# Cloud Run Deployment Models for LangGraph-Based Heterogeneous Agent Systems

This document outlines three architectural options for deploying LangGraph-based heterogeneous agent systems on Google Cloud Run, highlighting trade-offs, best-fit use cases, and practical setups.

---

Option 1: One Cloud Run Service, One Endpoint

Overview:

- Deploy a single application (e.g., using FastAPI or Flask).
- The supervisor agent handles incoming HTTP requests.
- All sub-agents are defined internally and invoked as in-process functions.

Architecture:

Client --> [ Cloud Run Service ] --> Supervisor --> Sub-Agents (in-memory calls)

Characteristics:

- Only one public endpoint (e.g., /supervisor).
- No external access to sub-agents.
- Fast and simple communication via Python function calls.

Best For:

- Centralized systems
- Lower operational overhead
- Early-stage prototypes
- Cost-sensitive deployments

---

Option 2: Multiple Cloud Run Services (One per Agent)

Overview:

- Each agent (supervisor + sub-agents) is deployed as a separate Cloud Run service.

- Communication between supervisor and sub-agents occurs over HTTP.

Architecture:

```
Client --> [ Supervisor Service ] --> HTTP --> [ Sub-Agent 1 Service ]
--> HTTP --> [ Sub-Agent 2 Service ]
--> HTTP --> [ Sub-Agent N Service ]
```

Characteristics:

- Each service has its own endpoint.
- Agents operate independently and can be scaled individually.
- More complex DevOps and networking setup.

Best For:

- Microservice architectures
- Need for individual scaling, observability, or fault isolation
- Shared agents across multiple systems

---

Hybrid Option: One Cloud Run Service, Internal Routes

Overview:

- Deploy one FastAPI/Flask app with multiple routes.
- Supervisor agent is the default route.
- Each sub-agent is exposed via a separate internal route (e.g., /cloud_ops, /sysadmin).

Architecture:

```
Client --> [ Cloud Run Service ]
|-- /supervisor --> Supervisor logic --> Internal call to sub-agent
|-- /cloud_ops  --> Direct to sub-agent
|-- /sysadmin   --> Direct to sub-agent
```

Characteristics:

- One deployed service with multiple logical endpoints.
- Sub-agents are accessible for testing, debugging, or internal reuse.
- Maintains centralized control while allowing modular access.

Best For:
- Centralized logic with partial modularity
- Teams wanting easy testing or debugging of sub-agents
- Smooth transition to microservices later

Each approach has trade-offs in terms of scalability, complexity, and development speed. Choose based on your system needs and how independently each agent must operate.