The Agentic Primitives Framework

17 building blocks for designing, building, and governing AI agent systems

AGENTIC ACADEMY
agentic-academy.ai

ACTORS
Who participates?

Users
Human participants who interact with and oversee agents across a spectrum of autonomy.

Requesters · Operators · Administrators · Approvers

Agents
Autonomous AI entities that perceive, reason, and act toward goals using LLMs + tools.

Specialists · Orchestrators · Routers · Monitors

TOOLS
What can agents do?

Knowledge Tools READ-ONLY
Access information without modifying state. RAG, memory, search, knowledge graphs.

RAG · Memory · DB Queries · Web Search · Knowledge Graphs

Action Tools STATE-CHANGING
Modify external state, trigger processes, and cause real-world effects.

CRUD Ops · Comms · Workflow Triggers · Idempotency · Atomicity

INSTRUCTIONS
How is behavior defined?

Agent Instructions INDIVIDUAL
Persistent identity, expertise, and behavioral rules for a single agent.

Identity · Expertise · Rules · Tone

Workflow Instructions PROCESS
Step-by-step procedures: sequences, decisions, inputs/outputs, success criteria.

Deterministic · Adaptive · Runbooks · SOPs

System Instructions ORGANIZATION
Platform-wide rules and strategic objectives governing all agents.

Constraints · Intent · Guardrails · Highest Priority

COORDINATION
How is work orchestrated?

Workflow Orchestration DETERMINISTIC
Predefined sequences with known paths. Predictable, auditable, resumable.

Centralized · DAGs · BPM · Compliance-ready

Agentic Orchestration DYNAMIC
Central agent dynamically plans, decomposes, delegates, and adapts work.

Plan · Delegate · Evaluate · Adapt

Choreography DECENTRALIZED
No central controller. Agents react independently to events via shared streams.

Event-driven · Resilient · Scalable · Emergent

CONNECTIONS
How do components link?

Point-to-Point STATIC
Explicit, hardwired links between two specific components. Simple and fast.

Direct API · MCP Client→Server · Low Latency

Dynamic Discovery RUNTIME
Components find capabilities via registries at runtime. Loose coupling.

Registry · Load Balancing · Failover · Governance

Queued RESILIENT
Message infrastructure decouples sender and receiver in time and pace.

At-most-once · At-least-once · Exactly-once · DLQ

INTERACTIONS
How does communication happen?

Delegation IMPERATIVE
One actor instructs another to perform specific work and return a result.

Intent · Parameters · Sync / Async

Retrieval INTERROGATIVE
Request information without changing state. Read-only, safe to retry and cache.

Query · Cacheable · Attribution

Notification DECLARATIVE
Announce that something happened. Fire-and-forget, no expected response.

Events · State Transfer · Domain Events

Conversation COLLABORATIVE
Sustained, multi-turn contextual exchange. The only inherently stateful pattern.

Thread · Context · Memory · Multi-turn

COMMON COMPOSITION PATTERNS

Simple Assistant
`User → Agent → Knowledge Tools`
Conversational Q&A. Agent retrieves information and responds to user queries.

Autonomous Worker
`Event → Agent → Action Tools → Event`
Event-triggered autonomous action. Agent reacts, executes, and emits results.

Supervised Delegation
`User → Orchestrator → Specialists → Tools`
Hierarchical multi-agent. Orchestrator decomposes and delegates to specialist agents.

Pipeline Processing
Event → Workflow → Agent$_1$ → Agent$_2$ → Output
Sequential agent chain. Each agent processes and passes to the next stage.

Event-Driven Swarm
`System Intent → Event Stream ↔ Agents → Output`
Decentralized coordination. Autonomous agents react to shared event streams.

START SIMPLE
Minimum primitives needed. Add complexity only when required.

EXPLICIT OVER IMPLICIT
Visible, auditable connections, permissions, and instructions.

COMPOSABILITY
Primitives combine cleanly without tight coupling.

GOVERNANCE BY DEFAULT
Guardrails built into architecture, not afterthoughts.

OBSERVABILITY
Every interaction can be logged, traced, and analyzed.