
Federated Swarm

Overview:

A Federated Swarm architecture involves multiple independent swarms collaborating to complete a task. Each swarm operates autonomously but can share information and results with other swarms.

Use-Cases:

- Distributed learning systems where data is processed across multiple nodes.

- Scenarios requiring collaboration between different teams or departments.

```mermaid
graph TD
    A[Central Coordinator]
    subgraph Swarm1
        B1[Agent 1.1] --> B2[Agent 1.2]
        B2 --> B3[Agent 1.3]
    end
    subgraph Swarm2
        C1[Agent 2.1] --> C2[Agent 2.2]
        C2 --> C3[Agent 2.3]
    end
    subgraph Swarm3
        D1[Agent 3.1] --> D2[Agent 3.2]
        D2 --> D3[Agent 3.3]
    end
    B1 --> A
    C1 --> A
    D1 --> A
```

Star Swarm

Overview:
A Star Swarm architecture features a central agent that coordinates the activities of several peripheral agents. The central agent assigns tasks to the peripheral agents and aggregates their results.

Use-Cases:
- Centralized decision-making processes.

- Scenarios requiring a central authority to coordinate multiple workers.

```mermaid
graph TD
    A[Central Agent] --> B1[Peripheral Agent 1]
```

    A --> B2[Peripheral Agent 2]
    A --> B3[Peripheral Agent 3]
    A --> B4[Peripheral Agent 4]
~~~

Mesh Swarm

Overview:
A Mesh Swarm architecture allows for a fully connected network of agents where each agent can communicate with any other agent. This setup provides high flexibility and redundancy.

Use-Cases:
- Complex systems requiring high fault tolerance and redundancy.

- Scenarios involving dynamic and frequent communication between agents.

```mermaid
graph TD
    A1[Agent 1] --> A2[Agent 2]
    A1 --> A3[Agent 3]
    A1 --> A4[Agent 4]
    A2 --> A3
    A2 --> A4
    A3 --> A4
~~~

---

### Cascade Swarm

**Overview:**
A Cascade Swarm architecture involves a chain of agents where each agent triggers the next one in a cascade effect. This is useful for scenarios where tasks need to be processed in stages, and each stage initiates the next.

**Use-Cases:**
- Multi-stage processing tasks such as data transformation pipelines.

- Event-driven architectures where one event triggers subsequent actions.

```mermaid
graph TD
    A[Trigger Agent] --> B[Agent 1]
    B --> C[Agent 2]
    C --> D[Agent 3]
    D --> E[Agent 4]
```

Hybrid Swarm

Overview:

A Hybrid Swarm architecture combines elements of various architectures to suit specific needs. It might integrate hierarchical and parallel components, or mix sequential and round robin patterns.

Use-Cases:

- Complex workflows requiring a mix of different processing strategies.

- Custom scenarios tailored to specific operational requirements.

```mermaid
graph TD
    A[Root Agent] --> B1[Sub-Agent 1]
    A --> B2[Sub-Agent 2]
    B1 --> C1[Parallel Agent 1]
    B1 --> C2[Parallel Agent 2]
    B2 --> C3[Sequential Agent 1]
    C3 --> C4[Sequential Agent 2]
    C3 --> C5[Sequential Agent 3]
```

These swarm architectures provide different models for organizing and orchestrating large language models (LLMs) to perform various tasks efficiently. Depending on the specific requirements of your project, you can choose the appropriate architecture or even combine elements from multiple architectures to create a hybrid solution.