SAMIR GENAI ACADEMY

PROFESSIONAL STUDY GUIDE SERIES

Phase 04 · Advanced Agentic Patterns

16

Multi-Agent Orchestration

Hierarchical, peer-to-peer & event-driven agent networks

Module 16 of 25 · **1 hour**

Instructor: **Samir Kumar** · Edition 2026

STUDY GUIDE

01 Overview

> Hierarchical, peer-to-peer & event-driven agent networks. This guide gives you the core theory, the tools that matter, a hands-on build, working code, and the pitfalls to avoid — everything you need to learn the topic to a professional standard and apply it in real projects.

1 hour	4 + 4	3	16/25
LESSON LENGTH	CONCEPTS + BUILD STEPS	FRAMEWORKS & TOOLS	PHASE 04

02 Learning Objectives

- Compare hierarchical, peer-to-peer, and event-driven topologies.
- Coordinate agents via shared state and message passing.
- Assign manager/worker roles with review loops.
- Build an engineering-team agent network.

03 Core Concepts

Topologies

Hierarchical: a manager delegates to workers and integrates results — predictable and easy to control. Peer-to-peer: agents negotiate directly — flexible but harder to steer. Event-driven: agents react to a shared event bus — scalable and decoupled. Choose by control vs flexibility needs.

Coordination mechanisms

Agents coordinate through shared state (a blackboard), explicit messages, or events. Clear contracts — who owns what, what each produces — prevent the chaos and duplicated work that kill naive multi-agent setups.

Manager/worker pattern

A manager decomposes the goal, dispatches subtasks, and reviews/merges outputs, optionally looping back for revisions. This mirrors a real team and yields reliable quality control.

Cost & failure modes

More agents means more tokens and more ways to fail (loops, conflicting edits, miscommunication). Add budgets, timeouts, and a final verifier. Often a small focused team beats a large one.

04 Visual Diagram

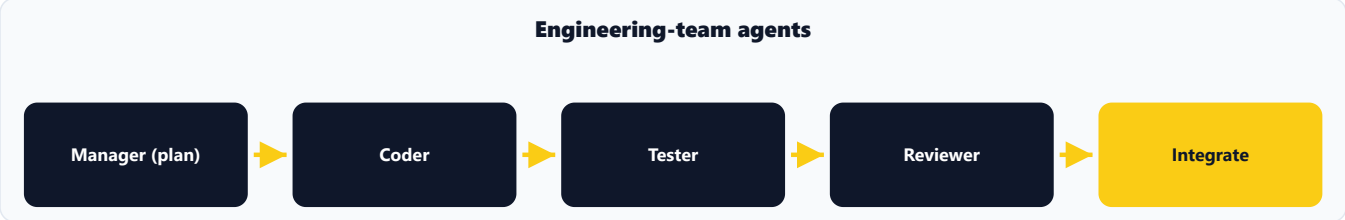

Figure 16 — A manager decomposes work; coder, tester, and reviewer collaborate with a revision loop.

05 Key Frameworks & Tools

LangGraph / CrewAI | AutoGen | Message bus / shared state

06 Hands-On Build

🛠 Engineering Team Agent (Manager, Coder, Tester, Reviewer)

1. Define manager, coder, tester, reviewer roles and contracts.
2. Manager decomposes a feature into subtasks.
3. Coder implements; tester runs checks; reviewer approves or returns.
4. Add a revision loop and a max-rounds budget.

07 Code Walkthrough

Manager dispatch + review loop (pseudo-pattern) — PYTHON

```python
def run_team(goal, agents, max_rounds=3):
    plan = agents["manager"].decompose(goal)
    for task in plan:
        for _ in range(max_rounds):
            code = agents["coder"].do(task)
            report = agents["tester"].test(code)
            if agents["reviewer"].approve(code, report):
                break
            task = agents["reviewer"].feedback(code, report)
    return agents["manager"].integrate()
```

08 🧪 Lab Assistant

Lab 16 · Manager/worker team

Prerequisites: LangGraph or CrewAI · An API key

1. **Give the manager a goal to decompose into subtasks**
   > Expected: A concrete task list
2. **Run coder→tester→reviewer with a revision loop**
   > Expected: Code iterates until the reviewer approves
3. **Add a max-rounds budget**
   > Expected: The loop always terminates

Troubleshooting

- Agents clash → define clear ownership contracts.
- Endless revisions → cap rounds and add a final verifier.

09 Pitfalls & Key Takeaways

⚠ COMMON PITFALLS

- Agents duplicating or overwriting each other's work with no contracts.
- No round budget, so revision loops never terminate.
- Scaling agent count before proving the topology works.

✓ KEY TAKEAWAYS

- Match topology (hierarchical/P2P/event) to control needs.
- Clear contracts and shared state prevent multi-agent chaos.
- Budgets, timeouts, and a final verifier keep teams reliable.

📺 **Companion video:** Watch the module 16 lesson in your Student Dashboard, then use this guide to revise, copy the code, and complete the build. Download both for offline study.

10 Further Resources

AutoGen docs • LangGraph multi-agent guides • CrewAI hierarchical process