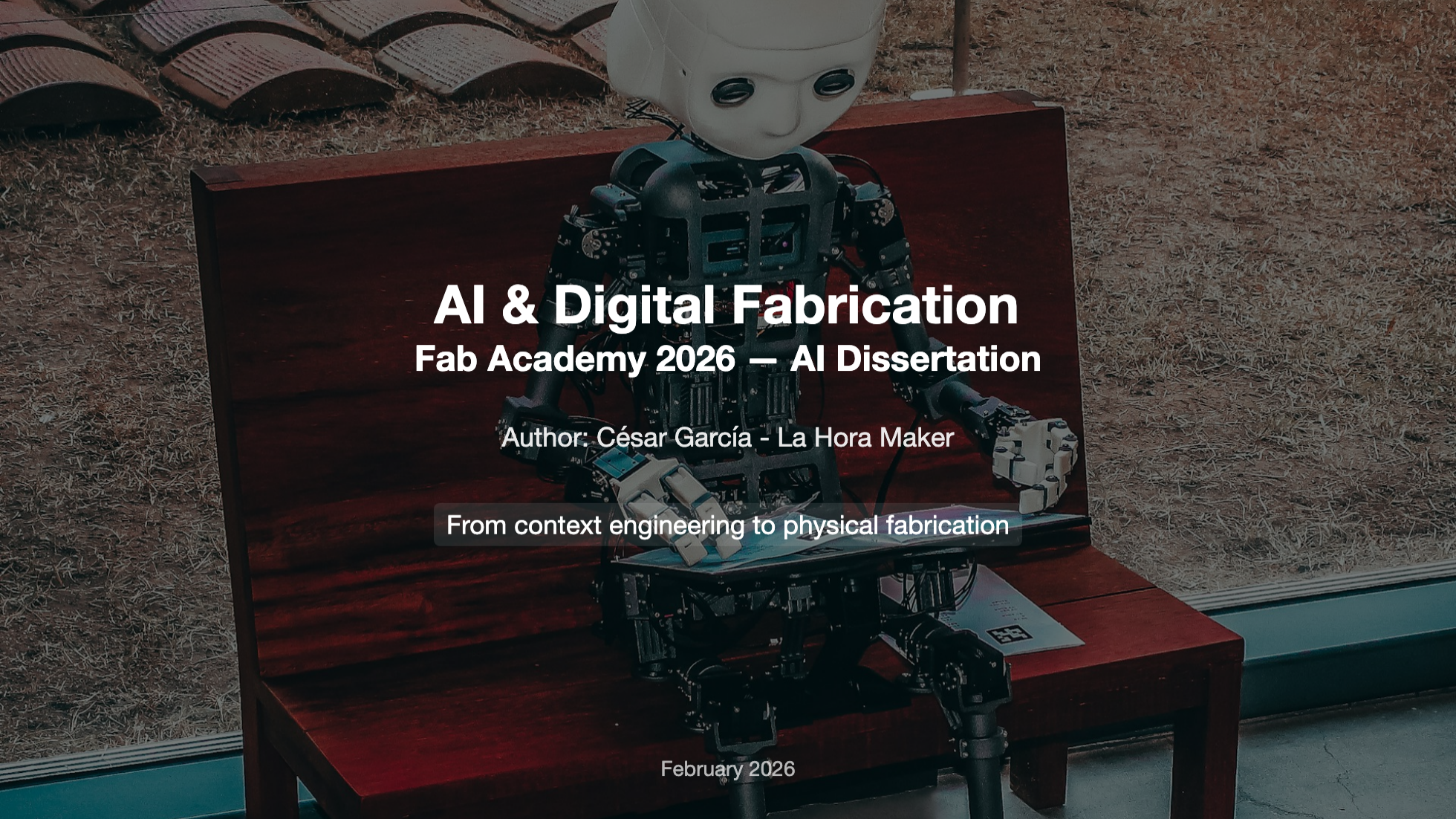


AI & Digital Fabrication

Fab Academy 2026 — AI Dissertation

Author: César García - La Hora Maker

From context engineering to physical fabrication

February 2026

🙋‍♂️ Quick Poll

Can you raise your hand if you've used…

ChatGPT? Gemini? Copilot? Others?

Agenda

Introduction to AI — GenAI, models, and architectures

Closed vs Open Models — The evolving landscape

Challenges — Scaling, energy, and learning limits

From Prompt Engineering to Context Engineering — RAG, fine-tuning, skills

MCP Servers — Connecting AI to the physical world

Development Tools — Cursor, spec-driven development, agents

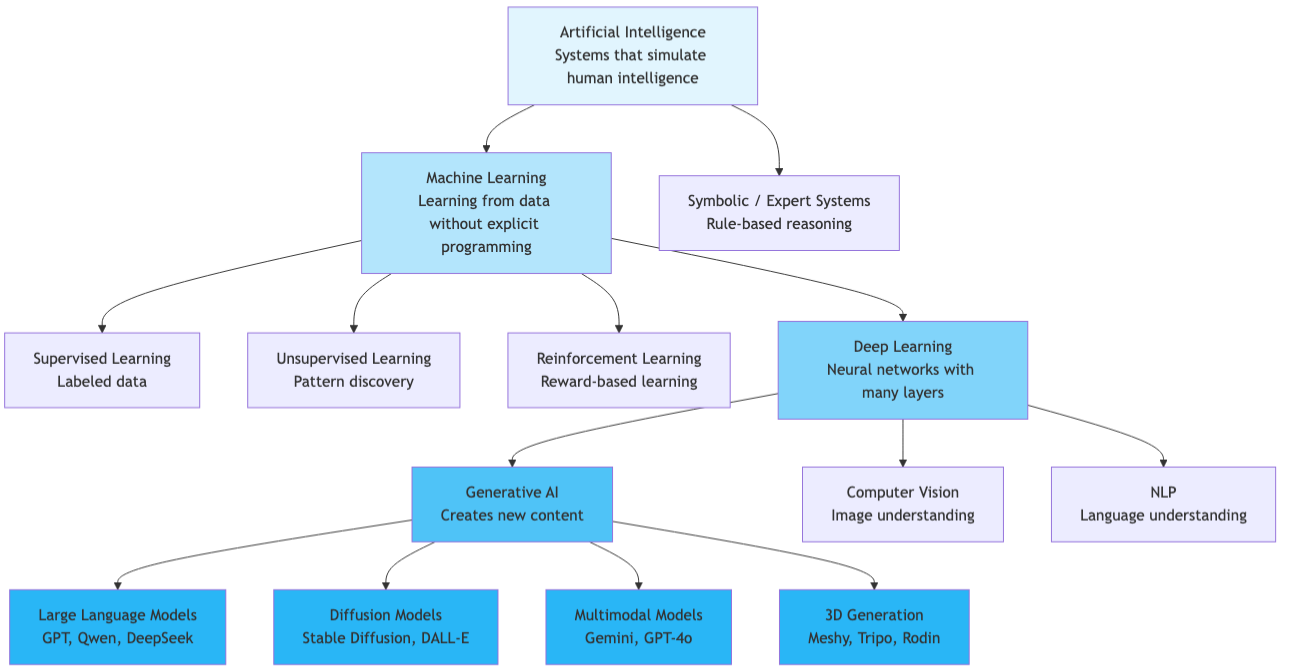
Multimodal Models — Text, image, 3D, speech, and beyond

Humans in the Loop — Feedback channels and IP considerations

Introduction to AI

What is Generative AI?

AI systems that **create new content** — text, images, code, 3D models, music, video — rather than just classifying or predicting.



Foundational Models

WHAT IS A MODEL?

- A **neural network** trained on massive datasets
- Trained on **trillions of tokens** (text, code, images)
- Training produces a set of **weights** — billions of numerical parameters
- These weights encode patterns, knowledge, and reasoning capabilities
- The model stores **statistical relationships**

THE TRANSFORMER ARCHITECTURE

The dominant architecture since 2017 ("Attention Is All You Need"):

- **Self-attention mechanism** — lets the model weigh the relevance of every part of the input
- Massively parallelizable (unlike RNNs)
- Scales with data and compute
- Powers virtually every modern LLM

```
Input → Tokenizer → Embeddings
  → [Attention Layers × N]
  → Output Probabilities
  → Next Token
```

Closed vs Open Models

The Model Landscape (2026)

CLOSED / PROPRIETARY

- **GPT-4o / GPT-5** (OpenAI)
- **Claude Sonnet / Opus** (Anthropic)
- **Gemini 3** (Google)
- Access via API or subscription
- You don't see the weights
- Generally higher performance

OPEN SOURCE / OPEN WEIGHTS

- **Qwen 3** (Alibaba) — general & coding
- **DeepSeek V3.2** — 685B params (37B active, MoE)
- **Qwen 3 Coder** — code-specialized (Ollama: `qwen3-coder`)
- **Mistral Large**
- Weights available for download
- Run locally, fine-tune, modify

The gap is closing: Open-source models trail proprietary ones by only 5–7 quality index points. DeepSeek R1 demonstrated that open-source can achieve state-of-the-art reasoning performance.

Running Models Locally

Open models mean you can run AI **on your own hardware** — no cloud, no API keys, full privacy:

LOCAL INFERENCE CLIENTS

- **Ollama** — CLI-first, dead simple: `ollama run qwen3-coder`
- **LM Studio** — Desktop GUI, discover & run models from HuggingFace
- **Open WebUI** — Self-hosted ChatGPT-like web interface, connects to Ollama
- All support **GGUF** quantized models (run on consumer GPUs or even CPU)

PINOKIO: ONE-CLICK GENAI INSTALLER

A general-purpose installer for AI programs:

- **One-click install** for ComfyUI, Stable Diffusion, Whisper, and dozens more
- Manages dependencies, Python environments, models
- Cross-platform (Windows, macOS, Linux)
- Think of it as an **app store for local AI tools**
- No terminal knowledge required

References: ollama.com | lmstudio.ai | openwebui.com | pinokio.computer

Model Distillation: Use LLM Output to Fine-Tune SLMs

Idea: Use **output from large language models** (Claude Opus, GPT-5.2, Gemini 3, etc.) as **training data** to **fine-tune much smaller models** (e.g. **Qwen 3**, **GPT-OSS**) so they replicate the same workflows locally or on edge.

Run the big model on your workflows (code, specs, fabrication steps).

Collect the LLM's **outputs** (responses, edits, decisions) — that becomes **training data**.

Fine-tune a **small language model (SLM)** on that data so it mimics the big model's behavior.

Deploy the SLM locally or on edge; **or** skip training and **update your skills** with the curated examples.

SLMs (Qwen 3, GPT-OSS, etc.) are **far smaller** than big proprietary models — distillation lets you get "big model" behavior in a small, deployable model.

Challenges

Why Can’t We Just Scale Forever?

COMPUTE & ENERGY

- **Quadratic scaling** with context length — attention is $O(n^2)$
- Doubling context doesn’t double cost — it **quadruples** it
- Requires powerful AI accelerators (GPUs/TPUs)
- Training a frontier model costs **$100M–$1B+**
- A single training run can consume as much energy as a small city

DATA & KNOWLEDGE

- Models are trained on massive datasets, but we’re running out of fresh high-quality data
- **Static models** — once trained, they don’t learn new things on their own
- Knowledge cutoff dates mean the model doesn’t know about recent events
- Hallucinations — confident but incorrect outputs
- Context window limits — even with 200K+ tokens, it’s finite

So how do we make models more capable **without** retraining from scratch?

Overcoming Limitations

Approaches to Extend Model Capabilities

IN-CONTEXT LEARNING

- Provide examples directly in the prompt
- No retraining needed
- "Few-shot" prompting
- Limited by context window

RAG

- **Retrieval-Augmented Generation**
- Connect to external knowledge bases
- Fetch relevant docs at query time
- Always up-to-date information

FINE-TUNING

- Train on your specific data
- Specialize for your domain
- Smaller, efficient models
- Your expertise encoded

FabRAG: RAG for Digital Fabrication

A project born at **Fab Lab León** during the global Fab Academy instructors' bootcamp (AI proposed by Neil Gershenfeld at Global Instructor Conference).

- Generates answers based on **Fab Academy documentation**
- Built using **Nomic gpt4all** (MIT licensed)
- **Basis:** Adam Stone Expert Map — most cited documents in Fab Academy pages, sorted by session
- **Workflow:** Expert pages → page extractor (text + images) → documents fed to RAG so responses are grounded in experts' contents
- Three development phases:
  - ✅ Core RAG functionality
  - 🔄 Authorship & licensing metadata
  - 📷 Multimodal search with images

References: FabRAG GitHub | FabRAG Phase 1 | Expert Network Map

From Prompt Engineering to Context Engineering

Let’s Do AI as if It’s 2026

PROMPT ENGINEERING (2023)

- Craft the perfect single prompt
- "You are an expert in…"
- Chain-of-thought reasoning
- One-shot interactions
- Limited context, manual effort

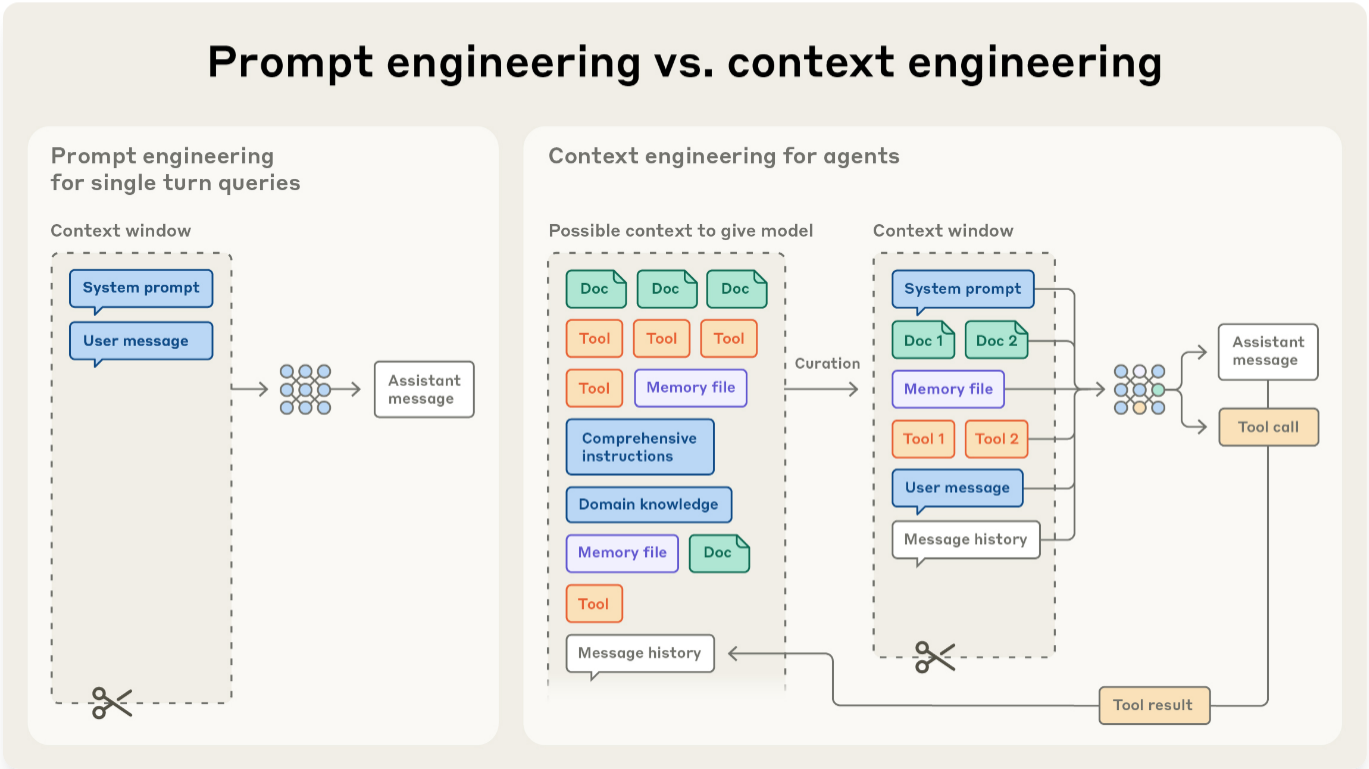
CONTEXT ENGINEERING (2026)

- **Orchestrate the entire context** the model sees
- System prompts + skills + tools + memory
- Structured specifications (not just chat)
- Persistent project context
- Automated workflows with agents

Context engineering is about **what you put around the model** — tools, documents, instructions, skills, and memory — not just the words you type into the chat box.

Effective Context Engineering for AI Agents

Effective context engineering for AI agents — Anthropic Engineering



Skills

Reusable, structured instructions that augment model capabilities:

WHAT ARE SKILLS?

- Pre-written **best practices** bundled as files
- Loaded into context when relevant
- Can include templates, examples, rules
- Shared across projects and teams
- Think of them as "expertise modules"

EXAMPLES

```
/skills/
├── pcb-design/
│   └── SKILL.md       # PCB layout rules
├── 3d-printing/
│   └── SKILL.md       # Print settings guide
├── laser-cutting/
│   ├── SKILL.md       # Material parameters
│   └── scripts/       # Executable scripts the AI can ru
└── documentation/
    └── SKILL.md       # Fab Academy docs style
```

Skills turn generic LLMs into **domain-specific assistants**. Skills can reference **executable scripts** (e.g. a `scripts/` folder under a skill) for the AI to run.

Reference: agentskills.io

MCP Servers

Connecting AI to the Physical World

What is MCP?

Model Context Protocol — Open standard (November 2024)

THE PROBLEM

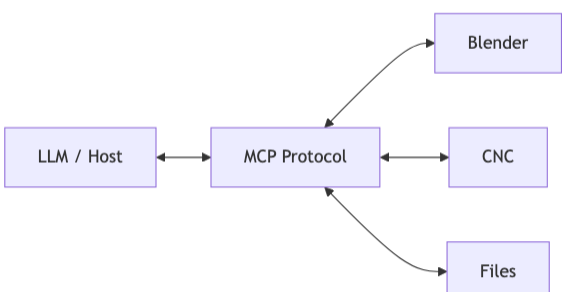
LLMs are isolated from:

- External data sources
- Real-time information
- Physical systems
- Your specific tools

THE SOLUTION

MCP provides a **standardized way** to connect LLMs with:

- APIs and databases
- File systems
- Web browsers
- **Physical machines** 🔧



Reference: modelcontextprotocol.io | MCP Servers | GitHub MCP Registry

MCP for Digital Fabrication

Tool	MCP Server	Use Case
Blender	ahujasid/blender-mcp	3D modeling, rendering
3D Printing	Community	Slicing, printer control
FreeCAD	Community servers	Parametric CAD
Inkscape	grumpydevorg/inkscape-mcps	CLI & DOM operations, SVG
Git	Official	Version control
PostgreSQL	Official	Database for projects
Puppeteer	Official	Web automation
Filesystem	Official	Local file access
Docker	Community	Container management

Blender MCP: AI-Powered 3D Modeling

WHAT IT DOES

- Control Blender with **natural language**
- Create 3D objects via text prompts
- Apply materials with AI assistance
- Adjust cameras, lighting, scenes
- Run Python code in Blender
- Import from Poly Haven & Sketchfab
- Generate 3D models via Hyper3D Rodin API

EXAMPLE WORKFLOW

```
User: "Create a low-poly tree
       with autumn colors"

LLM:  [Calls Blender MCP]
      → Creates mesh
      → Applies material
      → Sets up lighting
      → Returns viewport screenshot

User: "Now make it taller and
       add some falling leaves"

LLM:  [Modifies existing scene]
```

References: blender-mcp.com | GitHub: ahujasid/blender-mcp

MODS + MCP: The Vision

MODS — Browser-based modular tool for fab labs

WHAT IS MODS?

- Modular, node-based workflow system
- CAD/CAM/Machine control
- Runs in browser
- Powers Fab Lab machines worldwide
- Open source (MIT CBA)

THE POSSIBILITY

What if we could **control MODS via MCP**?

```
User: "Mill a PCB from this
       KiCad file on the SRM-20"

LLM:  → Generates traces PNG
      → Calculates toolpaths
      → Sends to MODS
      → Controls the mill
```

Natural language fabrication!

Reference: modsproject.org | MIT CBA GitLab

MODS + MCP: The Demo

mods-mcp — MCP server to control MODS (Mods CE) from natural language. Tested with **Claude Code**.

WHAT IT DOES

- Launches Mods CE in Chrome via Playwright; exposes **12 MCP tools**
- **Discover** programs (CNC, laser, 3D print) and **load** them into the browser
- **Configure** parameters (tool diameter, feed, depth), **load files** (SVG, PNG)
- **Execute** workflows, **export** toolpaths (RML, G-code)
- **Create** custom programs by wiring modules

EXAMPLE: PCB MILLING

```
1. load_program   → "Roland SRM-20 mill 2D PCB"
2. set_parameter  → toggle save-file switch ON
3. load_file      → load SVG into read module
4. trigger_action → "mill traces (1/64)", calculate
5. export_file    → get generated .rml toolpath
```

Machines: Roland SRM-20, Epilog, Shopbot, and more.

Repo: github.com/lahoramaker/mods-mcp · Node.js + Playwright · Config for Claude Code & Claude Desktop

MCP Hosts: Development Tools

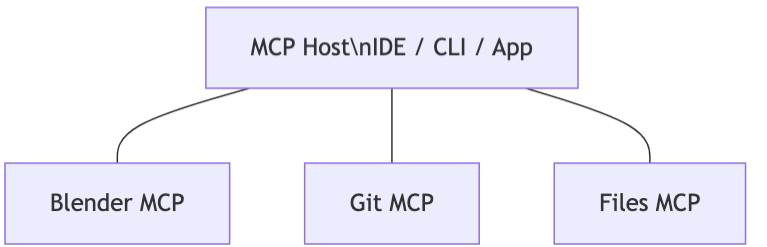
MCP Hosts: IDEs and CLI

MCP **hosts** are the applications that connect to MCP servers and let you use tools:

DESKTOP / CLI HOSTS

- **Desktop apps** — Chat interfaces with MCP support
- **CLI tools** — Terminal-native coding agents
- **IDE extensions** — Integrated into your editor
- Any host can connect to **any MCP server**

THE POWER



One host, many servers, many capabilities.

Cursor

AI-NATIVE CODE EDITOR

- Built on VS Code
- Full project context awareness
- Proactive AI suggestions
- 40–60% faster on coding tasks
- Multi-model support (GPT, Claude, etc.)
- **Tab completion** + **chat** + **composer**

CURSOR MARKETPLACE

The **Cursor Marketplace** combines:

- **Skills** — Domain-specific rules and templates
- **MCP Servers** — Tool integrations
- Browse, install, and share configurations
- One-click setup for common workflows
- Community-driven ecosystem

Think of it as an **app store for AI development context**.

Reference: cursor.sh

Claude Code

CLI-FIRST CODING AGENT

- **Claude Code** (formerly Claude for VS Code) — **CLI tool** that runs on your machine or a **remote one**
- Chat, edit, and run code with Claude; deep IDE integration where available
- **MCP support** — connect to Blender, Inkscape, filesystem, custom servers
- **Tiers:** Pro $20/mo, Max $100/mo, Team $200/mo (no free tier)

WHY USE BOTH?

- **Cursor** — Strong for code generation, composer, multi-model
- **Claude Code** — Native Claude experience, MCP, skills, local or remote
- Many developers use both depending on task
- Configure MCP servers once, use in either host

Reference: claude.ai/code (Anthropic)

Spec-Driven Development

A structured approach to working with AI coding agents:

THE IDEA

Instead of chatting back and forth, write a **specification** first:

Define requirements in a structured doc

Include constraints, edge cases, examples

Feed the spec to the AI agent

The agent implements against the spec

Iterate on the spec, not the code

Spec-Kit and **OpenSpec** (Fission-AI) bring artifact-driven workflows to your editor. Use **/opsx** commands to drive the flow:

- `/opsx:new add-feature` — start a change, create folder structure

WHY IT WORKS

- **Reproducible** — same spec, similar output
- **Reviewable** — humans review the spec, not 500 lines of code
- **Composable** — specs reference other specs
- **Version-controlled** — specs live in git
- Aligns with how AI agents work best: **clear, complete context**

OpenSpec Workflow: Full OPSX Cycle

One complete pass from idea to implemented change:

Step	Command / phase	What happens
1. New	`/opsx:new`	Start a new change; name it and choose schema (e.g. spec-driven).
2. Plan	`/opsx:continue` or `/opsx:ff`	Create **proposal** (intent, scope, approach) and **specs** (requirements, scenarios).
3. Design	`/opsx:continue`	Add **technical design** artifact (architecture, interfaces, diagrams).
4. Tasks	`/opsx:continue`	Break work into **implementation tasks** the agent can execute.
5. Apply	`/opsx:apply`	Agent implements from tasks; you can update specs/design/tasks as you go.

Agents

AI Agents and A2A

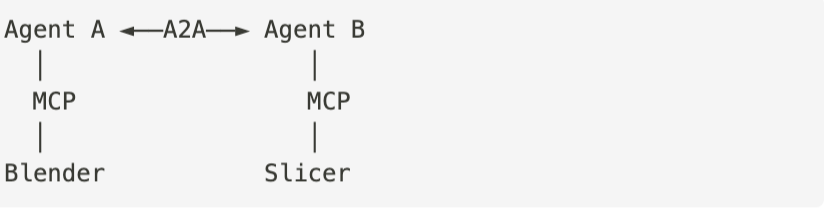
WHAT IS AN AGENT?

- An LLM that can **take actions** autonomously
- Plans, executes, observes, and iterates
- Uses tools (MCP servers, APIs, file system)
- Can run multi-step workflows
- Examples: coding agents, research agents, fabrication agents

A2A: AGENT-TO-AGENT PROTOCOL

Agent2Agent — Google's open protocol for agent communication:

- Agents discover each other's capabilities
- Delegate tasks across boundaries
- Coordinate multi-agent workflows
- Complementary to MCP



Reference: google/A2A GitHub

Multimodal Models

The Multimodal Landscape

Models that understand and generate **multiple types of content**:

🖼️ TEXT → IMAGE

- DALL-E 3, Midjourney, Stable Diffusion 3.5, Flux

🎬 TEXT → VIDEO

- Sora 2, Runway Gen-4.5, Veo 3, Kling AI

🧊 TEXT / IMAGE → 3D

- Meshy, Tripo, Sloyd, Rodin, **Trellis 2** (Microsoft)

🎤 SPEECH ⇔ TEXT

- Whisper, Deepgram, AssemblyAI

🔊 TEXT → SPEECH

- ElevenLabs, Bark, XTTS

🎵 TEXT → MUSIC

- Suno, Udio, Stable Audio

HuggingFace: The Model Hub

WHAT IS HUGGINGFACE?

- The **GitHub for AI models**
- 1M+ models available
- Datasets, spaces, and inference APIs
- Community-driven, open-source first
- Browse by modality, task, license

KEY MODALITIES ON HUGGINGFACE

Task	Examples
Text Generation	Qwen, Mistral, DeepSeek
Image Generation	Stable Diffusion, Flux
Text-to-3D	TripoSR, InstantMesh
Speech-to-Text	Whisper, wav2vec
Text-to-Speech	Bark, XTTS
Object Detection	YOLO, DETR
Image Segmentation	SAM, Mask2Former

Humans as Providers of Feedback Channels

Humans in the Loop

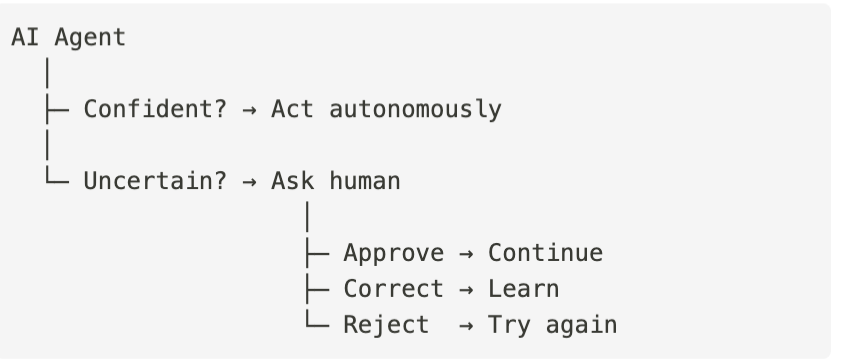
AI agents are powerful, but they need **human feedback** at critical moments:

BROWSER-BASED FEEDBACK

- **Browser extensions** — AI operates in your browser, asks you when unsure
- **Visual confirmation** — See what the AI sees, approve actions
- Cursor **Browser Mode** — AI can browse, test, and iterate on web UIs
- The human becomes a **quality gate**, not a bottleneck

Lessons learnt: Have the LLM **review past iterations** and save what worked and didn't — as **lessons learnt**. Persist in **CLAUDE.md**, **AGENTS.md**, Cursor rules, or your own file (e.g. **kaizen.md**).

WHY THIS MATTERS



The best AI workflows keep humans in control of **what matters** while automating the rest.

Intellectual Property

The Licensing Question

THE CONTROVERSY

- Training data often scraped without consent
- Copyright and intellectual property debates
- Artists and creators fighting back
- Ongoing lawsuits (Getty, NYT, etc.)
- Stable Diffusion 3D models have **different licenses based on source data**

THE OPPORTUNITY

- **Open models trained on open data** exist (but often perform worse)
- **Local deployment** = data sovereignty
- **Fine-tuning** = your expertise, your model
- **MIT/Apache** licensed options available
- Attribution and transparency matter

BEST PRACTICE

Use AI as a **tool**, not a replacement. Fine-tune models for your specific use case. Respect licenses and attribution. Consider the provenance of training data. The most powerful approach: **AI augments your expertise**.

Summary & Resources

Key Takeaways

AI is a tool, not magic — Understand its capabilities and limitations

Context engineering > prompt engineering — Orchestrate tools, skills, and memory

MCP is the bridge — Connect AI to your physical tools and machines

Agents are the future — Autonomous workflows with human oversight

Multimodal is the norm — Text, image, 3D, speech, video — all connected

Respect the ecosystem — Consider licensing, attribution, and training data provenance

Experiment and iterate — These tools are evolving week by week

Resources

MODELS & PLATFORMS

- HuggingFace
- NotebookLM
- FabRAG

SKILLS & DEVELOPMENT TOOLS

- agentskills.io
- Cursor
- Claude Code
- GitHub Copilot
- OpenSpec / Spec-Kit

MCP & FABRICATION

- MCP Protocol
- MCP Servers
- Blender MCP
- Inkscape MCP
- MODS Project

AGENTS

- A2A Protocol

LEARNING

- Fab Academy
- MIT CBA

Thank You!

Questions?

🔗 **Fab Academy 2026**: fabacademy.org

🤖 **MCP Protocol**: modelcontextprotocol.io

📚 **These Slides**: Built with Slidev