Tools, agents, and agentic

The capability ladder — and why "agentic" is a rung, not a word.

Nine words.

A thing people have wanted to say to software for 20 years and couldn't.

> ***"Book me a flight to Berlin for Thursday."***

When the model can actually do this — not describe it, not suggest it, do it — something qualitatively different has happened.

It stopped being a tool that answers. It became a tool that acts.

THE WORD PROBLEM

"Agentic" tells you almost nothing.

Applied to everything from a chatbot-with-search to a self-improving autonomous research program.

"Our AI agent books flights"	*Could mean: rung 4 autonomous purchase*
"Our agentic chatbot answers questions"	*Could mean: rung 1 with a search button*

The fix: a ladder.

The capability ladder.

Five rungs. Each distinct in a way you can point at.

RUNG 05	**Autonomous system** *Infrastructure · persistent · generates own sub-goals*	HUMAN SETS POLICY
RUNG 04	**Agent** *Plans · acts · notices its own errors · longer horizon*	GOAL + BOUNDARY
RUNG 03	**Workflow** *Multi-step sequence between turns · human reviews*	INITIATES + REVIEWS
RUNG 02	**Tool-using chat** *Calls tools during a turn · still turn-based*	IN LOOP EVERY TURN
RUNG 01	**Chat** *Pure question-answer · produces text only*	DRIVES EVERYTHING

Tools are the hinge.

Rung 2 is where the closed world of the model breaks open.

A TOOL CALL

```
search_flights(origin="BER", destination="MUC", date="2026-04-25")
```

LIVE DATA

Break the closed world of training data. Query what's true now.

EXACT OPS

Arithmetic, lookups, anything that needs precision the model can't provide.

REAL ACTIONS

Send, schedule, transact. Everything called "agentic" is tools underneath.

MCP — the HTTP of this layer.

Model Context Protocol. The standard that makes cross-vendor tool use possible.

THE PROBLEM

How does a model built by Company A use a tool built by Company B? Historically: custom integration per pair. Quadratic cost.

THE ANSWER

An open standard. Any MCP-compliant tool works with any MCP-compliant model. Linear cost.

THE TRAJECTORY

Originated at Anthropic (2024). Now under Linux Foundation's Agentic AI Foundation. In three years: default assumption, not novelty.

Every rung up adds failure modes.

Four that matter most. Each one gets harder at higher rungs.

01	ERROR COMPOUNDING	*95% per step × 10 steps = 60% end-to-end. × 100 steps = under 1%.*
02	RUNAWAY COST	*Chat = thousands of tokens. Agent loops = millions. Budget caps aren't optional.*
03	PROMPT INJECTION	*A model that reads external content and acts has a large attack surface.*
04	RECOVERY	*At rung 1 you retry the prompt. At rung 5 you need state that survives partial failure.*

WHY AGENT LOOPS ARE HARD

95% per step.

Feels brilliant on one turn. Still feels good on two.

1 step

95%

single turn

10 steps

60%

workflow

100 steps

<1%

autonomous system

Higher rung = more per-step accuracy matters.

Four questions cut through the marketing.

Answers place any "AI agent" on the ladder in about a minute.

01

What tools does it have access to?

Read-only lookups? Or tools that take actions?

02

What's the autonomy horizon?

One turn? Ten? An hour? A week?

03

Who's in the loop and when?

Before every action? Before some? Only in audit?

04

What's the blast radius of a mistake?

A wrong sentence? A misrouted email? An irreversible transaction?

A worked example: the expense report.

Same task, three possible rungs. Which actually fits?

RUNG 02

Tool-using chat

User uploads receipt, model extracts + asks confirmation. Two turns.

RUNG 03

Workflow

Model reads inbox, finds receipts, drafts report. User reviews before submit.

← FITS

RUNG 04

Agent

Model reads, drafts, submits. User sees result after. Overkill for this task.

Pick the lowest rung that delivers the capability.

Three practical rules.

For any system at rung 3 or higher.

01

Pick the lowest rung that works

Every rung up costs more, fails more, and is harder to oversee. Climb only when the lower rung genuinely can't deliver.

02

Plan verification before building

Rung 3 + human review. Rung 4 + guardrails as product. Verification posture (piece 05) isn't a retrofit.

03

Design for fallback

A good rung-3 system degrades to rung 2 on failure — asks the user instead of improvising. Graceful drop-down = production-ready.

What this changes.

Two shifts once you hold the ladder in mind.

CONVERSATIONS GET SPECIFIC

Not "should we use agents" but "for which task, at which rung, with what tools, under what oversight?" Radically different answers at rung 2 versus rung 4.

ARCHITECTURE GETS LAYERED

Real systems aren't one agent at one rung. They're rung 2 chat fronting rung 3 automations occasionally escalating to rung 4. Right rung per sub-task, not the highest rung overall.

THE CORE IDEA

"Agentic" isn't one thing. It's a rung.

Name the rung. Unlock the real decision.

Takeaways

01

Five rungs, not one "agentic".

Chat → tool-using chat → workflow → agent → autonomous system. Each distinct in a way you can point at.

02

Tools are the hinge. MCP is the standard.

Rung 2 is where the closed world breaks open. Model Context Protocol is the HTTP of this layer.

03

Pick the lowest rung that works.

Every rung up costs more, fails more, and is harder to oversee. Rung elevation must be earned.