Technical Playbook — Building Effective Agentic AI Systems

Production patterns for reliability, safety, evals, observability, and governance

Audience: Senior Tech Leads • CTOs • AI/MLOps Engineers • Product Leaders

◆ Start with minimum autonomy. Make tools safe. Add guardrails + evals + tracing. Govern rollouts with canaries + kill switches.

TL;DR

• Ship the smallest agent that solves the job; upgrade autonomy only when evals show a need.

• Tools are a contract: strict schemas, budgets, idempotency, and auditability.

• Guardrails + human approvals are product features, not afterthoughts.

• Tracing + evals are your safety net; failures must become replay tests.

1) The Agent Stack (4 primitives)

Agents are engineered systems built from four primitives: instructions, tools, memory/data, and orchestration.

◆ Mental model: every failure maps to one primitive. Fix the primitive—don’t just tweak prompts.

2) Orchestration Ladder

Level	Pattern	Use when	Main risk
1	Single-call + tools	one step + action	brittle prompts
2	Prompt chaining	fixed steps	latency
3	Routing	distinct categories → specialist flows	misroutes
4	Parallelization	speed or confidence	cost
5	Orchestrator–workers	dynamic decomposition	coordination bugs
6	Evaluator–optimizer loops	quality-critical outputs	loops + latency

3) Tools are product (Tool Contracts + ACI)

• Strict schemas + validators (reject ambiguous outputs).

• Least-privilege tool allowlists per agent/subagent.

• Budgets: max steps/tokens/tool calls/cost; timeouts and retries.

• Idempotency for side effects; support dry-run and staged writes.

• Audit logs and rollback plans for high-impact actions.

4) Failure modes & mitigations

◆ **Goal: small blast radius, fast detection, non-repeatable failures (replay → regression).**

Category	What breaks	Detect	Constrain	Prevent regression
Prompt injection	tool misuse, exfiltration	policy flags; anomalous tool mix	allowlists; approvals; sandbox	adversarial eval pack + replay tests
Tool hallucination	invalid args; wrong tool	schema validation failures	strict parsers; constrained retries	golden tests + contract tests
Excessive agency	does too much	cost/task spikes; step count	budgets + stop rules	KPIs in CI + canary alarms
RAG brittleness	wrong context	citation checks; retrieval metrics	top■k tuning; source allowlists	eval set with grounded answers
Looping/thrashing	runaway retries	loop counters; timeout	hard caps + backoff	regression tests on loop triggers

Failure postmortem template

- Incident summary + impact
- Reproduction trace (run_id) + inputs
- Root cause mapped to: instructions / tools / memory / orchestration
- Mitigation shipped + rollback criteria
- New replay/regression tests added

5) Governance posture

◆ Treat agent deployments like production systems with additional uncertainty: permissioning, approvals, audit, and controlled rollouts.

Permissions (capability-based)

• Deny-by-default tool access; explicit allowlists per role and environment (dev/staging/prod).

• Split read vs write tools; default to read-only.

• Scoped credentials (time-bound, per-tenant, minimal scope).

Approvals & escalation

• Risk tiers (low/medium/high/critical) with policy-driven HITL gates.

• Interrupt + resume workflow for sensitive tools (approve/edit/reject).

• Escalation path: reviewer → domain owner → security/compliance for critical actions.

Audit trails

• Immutable logs: tool calls, statuses, approvals, agent & model versions.

• Prompt hashing / config hashes for reproducible runs.

• Trace IDs preserved end-to-end for replay and incident response.

Rollout strategy

• Feature flags + progressive exposure; canary on a small cohort before full rollout.

• A/B test major behavior changes; monitor safety/quality/cost/latency KPIs.

• Automated rollback + always-on kill switch for risky tools.

One-page Agent Spec (required for each workflow)

• Goal/non-goals; autonomy level; tools + risk ratings + approval policy; memory policy; budgets/stop rules.

• Evals plan (CI/staging/prod) + observability plan (traces + dashboards).

• Rollout plan (flags/canary/kill switch) + ownership/on-call.

References (selected)

• OpenAI Agents SDK (handoffs, guardrails, tracing)

• LangGraph / LangChain (interrupts, human-in-the-loop tool approval)

• OWASP LLM Top 10 + Prompt Injection Prevention Cheat Sheet

• Google SRE Workbook (canarying releases)

• NIST AI Risk Management Framework (AI RMF 1.0)

• ISO/IEC 42001 (AI management system standard)

• Anthropic Claude Code (subagents, best practices)

• Model Context Protocol (MCP) specification

Generated: 20 Dec 2025