AGENT HARNESS VS LOOP VS GRAPH — KNOW THE DIFFERENCE

You're building AI agents the hard way

Three patterns. One right fit.

✕ WHAT MOST BUILDERS DO

- One giant loop that does everything
- No separation of concerns
- Brittle, hard to debug
- Breaks as complexity grows

✓ WHAT ACTUALLY SCALES

- **Harness, Loop, and Graph — each in its role**
- **Clear agent orchestration layers**
- **Observable, testable, maintainable**
- **Scales with your use case**

💡 **Key Takeaway:** Picking the wrong pattern early costs weeks of rewrites.

The 3 Agent Patterns

Each solves a different orchestration problem

—

INFRASTRUCTURE

Harness

Thin wrapper that connects your agent to tools, memory, and environment. The setup layer.

—

EXECUTION

Loop

The agent's run cycle — perceive, plan, act, observe — repeated until the task is done.

—

ORCHESTRATION

Graph

Nodes and edges define multi-step, conditional, or multi-agent workflows with branching logic.

Harness sets up. Loop runs. Graph routes.

Inside the Agent Loop

01
Perceive
Receive input, context, or observation

↓

02
Plan
Reason over the current state and goal

↓

03
Act
Call a tool, API, or sub-agent

↓

04
Observe
Read the result and update memory

↓

05
Repeat or Exit
Loop again or return final output

A Loop is always the same 5-step cycle — until the task is complete.

Which pattern do you need?

- [ ] **Single task, no branching → use a Loop**
- [ ] **Need tools, memory, env setup → add a Harness**
- [ ] **Multiple agents or conditional paths → use a Graph**
- [ ] **Agent calls another agent → Graph with sub-nodes**
- [ ] **Debugging a runaway loop → check your exit condition**

Use this as your decision checklist before you write a single line of agent code.

Harness vs Loop vs Graph at a glance

	BEST FOR	WATCH OUT FOR
Harness	Tool wiring, memory, env config	**Bloating it with run logic**
Loop	Single-goal, linear agent tasks	**Infinite loops, no exit guard**
Graph	Multi-agent, conditional workflows	**Over-engineering simple tasks**

Right pattern = fewer rewrites, faster debugging.

Eagentix

Agent architecture done right

Bring in the experts

eagentix.com