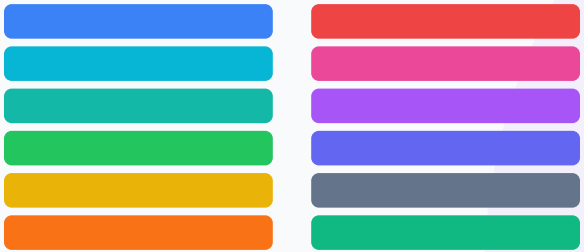
AI Agents

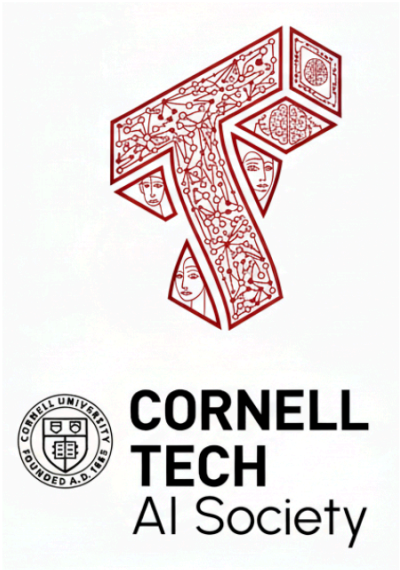
from First Principles

Building a mental model for the entire AI agent stack



Pranav Dhingra · No-Code Works × Cornell Tech AI Society · March 22, 2026


NCW


CORNELL UNIVERSITY
FOUNDED A.D. 1865
CORNELL
TECH
AI Society

The Jargon Problem

Transformers

RLHF

LLMs

MCP

GPT

Agents

RAG

Context Window

Fine-tuning

Tokens

Orchestration

CoT

Function Calling

Evals

A2A

Once you understand the layers, all of this makes sense.

The Mental Model

12 layers. Each builds on the one below.

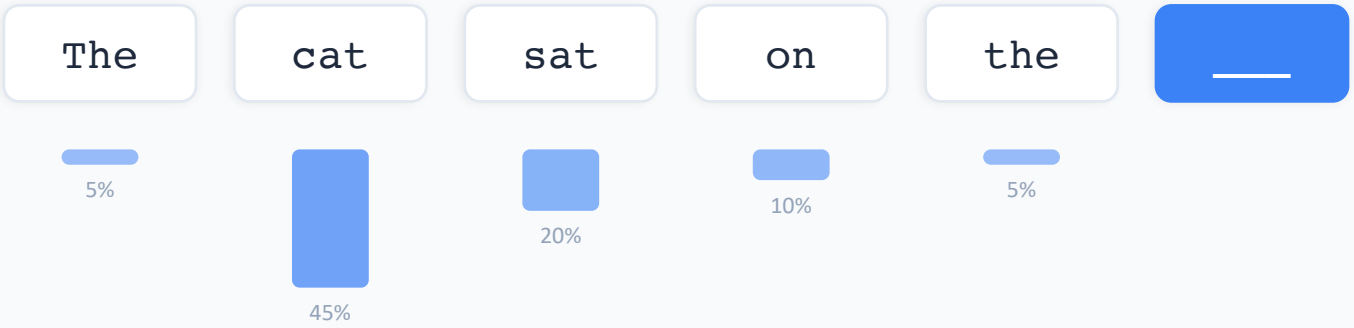
1 Attention

Which words matter most?

Where we are in the stack

LAYER 1 · ATTENTION

"What should I focus on?"



The model learns which words to pay attention to when predicting the next word.

Query / Key / Value

Query (Q)

"What am I looking for?"

match →

Key (K)

"What does each word offer?"

extract →

Value (V)

"The actual information"

Jargon Unlocked

Layer 1: Attention

Attention

Weights

Tokens

3 terms learned so far

2 Transformer

Stack attention blocks into a powerful architecture

Where we are in the stack

Stacking Attention Blocks

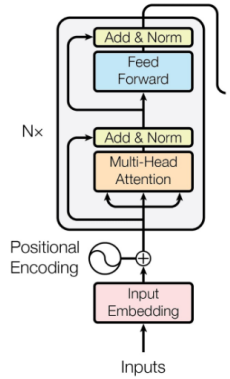
Attention + Feed-Forward Network

Attention + Feed-Forward Network

Attention + Feed-Forward Network

Attention + Feed-Forward Network

= Transformer



"Attention Is All You Need" (2017) — the paper that started everything. [1]

Jargon Unlocked

Layer 2: Transformer

Transformer

Parameters

Neural Network

References:

[1] arxiv.org/abs/1706.03762

6 terms learned so far

3 LLM

Scale up + next-token prediction

Where we are in the stack

Scale It Up → Large Language Model

Small

Millions

parameters

Complete sentences

Medium

Billions

parameters

Write paragraphs

Large

100B+

parameters

Reason, code, analyze

Training objective: predict the next word. That's it.

LAYER 3 · LLM

The LLM Landscape

GPT-5.4

OpenAI

1M context, native computer use

Proprietary

Claude Opus 4.6

Anthropic

Extended thinking, 200K context

Proprietary

Gemini 3.1 Pro

Google

1M context, multimodal

Proprietary

Llama 4 Maverick

Meta

Outperforms GPT-4o

Open Source

DeepSeek V3.2

DeepSeek

$0.28/M input tokens

Open Source

Mistral / Grok

Mistral AI / xAI

European & X alternatives

Open Source

Jargon Unlocked

Layer 3: LLM

- LLM
- Pre-training
- Next-token prediction
- Context window
- Inference

References:

[1] openai.com [2] anthropic.com/claude [3] deepmind.google/technologies/gemini

11 terms learned so far

4 Post-training

From autocomplete to assistant

Where we are in the stack

Post-training

Two Techniques

SFT — Supervised Fine-Tuning

How: Humans write example conversations — 'here's a question, here's a great answer'

Analogy:
Apprenticeship — learn by seeing expert examples

Teaches the model WHAT to do

RLHF — RL from Human Feedback

How: Model generates two responses. Humans rank: 'this one is better'

Analogy:
Performance review — learn from feedback over time

Teaches the model HOW to do it well

Jargon Unlocked

Layer 4: Post-training

Fine-tuning

SFT

RLHF

Post-training

Alignment

References:

[1] huggingface.co [2] platform.openai.com/docs/guides/fine-tuning

16 terms learned so far

5

Reasoning

Teaching models to think harder

Where we are in the stack

Reasoning

Three Approaches to Reasoning

Prompt It

"Think step by step"
Works with any model
+10-20% accuracy

Chain-of-thought prompting

Train It

Models trained with RL
to reason and self-correct
o3 scores 87% on AIME

Reasoning models (o3, R1)

Extend It

Same model, more
compute budget to think
Visible reasoning process

Extended thinking (Claude)

[1] openai.com/o3 [2] docs.anthropic.com — Extended Thinking [3] github.com/deepseek-ai/DeepSeek-R1

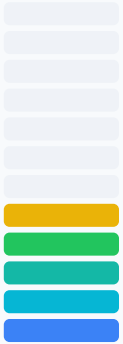
Jargon Unlocked

Layer 5: Reasoning

Chain-of-thought (CoT)

Reasoning

Extended thinking



References:

[1] openai.com/index/openai-o3 [2] docs.anthropic.com — Extended Thinking [3] github.com/deepseek-ai/DeepSeek-R1

19 terms learned so far

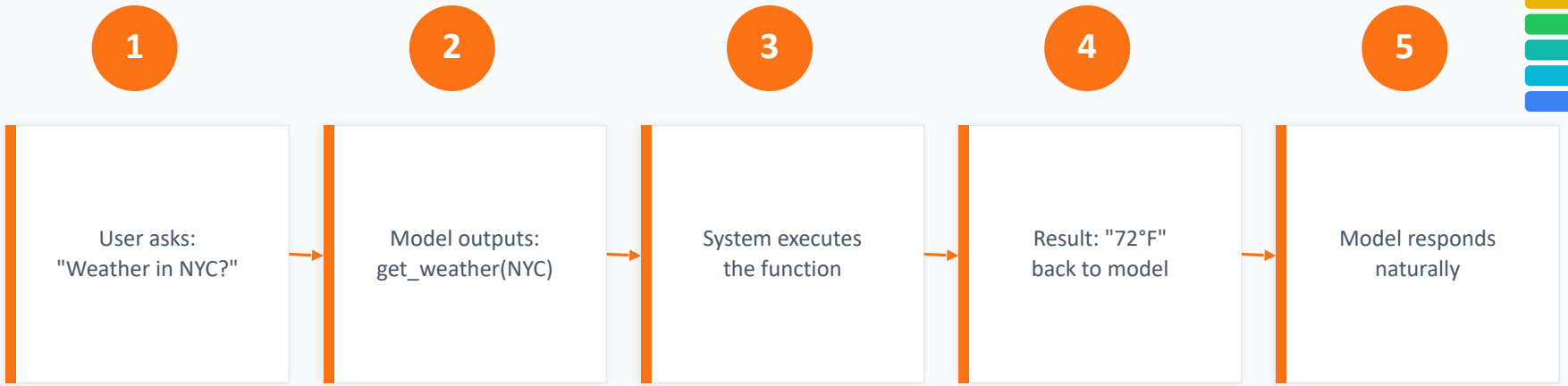
6 Tool Calling

Now the model can DO things

Where we are in the stack

Tool Calling

How Tool Calling Works



[1] docs.anthropic.com — Tool Use [2] platform.openai.com — Function Calling

Who's Actually Calling the Tool?

The Model

"I think we should call **get_weather(NYC)"**

Outputs structured text

structured output

The Harness / Product

Actually executes the tool
Handles errors
Feeds result back

ChatGPT, Claude Code, Cursor...

The model doesn't call tools. It asks for tools to be called. The harness does the rest.

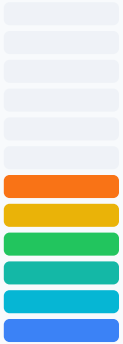
Jargon Unlocked

Layer 6: Tool Calling

Tool calling

Function calling

API



References:

[1] docs.anthropic.com — Tool Use [2] platform.openai.com — Function Calling

22 terms learned so far

7

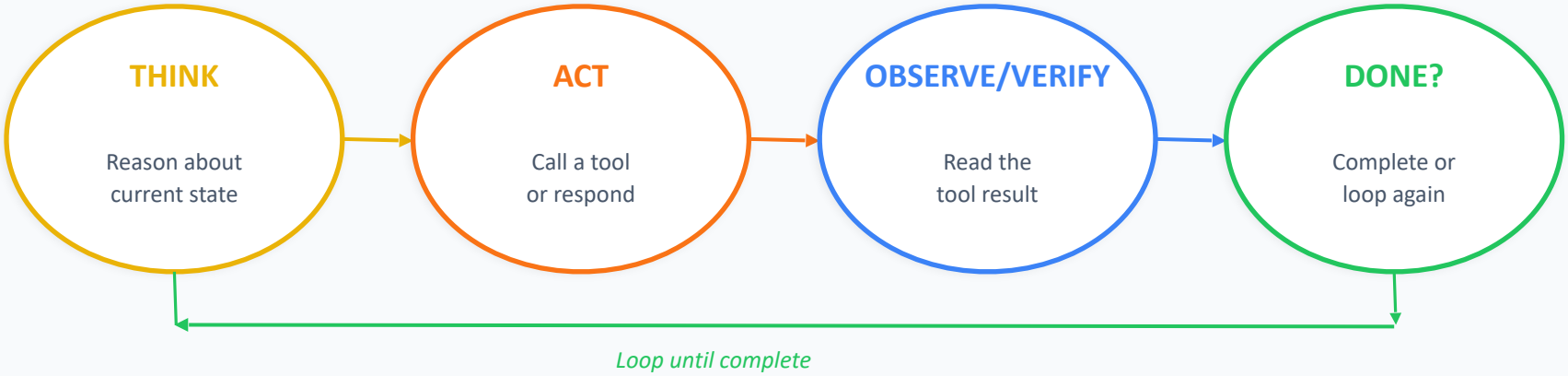
Agent Loop

LLM + Reasoning + Tools = Agent

Where we are in the stack

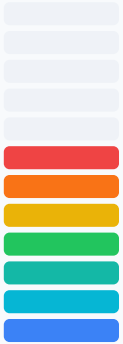
Agent Loop

The ReAct Loop



Agent Landscape

	Developer	Everyone
Local	Claude Code (46% most loved) Cursor · GitHub Copilot	Claude Cowork
Cloud	Devin V2 ($20/mo)	ChatGPT
Hybrid	OpenAI Codex	OpenClaw (216K+ stars)



Jargon Unlocked

Layer 7: Agent Loop

Agent

Agent loop

ReAct

Agentic

References:

[1] docs.anthropic.com — Claude Code [2] cursor.com [3] cognition.ai (Devin)

26 terms learned so far

8

Harness

The invisible infrastructure that runs the agent

Where we are in the stack

Harness

The Harness Flow

1. **Receive & Parse**
   User input + context
2. **Construct Prompt**
   System prompt + tools + history
3. **LLM Decides**
   Text response or tool call
4. **Dispatch Tool**
   Validate → Execute → Handle errors
5. **Feed Result Back**
   Tool output → back to LLM
6. **Loop or Complete**
   More work? Loop. Done? Return.

GUARDRAILS throughout: safety checks · token limits · cost tracking · timeout enforcement

Frameworks & SDKs

"Think of these like different car chassis — the engine (LLM) is the same, but the chassis determines how it drives."

LangChain / LangGraph

Most popular. 34.5M monthly downloads.
Graph-based orchestration.

[1] langchain.com

Claude Agent SDK

Anthropic's official SDK.
Built-in tools and guardrails.

[3] docs.anthropic.com

OpenAI Agents SDK

Minimal abstraction.
Built-in guardrails.

[4] openai.github.io

CrewAI

Role-based agent teams.
Beginner-friendly.

[5] crewai.com

AutoGen

Microsoft. Multi-agent
conversation framework.

[2] microsoft.github.io/autogen

Jargon Unlocked

Layer 8: Harness

- Harness
- Scaffolding
- SDK
- Framework

References:

[1] langchain.com [2] crewai.com [3] openai.github.io/openai-agents-python

30 terms learned so far

9

Multi-Agent

Teams of specialized agents

Where we are in the stack

Multi-Agent

Hub-and-Spoke Pattern

Landscape: LangGraph · CrewAI · n8n · AutoGen Studio · Claude Code (spawns sub-agents)

Jargon Unlocked

Layer 9: Multi-Agent

Multi-agent

Orchestration

Sub-agent

Hub-and-spoke

References:

[1] n8n.io [2] microsoft.github.io/autogen

34 terms learned so far

10

Communication

Open standards for agent communication

Where we are in the stack

Communication

What Is an Open Standard?

A publicly available specification that anyone can implement.

Before Standards

Every phone has a different charger.
Every agent needs custom integrations.
Nothing works together.

After Standards

USB-C works with everything.
One connector for any tool.
Interoperability by default.

Two Protocols, Two Problems

MCP — Model Context Protocol

Agent ↔ Tools

"USB-C for AI"
One standard connector for any tool
Managed by Linux Foundation (AAIF)

A2A — Agent-to-Agent

Agent ↔ Agent

Agents publish capability cards
Other agents discover & delegate
150+ partner organizations

Complementary, not competing. MCP = tools. A2A = agents.

MCP vs CLI — Use Both

	MCP	CLI
Cost	~550-1,400 tokens/tool	10-32x cheaper
Reliability	~72% in benchmarks	~100%
Security	Structured, auditable	OS-level (ACLs, SSH)
Discovery	Built-in (sees all tools)	Manual
Best for	External SaaS, enterprise	Local dev, fast iteration

CLI for the inner loop (fast, local). MCP for the outer loop (enterprise, SaaS).

LAYER 10 · COMMUNICATION

The Open Standards Landscape

MCP

Agent ↔ Tools

Anthropic → Linux Foundation

A2A

Agent ↔ Agent

Google (150+ partners)

AGENTS.md

Agent instructions in repos

Collaborative (AAIF)

CLI Tools

Direct tool execution

Unix heritage

Agentic AI Foundation (AAIF) — 146 members including AWS, Anthropic, Google, Microsoft, OpenAI [5]

Jargon Unlocked

Layer 10: Communication

MCP

A2A

Protocol

CLI

Open standard

References:

[1] modelcontextprotocol.io [2] a2a-protocol.org [3] aaif.io

39 terms learned so far

11

Memory & Advanced

Memory, RAG, local vs cloud, long-running agents

Where we are in the stack

Memory & Advanced

Four Types of Agent Memory

Working

Current conversation. Disappears when chat ends.

"Your desk right now"

Episodic

Past interactions. 'You prefer Python over JS'

"Personal diary"

Semantic

Factual knowledge. Company docs, codebases.

"Encyclopedia"

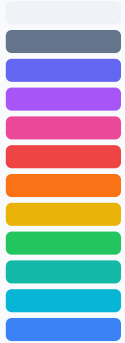
Procedural

Learned skills & patterns. How to do certain tasks.

"Muscle memory"

RAG — Retrieval-Augmented Generation

"Giving the model an open-book exam"



1

Your Docs

PDFs, wikis, Slack, code

2

Chunk & Vectorize

Split into pieces, convert to numbers

3

Search

Find matching chunks

4

Answer

LLM answers using your data

Landscape: Pinecone · Weaviate · Milvus · Chroma · LlamaIndex · Glean

Local vs Cloud Agents

Local

Runs on YOUR machine.
Data stays local. Fast.

Claude Code, Cursor,
Claude Cowork

Cloud

Remote servers.
More powerful.
Works in background.

Devin, Codex, Claude, Cursor, etc…

Hybrid

Best of both.
Where the industry
is heading.

OpenAI Codex, OpenClaw

~40% of enterprises use hybrid — local for routine, cloud for complex — cutting costs 60%

Jargon Unlocked

Layer 11: Memory & Advanced

Memory

Working memory

Episodic memory

RAG

Vector database

Sandbox

References:

[1] mem0.ai [2] pinecone.io — RAG Guide [3] ghuntley.com/ralph

45 terms learned so far

1

2

Evals

How do you know it works?

Where we are in the stack

Evals

Input → Run → Grade

Code-Based

Tests pass?
Output contains X?
Fast, cheap, objective

Model-Based

Another LLM judges quality. Flexible, handles nuance.

Human

Expert review.
Gold standard.
Expensive and slow.

pass@k = succeeds at least once in k tries · pass^k = succeeds EVERY time (production)

[1] anthropic.com — Demystifying Evals [2] swebench.com [3] braintrust.dev

Jargon Unlocked

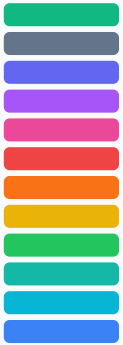
Layer 12: Evals

Evals

Benchmark

Grader

pass@k



References:

[1] anthropic.com — Demystifying Evals [2] swebench.com [3] braintrust.dev

49 terms learned so far

The Complete Stack

Every AI product, announcement, and buzzword maps to one or more of these layers.

Jargon Decoder

Can you spot the layers?

"The Agentic AI Foundation anchors founding contributions including Anthropic's Model Context Protocol (MCP), Block's goose, and OpenAI's AGENTS.md — establishing interoperable standards for orchestrating multi-agent systems at scale."

Answer: Communication (10) + Multi-Agent (9) + Open Standards

Jargon Decoder

Can you spot the layers?

Answer: LLM (3) + Post-Training (RL) (4)

Jargon Decoder

Can you spot the layers?

March 19, 2026 Safety Publication

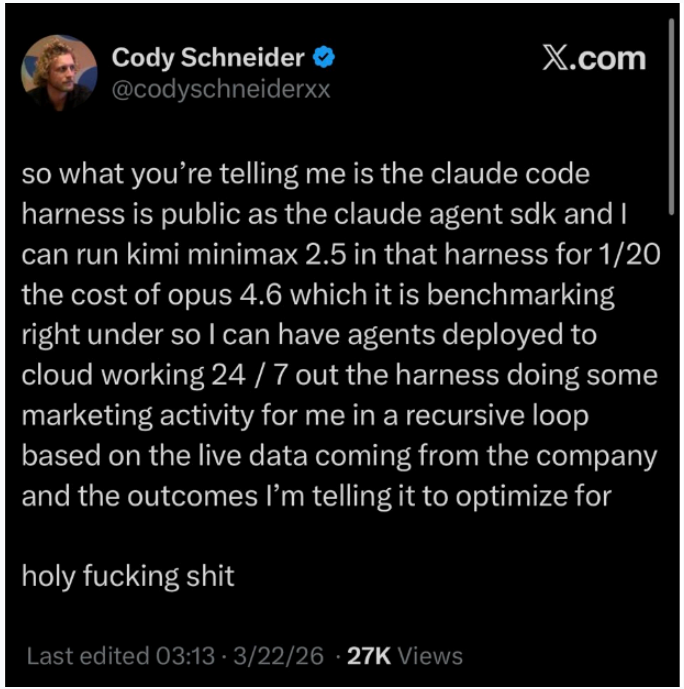
How we monitor internal coding agents for misalignment

Using our most powerful models to detect and study misaligned behavior in real-world deployments.

Answer: Post-Training (RL) (4) + Agent (7) + Harness (8)

Jargon Decoder

Can you spot the layers?



Answer: LLM (3) + Agent (7) + Harness (8) + Cloud Agents (11)

Key Takeaways

1

It's all layers

Every AI product is a combination of these building blocks

2

New releases just improve a layer

When you see an announcement, ask "which layer?"

3

The jargon maps to specific concepts

Now you know where each term lives in the stack

Thank You

Pranav Dhingra

[LinkedIn]

[Portfolio]

[No-Code Works]