NANYANG
THE INNOVATIVE POLYTECHNIC

Generative AI MultiAgents

Learning Outcomes

Upon completion of this session, the learners should be able to:

- Explain the principles and benefits of multi-agent systems, including role specialization, task decomposition, communication protocols, and coordination strategies.
- Implement multi-agent workflows using advanced frameworks (e.g.CrewAI, LangGraph, AutoGen), incorporating shared memory management, orchestration, and failure recovery mechanisms.

Why we need to have Agents?

Complex decision-making:

Workflows involving nuanced judgment, exceptions, or context-sensitive decisions, for example refund approval in customer service workflows.

Difficult-to-maintain rules:

Systems that have become unwieldy due to extensive and intricate rulesets, making updates costly or error-prone, for example performing vendor security reviews.

Heavy reliance on unstructured data:

Scenarios that involve interpreting natural language, extracting meaning from documents, or interacting with users conversationally, for example processing a home insurance claim.

Before committing to building an agent, validate that your use case can meet these criteria clearly

What is a Generative AI MultiAgent?

Why Use Multiple Agents?

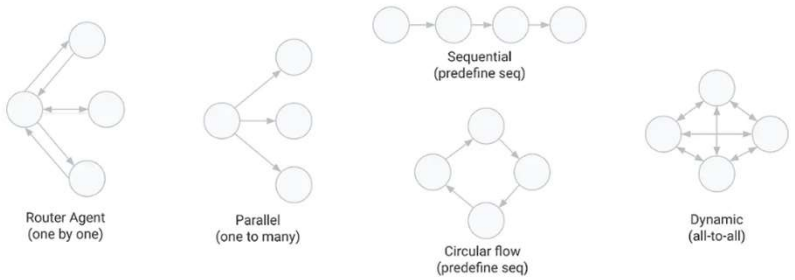
- **Single-agent limitation**: One LLM agent trying to solve everything often struggles with:
  - Context overload (too many instructions at once).
  - Weak specialization (generalist vs. expert).
  - Lack of modularity (hard to scale or debug).
- **Multi-agent benefits**:
  - **Role specialization** → different agents act as experts (e.g., planner, coder, reviewer).
  - **Parallelism** → tasks can be split among agents for faster execution.
  - **Modularity** → easier to upgrade or swap out one agent without redesigning the whole system.
  - **Human-like teamwork** → mirrors real-world collaboration.

Multi-Agent Planning & Task Decomposition

- **Hierarchical breakdown:**
  - Planner agent → breaks down the big problem into steps.
  - Worker agents → execute individual tasks.
  - Reviewer agent → checks correctness and quality.
- **Decomposition example: Build a website with product recommendations**
  - Planner: Defines steps → Data collection → Recommendation engine → Web design.
  - Data Agent: Retrieves product info.
  - ML Agent: Builds recommendation model.
  - UI Agent: Designs frontend.
  - Reviewer: Validates correctness.

Multiagent Pattern

- In the Router pattern, the router agent is using other agents as specialized tools, calling them one by one as needed.
- In the Parallel pattern, the initial agent is using multiple agents as tools simultaneously to speed up the process.
- In the Sequential and Circular Flow patterns, agents are used as tools in a predefined pipeline or loop.
- In the Dynamic pattern, agents are interacting more like a team, with each agent acting as both a user and a tool for other agents, depending on the situation.
- These patterns provide different ways to structure multi-agent interactions, allowing developers to choose the most appropriate approach based on the specific requirements of their application



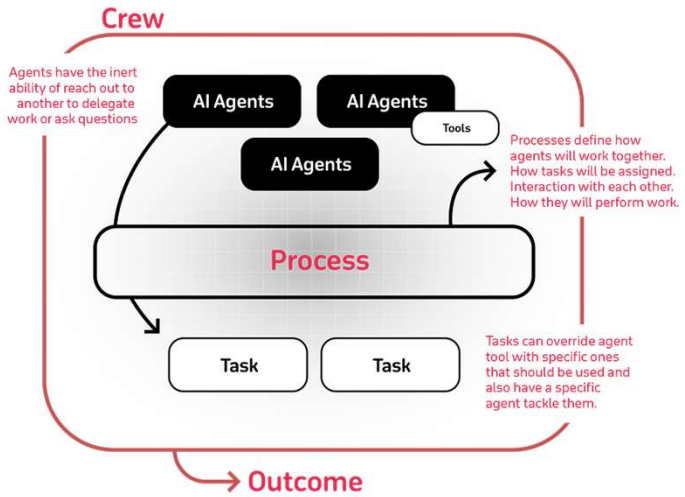
Message Passing & Communication Protocols

- **Direct messaging**: Agent A sends structured text to Agent B.
- **Shared memory**: Agents post updates in a shared space.
- **Protocols**:
  - JSON-based structured messages
  - Tool invocation signals (like OpenAI's function calling)
  - Pub/Sub messaging (subscribe to updates on a task)
- **Key challenges**:
  - Ensuring clarity (avoid ambiguous prompts).
  - Preventing infinite loops (agents endlessly asking each other).
  - Controlling cost (token usage increases with more agents).

Frameworks & Tools

CrewAI

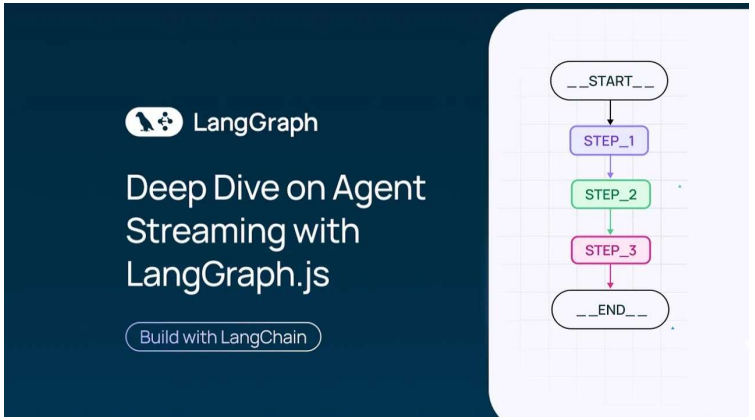
- Purpose: Organize **teams of agents** with defined roles and collaboration.
- Example roles: *Researcher, Analyst, Writer, Critic*.
- Supports **task delegation** + **team workflows**.
- Example use case: Generate a market report with multiple specialized agents.



Frameworks & Tools

LangGraph

- Model agent workflows as **state machines / graphs**.
- Nodes = agents or functions.
- Edges = transitions (who speaks next, what condition triggers a move).
- Good for **deterministic orchestration** (avoids chaos of "agents talking freely").
- Example: Chatbot that can switch between *Sales Agent → Technical Agent → Escalation Agent* based on user intent.



Frameworks & Tools

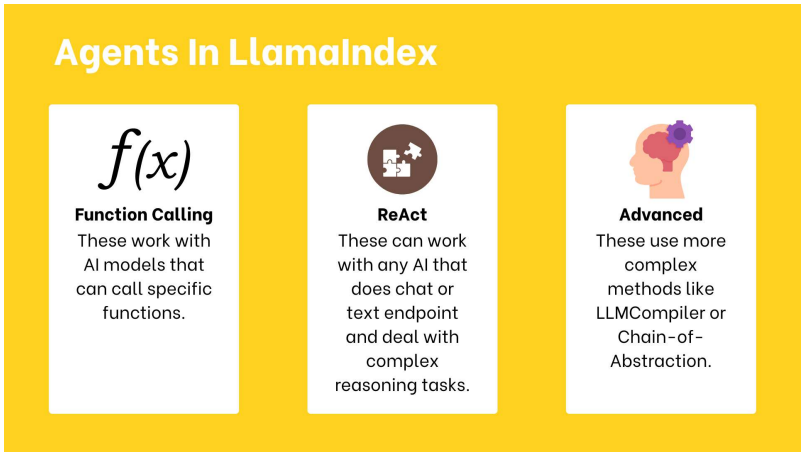
AutoGen (Microsoft)

- Multi-agent conversation framework.
- Agents can:
  - Collaborate (e.g., Coder ⟷ Reviewer loop).
  - Autonomously call tools.
  - Escalate to human-in-the-loop.
- Supports **“GroupChat” pattern** for open discussion among multiple agents.

Frameworks & Tools

LlamaIndex Agent

- Builds **data-aware agents** that can reason over structured + unstructured data.
- Uses **indexes** (vector, keyword, graph) for memory and context grounding.
- Supports **tool integration** (DB queries, APIs, Firecrawl, etc.).
- Works well as the **Knowledge Agent** inside multi-agent workflows.
- Example: Research Assistant agent that searches documents, summarizes results, and shares insights with a Planner or Reviewer agent.



Control & Coordination

Agent Orchestration

- **Centralized orchestration**: One “controller” agent assigns tasks.
- **Decentralized orchestration**: Agents coordinate via rules/messages, no single leader.
- Orchestration prevents chaos and ensures progress.

Managing Shared Memory or Knowledge

- Options:
  - Centralized memory store (vector DB, JSON file, blackboard).
  - Agent-specific memory (each agent has its own log/history).
  - Hybrid: Agents keep local context but sync key insights to global memory.
- Example: Planner maintains global plan, workers only store their progress.

Control & Coordination

Stopping Conditions & Failure Recovery

- Stopping conditions:
  - Goal achieved (final output validated).
  - Time/cost exceeded (budget control).
  - Deadlock detected (agents stop responding meaningfully).
- Failure recovery:
  - Retry mechanism with backoff.
  - Escalate to human (human-in-the-loop).
  - Replace failed agent with fallback.

References

- https://www.voicespin.com/glossary/voice-activity-detection/
- https://www.youtube.com/watch?v=xWhI8RkRSGQ&t=60s