Multi-Agent Architecture Decision Framework

Pattern Selection Guide

Use this framework to determine which multi-agent architecture fits your business scenario.

Start Here: Core Questions

1. Do you need centralized control over the workflow?

- YES → Consider Hierarchical (Manager-Worker)
- NO → Go to Question 2

2. Do agents need to work together on shared goals with access to common data?

- YES → Consider Peer-to-Peer (Collaborative)
- NO → Go to Question 3

3. Do agents need to react independently to system events without tight coupling?

- YES → Consider Event-Driven (Pub-Sub)

Objectives

Factor	Hierarchical	Peer-to-Peer	Event-Driven
Control	Centralized manager	Distributed across agents	Event-based triggers
Coupling	Tight (manager knows workers)	Medium (shared dataverse)	Loose (event bus only)
Coordination	Manager orchestrates	Collaborative consensus	React independently
Scalability	Limited by manager capacity	Scales with shared state	Highly scalable
Complexity	Simple chain of command	Moderate (state management)	Complex (async handling)

When to Choose Each Pattern

Hierarchical (Manager-Worker)

Choose when you need:

- Clear approval workflows
- Predictable task sequences
- Single point of decision-making
- Result aggregation from multiple sources

Real examples:

- Document approval processes
- Multi-step customer onboarding
- Report generation from multiple data sources

Peer-to-Peer (Collaborative)

Choose when you need:

- Agents to share context and make joint decisions
- Common data access across autonomous agents
- Flexible collaboration without rigid hierarchy
- Agents with equal authority

Real examples:

- Team collaboration tools
- Shared project management
- Multi-department request handling

Event-Driven (Pub-Sub)

Choose when you need:

- High volume, asynchronous processing
- Independent agent reactions to system changes
- Minimal dependencies between agents
- Easy addition/removal of agents

Real examples:

- Real-time notification systems
- Inventory updates across systems
- Automated workflow triggers
- IoT sensor processing

Decision Matrix

Your Requirement	Recommended Pattern
Need strict workflow order	Hierarchical
One agent must oversee others	Hierarchical
Agents need shared memory	Peer-to-Peer
Multiple agents, equal authority	Peer-to-Peer
High transaction volume	Event-Driven
Agents work independently	Event-Driven
Need to scale quickly	Event-Driven
Async processing critical	Event-Driven

Red Flags by Pattern

Don't choose Hierarchical if:

- You need high scalability (manager becomes bottleneck)
- Agents should operate independently
- Workflow steps change frequently

Don't choose Peer-to-Peer if:

- You need strict process control
- Shared state becomes too complex
- Clear authority hierarchy is required

Don't choose Event-Driven if:

- You need synchronous responses
- Strong consistency is critical
- Debugging complex flows is difficult for your team