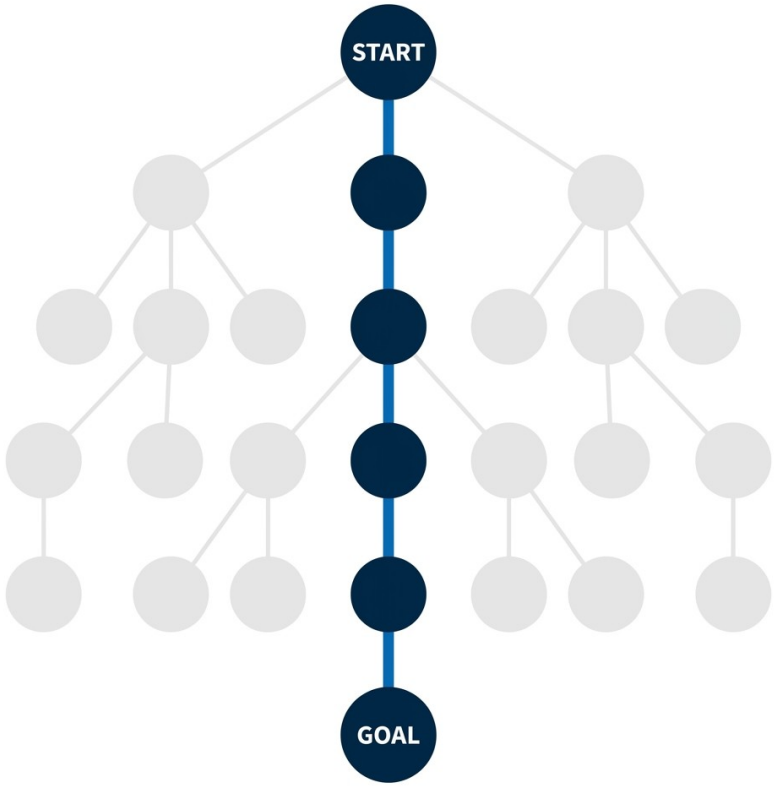
LinkedIn

Procedural memory

Provide the relevant context upfront.



Without procedural memory.

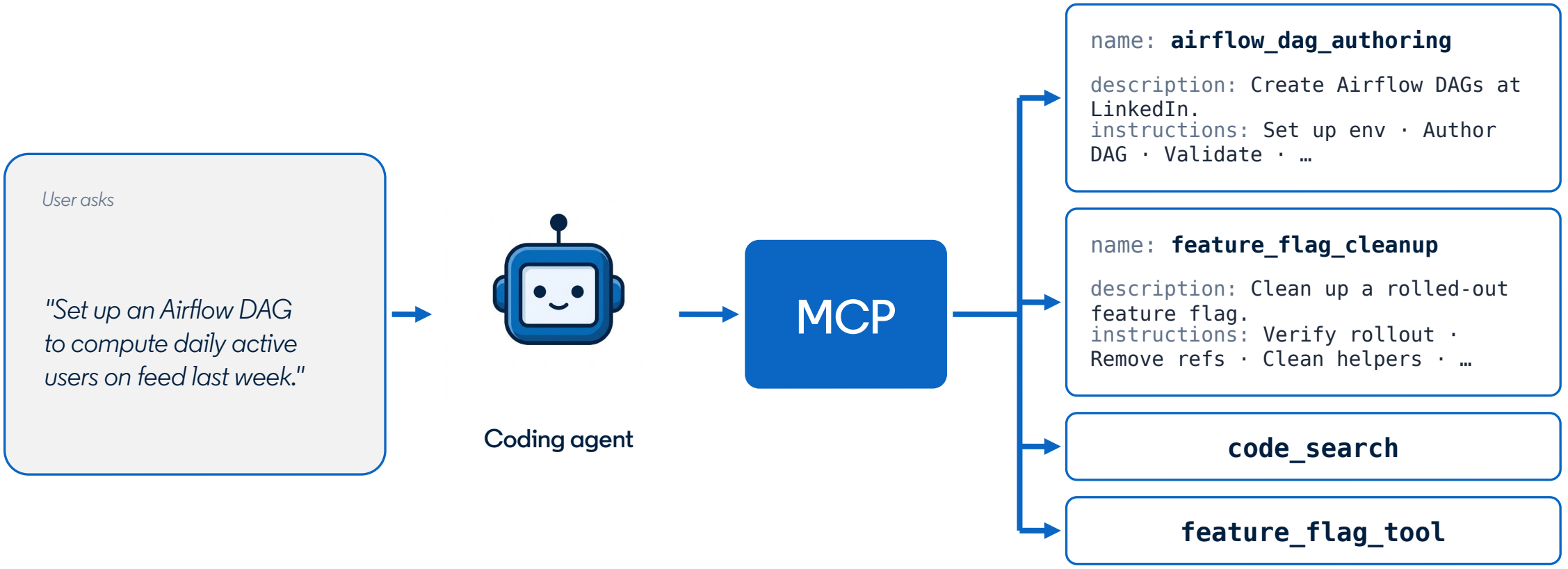


With procedural memory.

LinkedIn

Playbooks

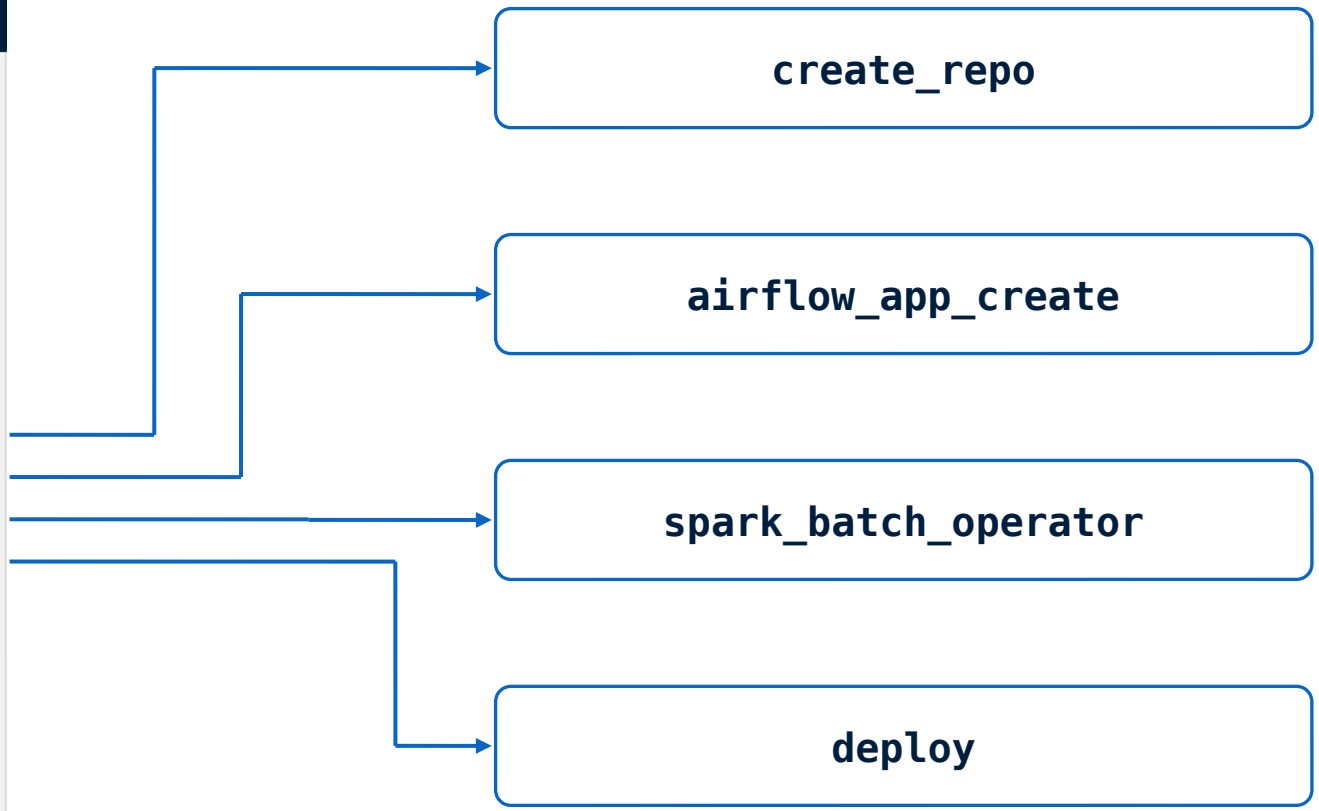
Procedural memory, served via MCP.



Anatomy of a playbook

airflow_dag_authoring.yaml

```yaml
name: airflow_dag_authoring
description: Create Airflow DAGs at LinkedIn.
inputs:
  dag_name: "user_features_daily"
  schedule: "0 13 * * *"
instructions: |
  ## Concepts
  ## Steps
    1. Set up env        → create_repo
    2. Author DAG        → airflow_app_create
    3. Add Spark task    → spark_batch_operator
    4. Deploy            → deploy
  ## Best practices
  ## Gotchas
    - timezone-naive datetimes
    - retries default to 0
  ## Verification
  ## Troubleshooting
```

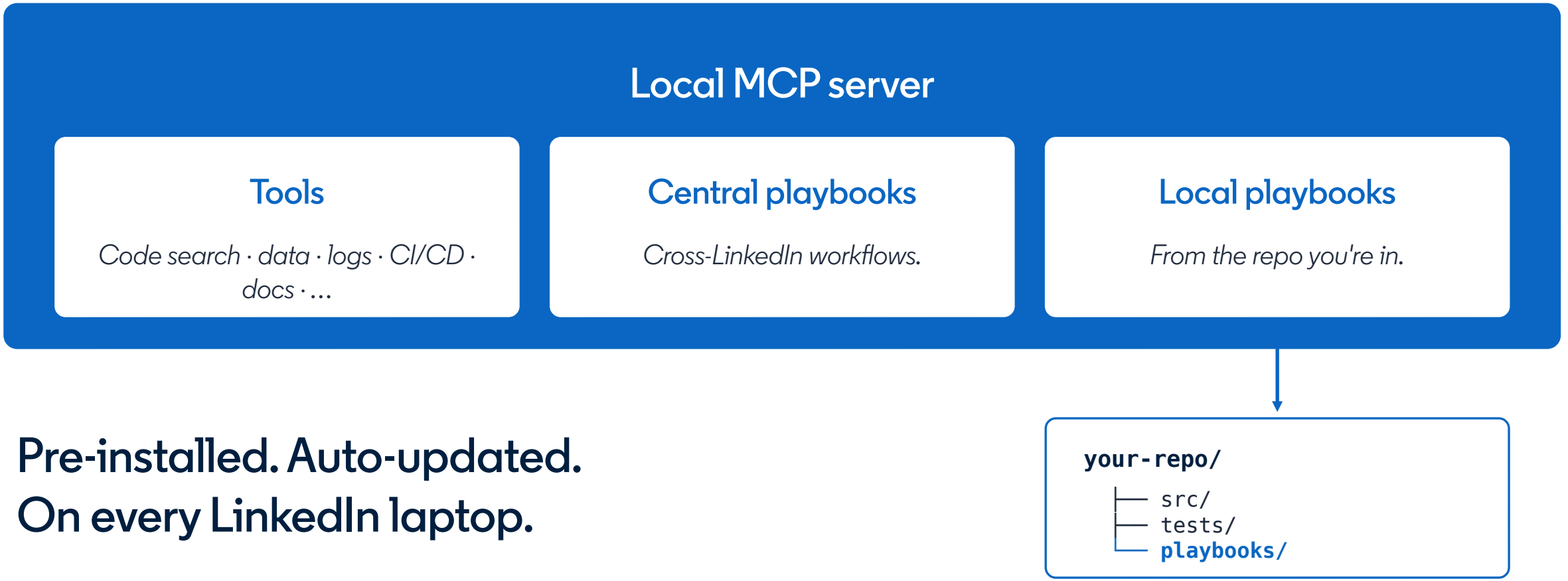


Self-contained. *Does one specific task.*

Composable. *Chains into reusable subtasks. Progressive discovery.*

LinkedIn

Architecture



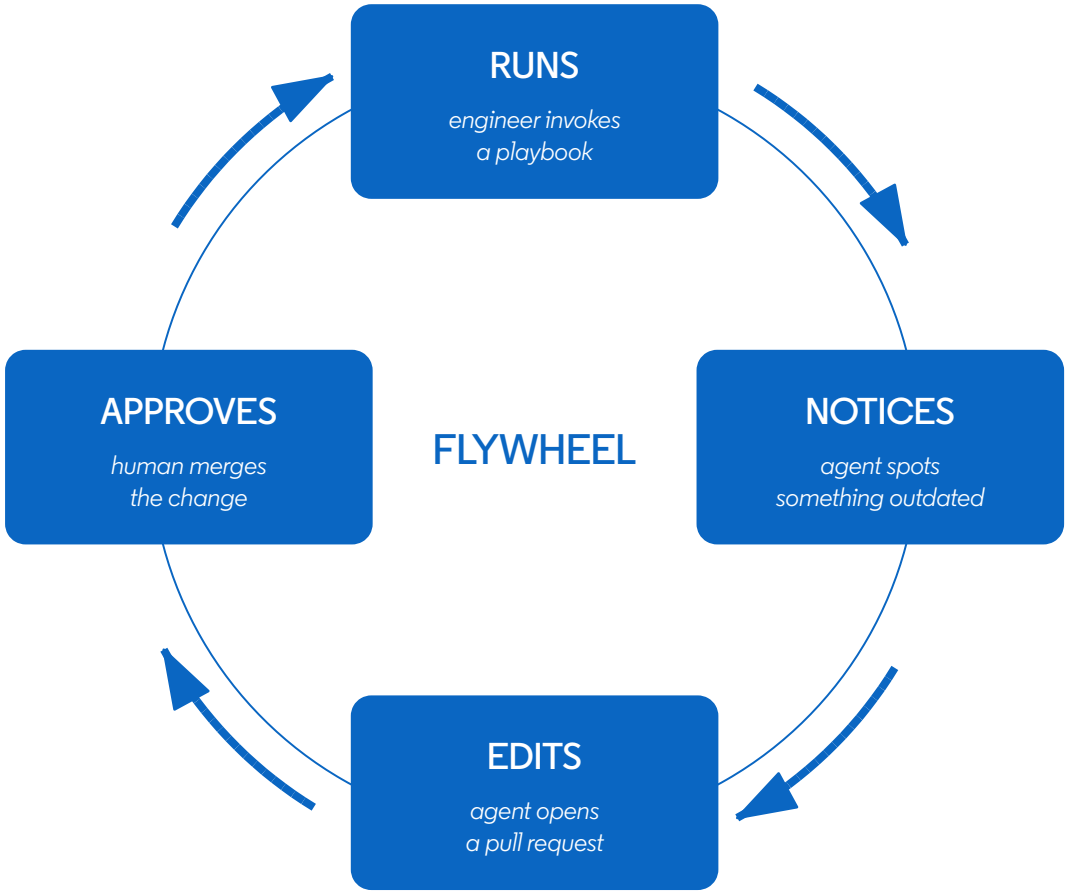
Pre-installed. Auto-updated.
On every LinkedIn laptop.

Scaling the tool catalog

Past ~30 tools, the agent slows down.

Meta-tools

Self-improving playbooks



1. Engineer runs a playbook.
2. Agent notices something outdated.
3. Engineer asks the same agent to update it.
4. Agent edits the playbook → opens a PR.
5. A human approves.

Every new learning flows back into the playbook.

LinkedIn

Telemetry

USAGE

- Daily / weekly active users per tool and playbook
- Retention
- Adoption by team

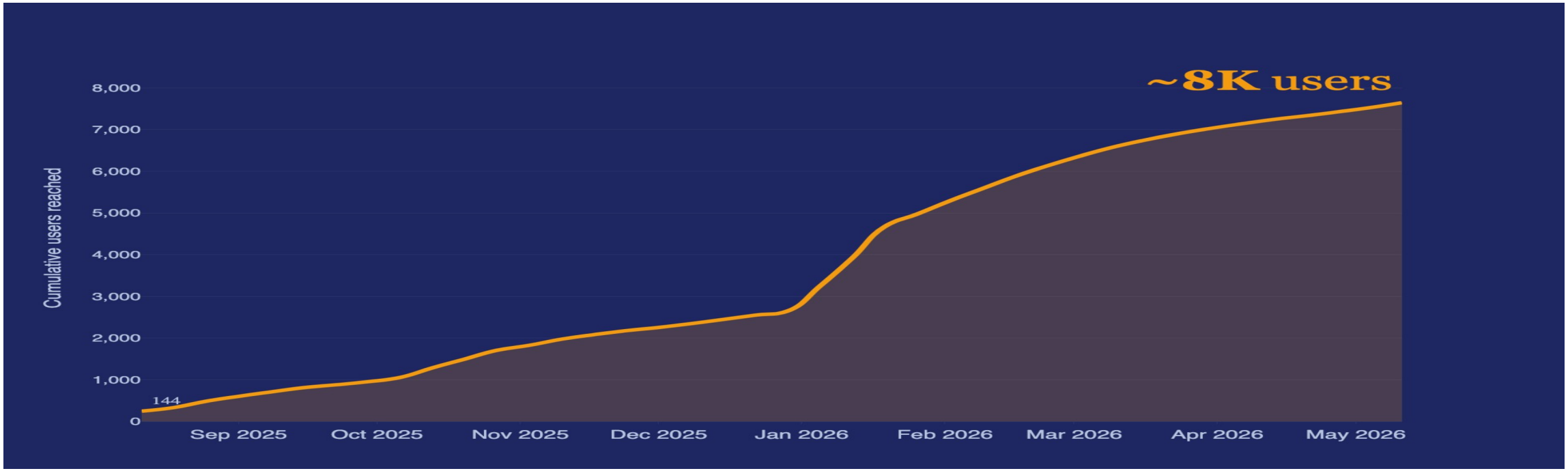
INSIGHTS

- Top playbooks
- Common tool combinations
- Cross-workflow patterns

OPERATIONS

- Failure alerts when tools break
- Server health and crash detection
- Latency per tool
- Debug logs

Beyond engineering



70%
faster triage

3×
faster data analysis

20%
productivity lift

What teams use playbooks for

1

Debugging, investigation & on-call

Logs, metrics, alerts, ticket triage, root-cause. When something breaks, the agent is the first stop.

2

Code & artifact scaffolding

Tests, repos, gRPC clients, Airflow DAGs, dbt models. Every team writes its own scaffolding for its stack.

3

Pipelines & deploys

Trigger and debug Airflow DAGs, Spark jobs, ML training. Watch rollouts. The long-running chores nobody volunteers for.

4

Code cleanup & migrations

Feature-flag cleanup (7 stack variants), framework migrations, deprecations. Forked across stacks.

5

Environment & repo setup

New repo, dev env, team onboarding. The "where do I even start?" moment.

What's next

1 Playbooks become skills

Migrating playbooks to skills, the emerging standard for packaging procedural memory so any coding agent can use them.

2 Playbooks write themselves

Generating new playbooks automatically by mining agent telemetry for repeated workflows. The patterns are already in the data.

3 Playbooks improve themselves

Offline jobs capture every learning from agent runs and update the playbooks automatically. The flywheel keeps spinning without a human in the loop.

LinkedIn

Thank you.

Ajay Prakash

Senior Staff Engineer · LinkedIn

linkedin.com/in/ajay-prakash-3780b132

Connect with me on LinkedIn

tinyurl.com/y3zapcj9

CAPT blog post. Full architecture, metrics, and lessons.

Connect

LinkedIn