AI Agents & MCP

A chat model tells you what to do. An agent does it. MCP is how it does it.

5	100s	4	∞
AGENT COMPONENTS	MCP SERVERS	MAJOR PLATFORMS	REAL-WORLD TASKS

The agent loop

while not done:

```
thought = model.think(history + tools)
// pick a tool, or finish
action = thought.next_tool_call
observation = tools.execute(action)
history.append(thought, action, observation)
```

The 5 components

REASONING

1. Model

Opus 4.7, GPT-5, Gemini Ultra. Smart enough to plan + self-correct.

2. Tools

Functions the agent can call. Each has a name, description, parameter schema.

3. Loop / harness

The orchestration code asking model what's next, executing, returning.

4. Memory

Working memory (this conversation) + optional long-term (RAG, vectors).

5. Safety / oversight

Confirmation gates, allow/deny lists, sandboxing, step budgets. Human-in-the-loop on destructive actions.

WHAT IS MCP?

Model Context Protocol — the open standard for connecting AI agents to external tools. USB for AI. Anthropic created it, every major vendor adopted it.

TOP FAILURE MODES

Goal drift · Loop divergence · Hallucinated tool calls · Costly tangents · Prompt injection from external data

PICK BY USE CASE

1. **Sustained coding** → Claude Code
2. **Browser automation** → ChatGPT Operator
3. **Google Workspace tasks** → Gemini
4. **Real-time X / news** → SuperGrok

HIGH-VALUE AGENT TASKS

1. **Multi-file refactors** — Across your codebase
2. **App Store metadata** — Build status, release notes
3. **Cross-tool workflows** — GitHub → Linear → Slack
4. **Continuous monitors** — Morning email summary

Full guide: **djenterprises.ai/blog/ai-agents-mcp-2026**