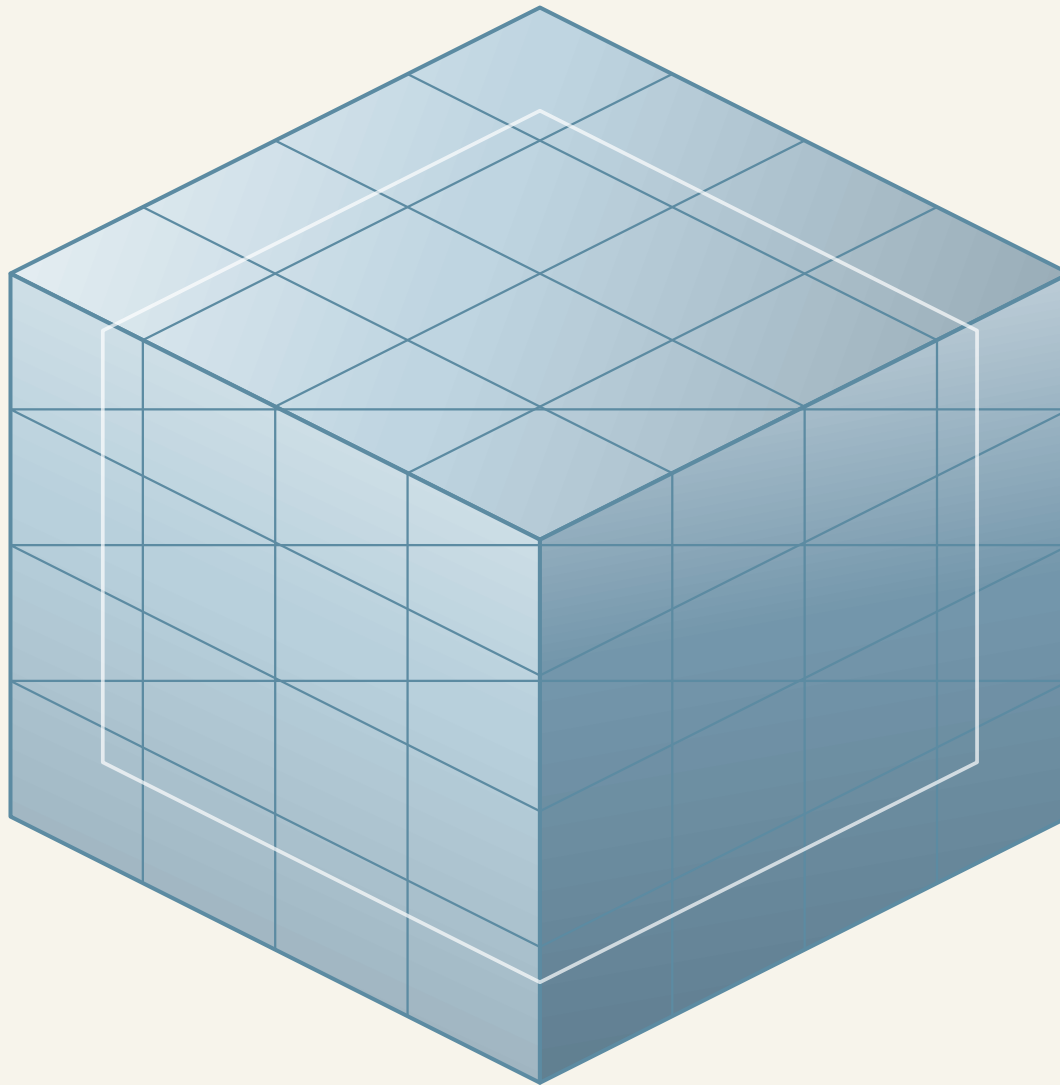




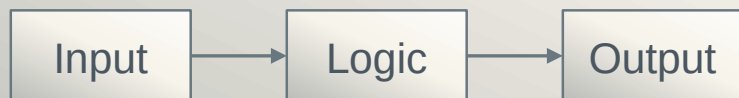
The Architecture of Agentic AI Systems

A primer on reasoning loops, memory, multi-agent topologies and protocols, plus the risks to plan for in production.

Dzung Tran · Founder, ZeroHarmAI, LLC

Why AI Agents: From Software to Autonomy

Traditional Software



- Executes predefined logic
- Strictly deterministic
- No tool orchestration
- No autonomy



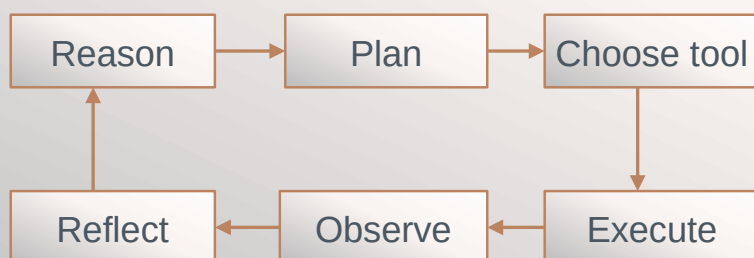
LLM Application



- Generates probabilistic responses
- No memory between calls
- Optional tool use
- Limited autonomy



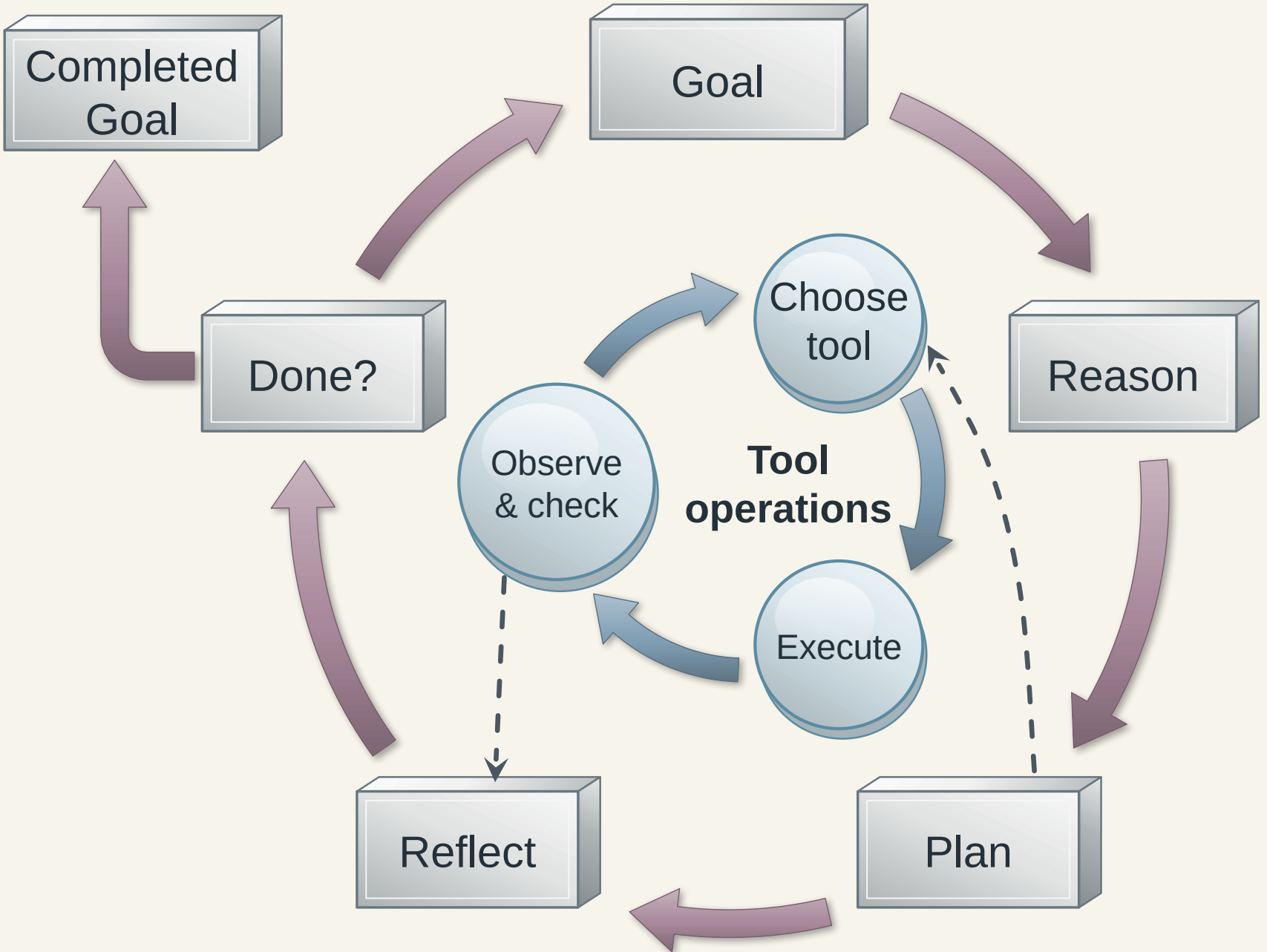
AI Agent



Repeats until the goal is met

- Reasons and acts independently
- Goal-driven planning
- Dynamic tool orchestration
- High autonomy

How an Agent Works: The Reasoning Loop



Typical sequence: Goal → Reason → Plan → Choose tool → Execute → Observe & check → Reflect → Done?

If the goal isn't met, the agent loops back and reasons again.

How an Agent Works:

Agent Memory

volatile

Short-term memory

Recent conversation and immediate context, held in the model's context window.

Working memory

In-process scratchpad: the active goal, current plan, and intermediate results.

Long-term memory

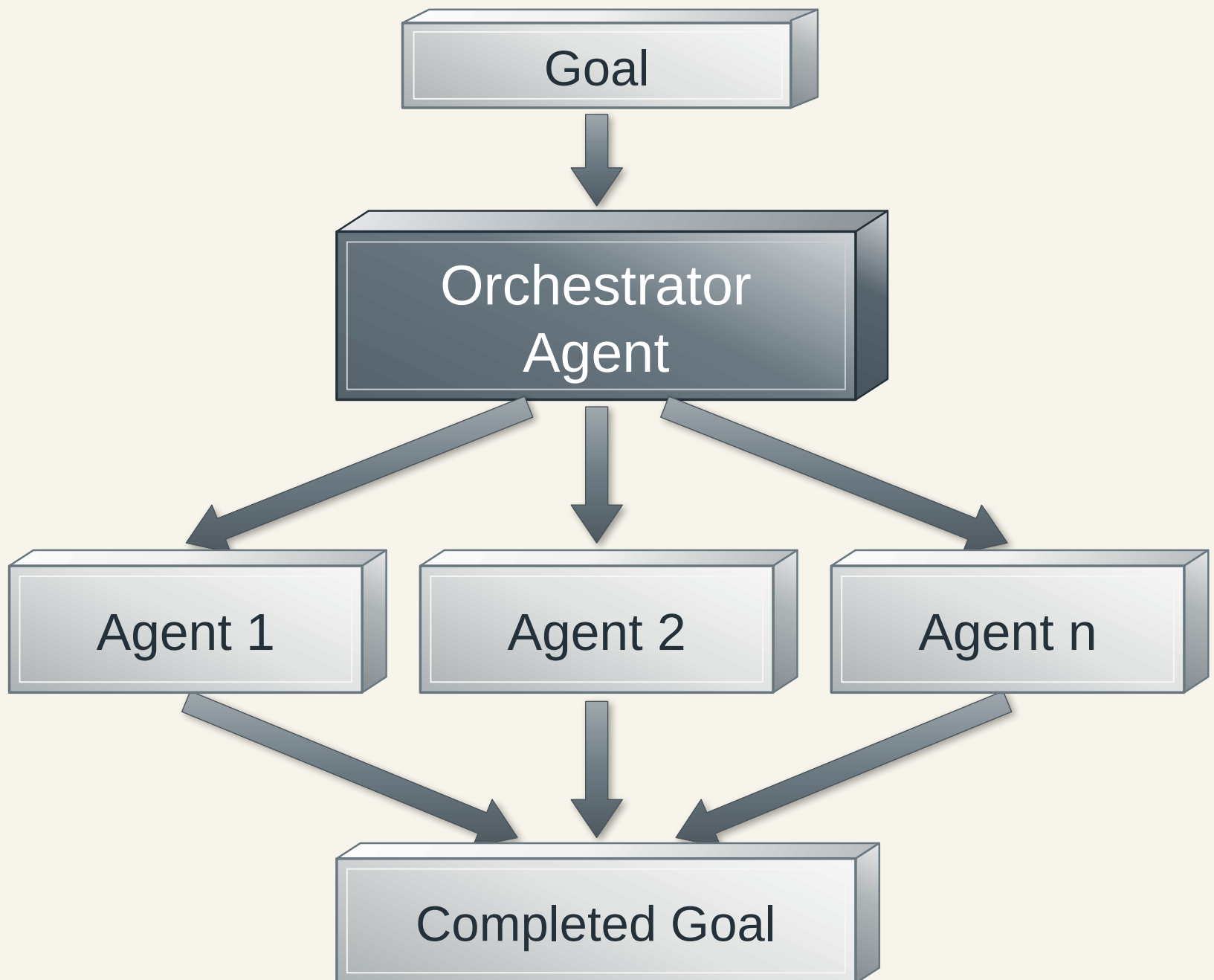
Knowledge that persists across sessions in vector databases and knowledge stores.

External systems of record

Durable stores read and written via tools (e.g. MCP): files, SQL databases, CRMs, calendars.

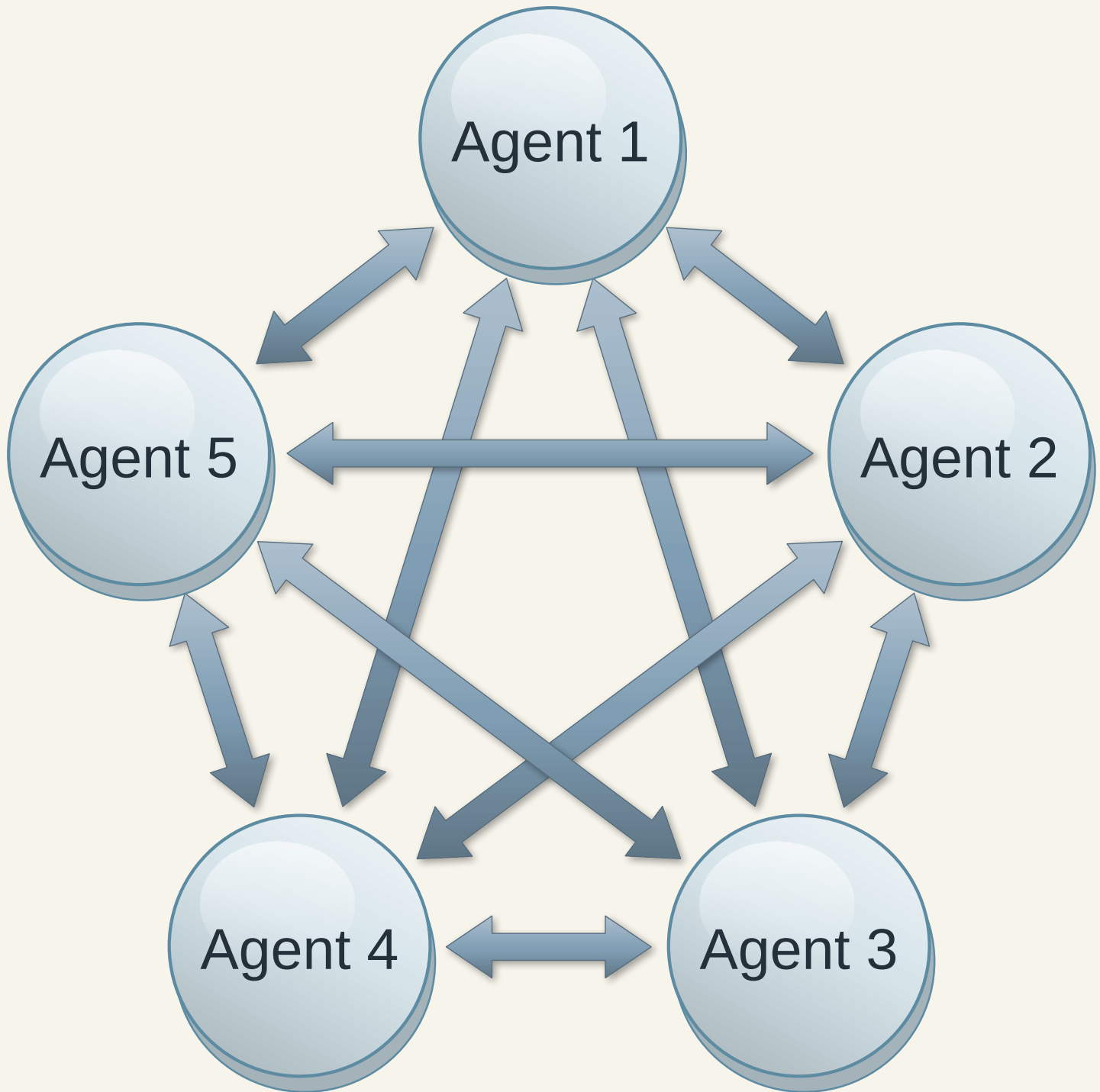
durable

Scaling Topologies: The Orchestrator



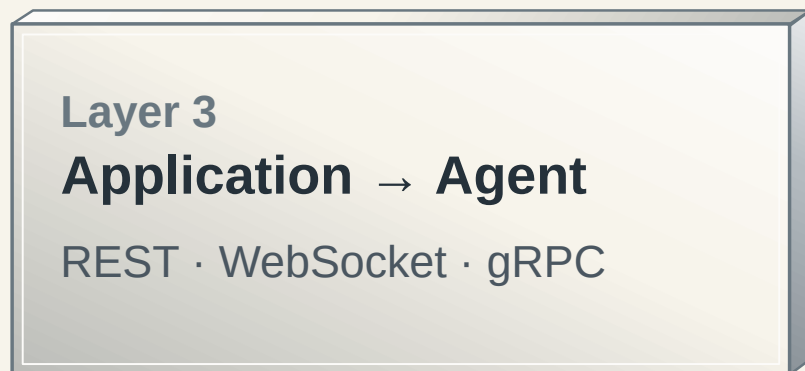
Orchestrator–worker: a top-level agent breaks down the goal and delegates to specialist agents. The most common pattern.

Scaling Topologies: Agent Mesh



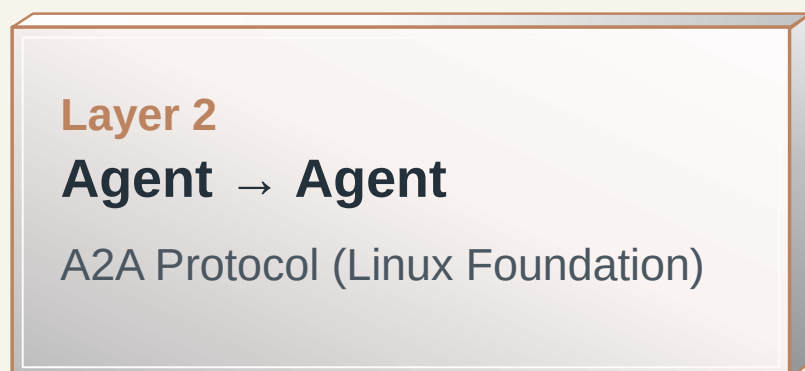
Agent mesh: agents discover and call each other directly, with no fixed hierarchy. Emerging and less common.

The Agent Protocol Stack: Apps, Agents & Tools



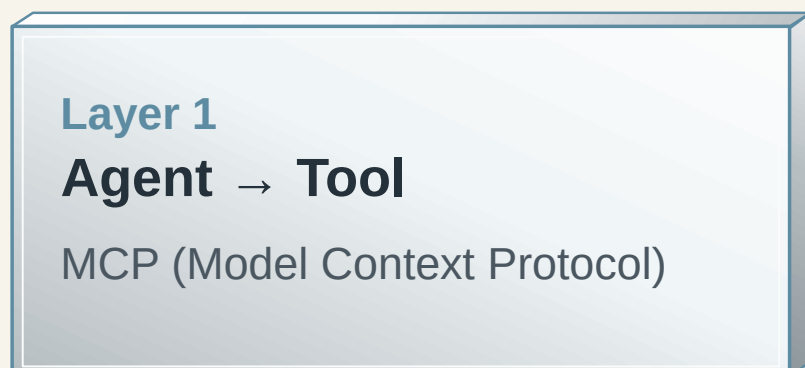
Connects the app to its agent.

No standard yet; teams use REST, WebSocket or gRPC.



Lets agents delegate to each other.

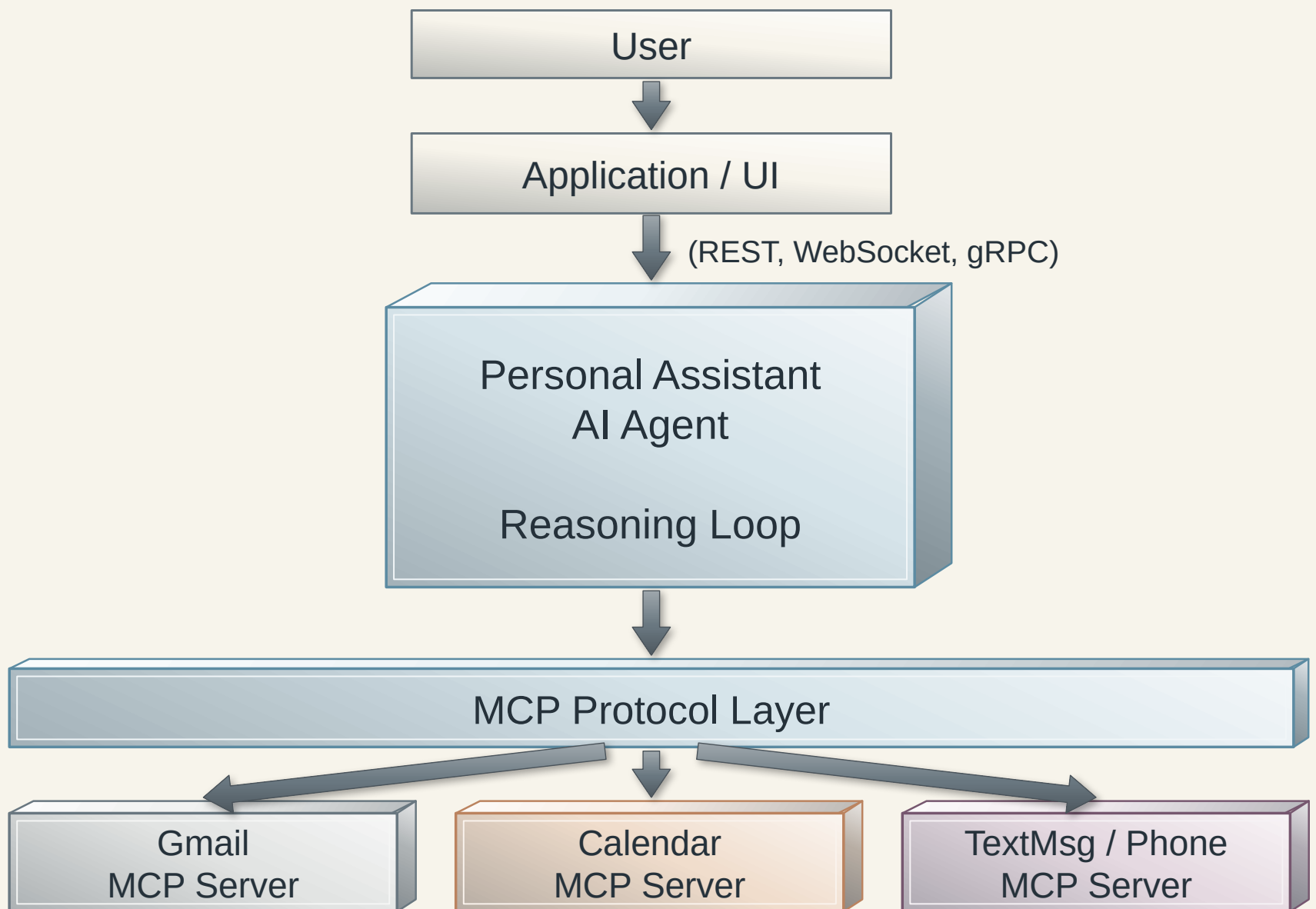
Agent Cards advertise what each agent can do.



Gives agents tools, data and prompts.

Agents call tools over JSON-RPC; results return to memory.

System in Practice: Personal Assistant AI



Example tasks

Flight confirmation:

Create calendar event.

Flight delay:

Text a family member.

Bills due in 7 days:

Add calendar reminders.

Every day at 7 AM:

Text today's schedule.

Production Reality: Risks to Design For

Trust Boundaries

Never blindly trust peers. Validate every input; each message is a potential threat.

Infinite Loops

Loop detection, maximum revision cycles, and human escalation thresholds.

Context Fragmentation

Pass state explicitly between agents and persist it in durable storage.

Accountability

If output is wrong, who is at fault? Keep immutable audit trails per agent boundary.

Hallucination & Security

Validate outputs. Approve high-impact actions. Least privilege and cost limits.

Key takeaways

- 1 One agent, one job: a single responsibility, like a container.
- 2 MCP standardizes tool access; A2A standardizes collaboration.
- 3 Production-ready agents validate every input, output and handoff.