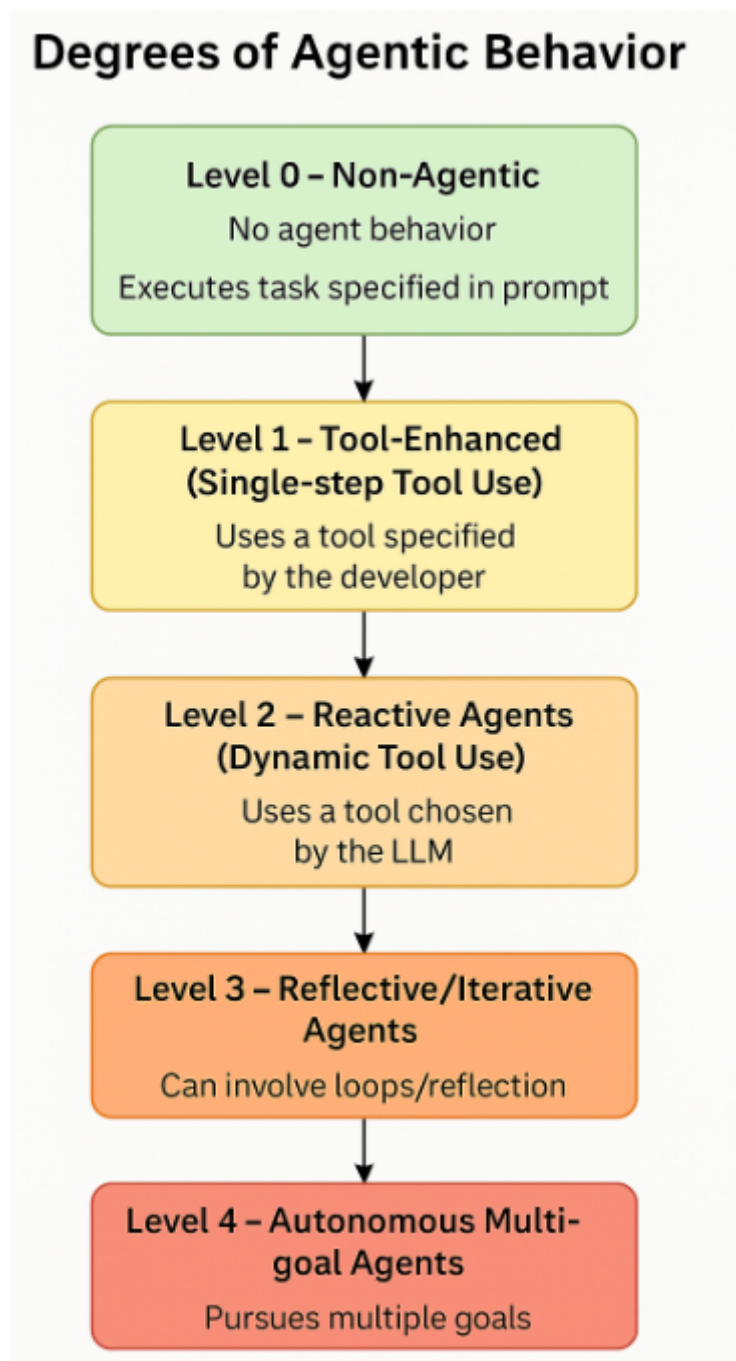


Cấp độ hành vi của tác nhân(Agentic) trong ứng dụng LLM



LLMs như GPT, Claude, hay Gemini, ngày càng nhiều ứng dụng được xây dựng theo kiến trúc **agentic** – tức là cho phép mô hình "**suy nghĩ**", **lập kế hoạch**, và **tự ra quyết định**. Nhưng không phải ứng dụng nào cũng cần mức độ agentic giống nhau.

Vì vậy, khái niệm Cấp độ hành vi của tác nhân(**Degrees of Agentic Behavior**) ra đời – mô tả các **cấp độ từ đơn giản đến phức tạp** mà một hệ thống có thể thể hiện hành vi như một tác nhân (agent).

Agentic Behavior là gì?

"Agentic behavior" đề cập đến **khả năng hành động một cách có mục đích** của mô hình – không chỉ trả lời từng lệnh một, mà còn:

- Lập kế hoạch hành động
- Chọn công cụ phù hợp
- Gọi nhiều hành động kế tiếp
- Đánh giá lại kết quả
- Điều chỉnh chiến lược

Ví dụ, thay vì chỉ trả lời “không biết”, một agentic app có thể:

“Tôi không có thông tin trong cơ sở dữ liệu. Tôi sẽ tìm kiếm trên web, tóm tắt, rồi trình bày kết quả.”

5 Cấp độ Hành vi Agentic (Degrees of Agentic Behavior)

Chúng ta có thể chia hành vi agentic thành **năm cấp độ**, từ đơn giản (non-agentic) đến phức tạp (autonomous agents):

Level 0 - Không có hành vi tác nhân(Non-Agentic)

- ☐ Chỉ thực hiện một hành động theo prompt đã lập trình sẵn.
- ☐ Không có logic điều phối, không lập kế hoạch.

Ví dụ:

```
response = llm("Hãy tóm tắt đoạn văn sau...")
```

Dùng cho:

- Tóm tắt, dịch, phân loại
- Chatbot trả lời trực tiếp

Level 1 - Tăng cường bằng công cụ - Sử dụng công cụ một bước (Tool-Enhanced - Single-step Tool Use)

□ Ứng dụng có thể sử dụng **một công cụ bên ngoài**, nhưng người lập trình phải chỉ định rõ **khi nào dùng tool nào**.

Ví dụ:

```
if "search" in user_input:
    result = web_search_tool(query)
else:
    result = llm(user_input)
```

Dùng cho:

- Tìm kiếm trên web theo trigger
- Trích xuất dữ liệu từ cơ sở kiến thức

Level 2 - Các tác nhân phản ứng - Sử dụng công cụ động (Reactive Agents - Dynamic Tool Use)

- LLM **tự quyết định khi nào cần dùng công cụ nào**
- Các công cụ (tools) được mô tả bằng prompt hoặc API
- LLM chọn hành động tốt nhất dựa trên ngữ cảnh hiện tại

Ví dụ: LLM tự chọn `Calculator` để tính toán thay vì chỉ đoán kết quả.

Dùng LangChain Agents, OpenAI Function Calling, hoặc LangGraph Nodes.

Dùng cho:

- Trợ lý cá nhân AI có khả năng tìm kiếm, phân tích, tóm tắt
- Chatbot hiểu ngữ cảnh và chọn hành động phù hợp

Level 3 - Các tác nhân tự phản hồi hoặc thực hiện vòng lặp (Reflective / Iterative)

☐ Agent có khả năng **tự phản hồi (self-reflect)** và thực hiện **vòng lặp xử lý** đến khi đạt được mục tiêu

☐ Có thể lập kế hoạch, thực hiện hành động, kiểm tra lại, và sửa lỗi

Ví dụ:

```
if "search" in user_input:
    result = web_search_tool(query)
else:
    result = llm(user_input)
```

Dùng **Looping Graphs** trong LangGraph hoặc Agent Executors có khả năng hồi quy.

Dùng cho:

- Tác vụ đòi hỏi tư duy nhiều bước (multi-step reasoning)
- Tự kiểm tra và cải thiện kết quả

Level 4 - Các tác nhân chủ đa mục tiêu (Autonomous Multi-goal Agents)

☐ Agent có khả năng:

- Nhận mục tiêu tổng quát ("Viết báo cáo thị trường tháng này")
- **Tự chia nhỏ thành các sub-goals**
- **Tự lên kế hoạch**
- **Tự chạy, điều chỉnh, giám sát tiến trình**

Ví dụ: AutoGPT, BabyAGI, OpenDevin

Có thể chạy **nhiều vòng lặp**, tự lưu trạng thái và chuyển hướng kế hoạch khi có lỗi.

Dùng cho:

- Quản lý dự án
- Trợ lý lập trình hoặc viết báo cáo tự động
- Tự động hóa tác vụ phức tạp không xác định trước logic

So sánh nhanh

Cấp độ	Tên gọi	Mức độ tự chủ	Có lập kế hoạch	Có hành vi phản xạ	Dùng cho tác vụ phức tạp
0	Không có hành vi tác nhân (Non-agentic)	□	□	□	□
1	Tăng cường bằng công cụ (Tool-enhanced)	●	□	□	□
2	Các tác nhân phản ứng (Reactive agents)	□	□	□	□ (mức vừa)
3	Các tác nhân tự phản hồi hoặc thực hiện vòng lặp (Reflective/Iterative agents)	□□	□	□	□□
4	Các tác nhân tự chủ đa mục tiêu (Autonomous agents)	□□□	□□	□□	□□□

Tác giả: **Đỗ Ngọc Tú**
Công Ty Phần Mềm **VHTSoft**

Phiên bản #3
Được tạo 14 tháng 4 2025 03:01:16 bởi Đỗ Ngọc Tú
Được cập nhật 14 tháng 4 2025 03:51:13 bởi Đỗ Ngọc Tú