



## **Trabajo Práctico Final**

Procesamiento del Lenguaje Natural

### **Ejercicio 2 - Agentes**

Tecnicatura Universitaria en Inteligencia Artificial

Facultad de Ciencias Exactas, Ingeniería y Agrimensura

Universidad Nacional de Rosario

2024

**Integrante:**

Augusto Farias

## Parte 1- Investigación sistemas multiagentes

### Introducción

Como introducción podríamos decir que en la actualidad, la comunicación es crucial en nuestras vidas y con la tecnología en desarrollo, se vuelve más rápida y sencilla.

La comunicación humano-máquina es vital en investigación e industria, con un sistema diseñado para mejorarla. Están en constante crecimiento tecnologías de procesamiento del lenguaje natural para comprender y generar lenguaje en las conversaciones. Últimamente, la eficiencia de estos modelos se maximiza con un sistema multiagente que permite la interacción entre varios modelos desarrollados con NLP. Este sistema permite la comunicación bidireccional en caso de ser 2 agentes, o multidireccional en caso de más de 2 agentes.

### **Pero, ¿qué es un Agente?**

Un agente LLM es un **sistema** de inteligencia artificial que utiliza un **LLM** como su **motor** computacional central para exhibir capacidades más allá de la generación de texto.

Un ejemplo bidireccional entre dos agentes, es la siguiente:

Chat Agents	Dialogues
Chatbot1	Hello
Chatbot2	Hi There
Chatbot2	How are you?
Chatbot1	I'm Doing Well
Chatbot1	Are you intelligent?
Chatbot2	Yes, Of Corse
Chatbot2	Tell me more about yourself?
Chatbot1	Chabots are an Intelligent AI Programs
Chatbot1	What kind of techniques you have used to generate response?
Chatbot2	I Think so this is Necessary for your to Know about that
Chatbot2	Tell me more about yourself?
Chatbot1	You are repeating yourself
Chatbot1	Do you know about humans?
Chatbot2	I'm not sure if I understand what you are talking about
Chatbot2	What is a Chatbot
Chatbot1	A computer program designed to simulate conversation with Humans Users
Chatbot1	bye
Chatbot2	It was nice talking to you chatbot2, see you next time!

Acá vemos una simple conversación entre dos chatbots.

Pero estás tecnologías van más allá de una conversación entre agentes, pudiendo un modelo de lenguaje jugar un juego tan complejo o con tantas variables como es el Minecraft.

## Voyager

El primer agente de aprendizaje continuo incorporado potenciado por LLM, que aprovecha GPT-4 para explorar el mundo de manera constante, desarrollar habilidades cada vez más sofisticadas y hacer nuevos descubrimientos de manera consistente sin intervención humana. Voyager muestra un rendimiento superior en descubrir elementos novedosos, desbloquear el árbol tecnológico de Minecraft, atravesar terrenos diversos y aplicar su biblioteca de habilidades aprendidas a tareas no vistas en un mundo recién instanciado. Voyager sirve como punto de partida para desarrollar agentes generalistas potentes sin ajustar los parámetros del modelo.

Voyager es un agente que utiliza GPT-4, es decir un modelo de LLM closed-source, es decir que no podemos conocer su codificación para realizar cambios y adaptarlo a nuestra conveniencia.

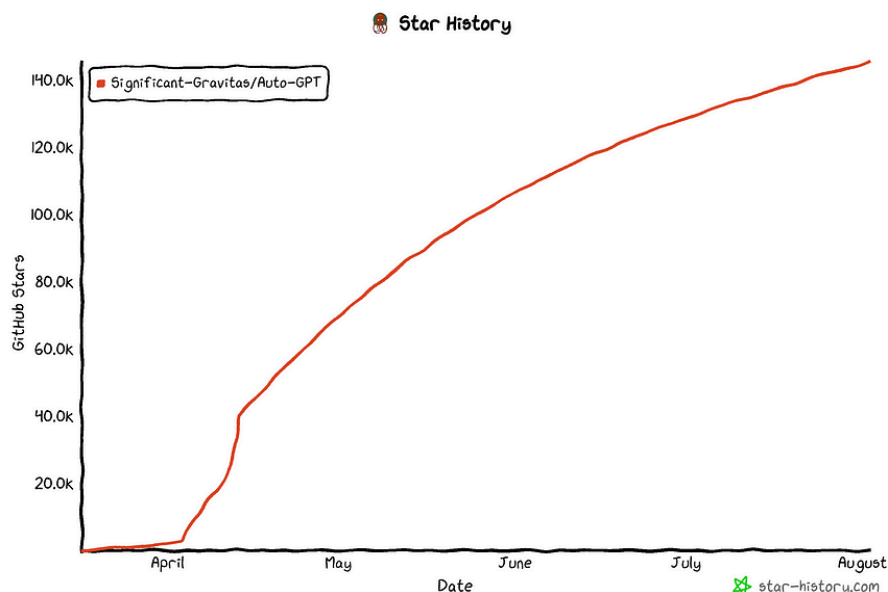
Existen otros modelos que son, también, closed source, como ReAct, Reflexion y AutoGPT.

Veamos el gráfico publicado por voyager en su página en cómo se desarrollaron los distintos modelos para realizar la misma tarea (crear una herramienta de madera, piedra, hierro y finalmente diamante)

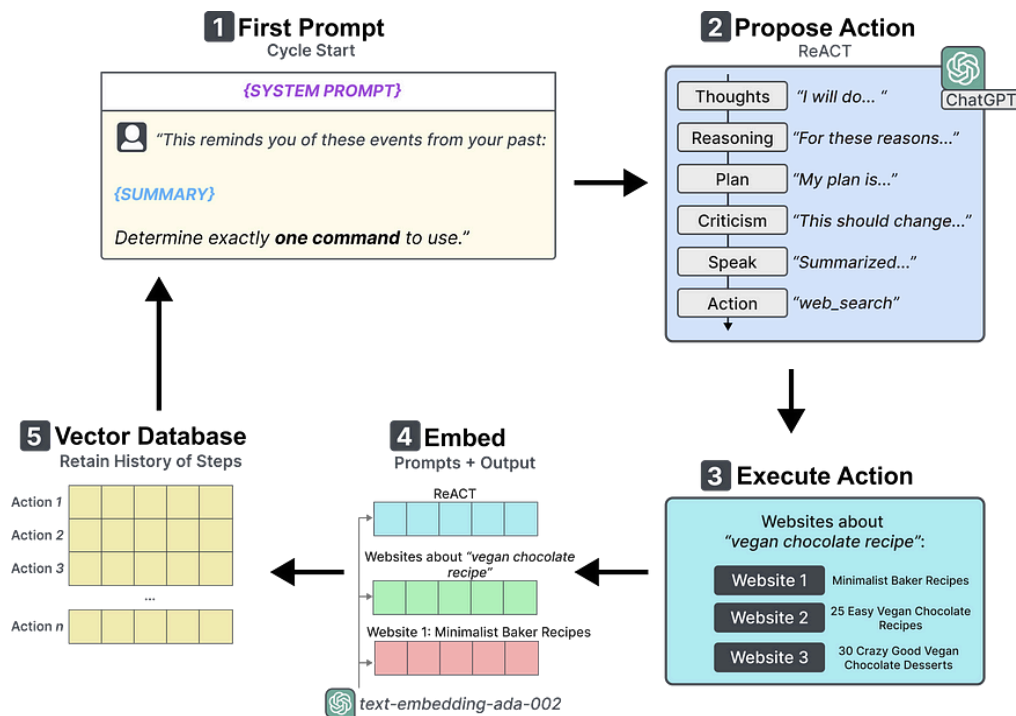
### Tech Tree Mastery

Method	Wooden Tool	Stone Tool	Iron Tool	Diamond Tool
ReAct	N/A (0/3)	N/A (0/3)	N/A (0/3)	N/A (0/3)
Reflexion	N/A (0/3)	N/A (0/3)	N/A (0/3)	N/A (0/3)
AutoGPT	92 ± 72 (3/3)	94 ± 72 (3/3)	135 ± 103 (3/3)	N/A (0/3)
VOYAGER w/o Skill Library	7 ± 2 (3/3)	9 ± 4 (3/3)	29 ± 11 (3/3)	N/A (0/3)
VOYAGER (Ours)	6 ± 2 (3/3)	11 ± 2 (3/3)	21 ± 7 (3/3)	102 (1/3)

AutoGPT es uno de los paquetes con mejor reputación y mejor rankeados en Github. En el siguiente gráfico podemos ver su crecimiento de forma logarítmica en cuanto a reputación.



Esta es la arquitectura del modelo:



Funciona de manera parecida al chatbot construido para el ejercicio 1, utilizando embeddings y un almacenamiento de la información en forma de vectores.

1. Inicializar el prompt con información resumida
2. GPT-4 propone una acción
3. Se ejecuta la acción Incrustar tanto en la entrada como la salida de este ciclo
4. Se guardan las incrustaciones en una base de datos de vectores

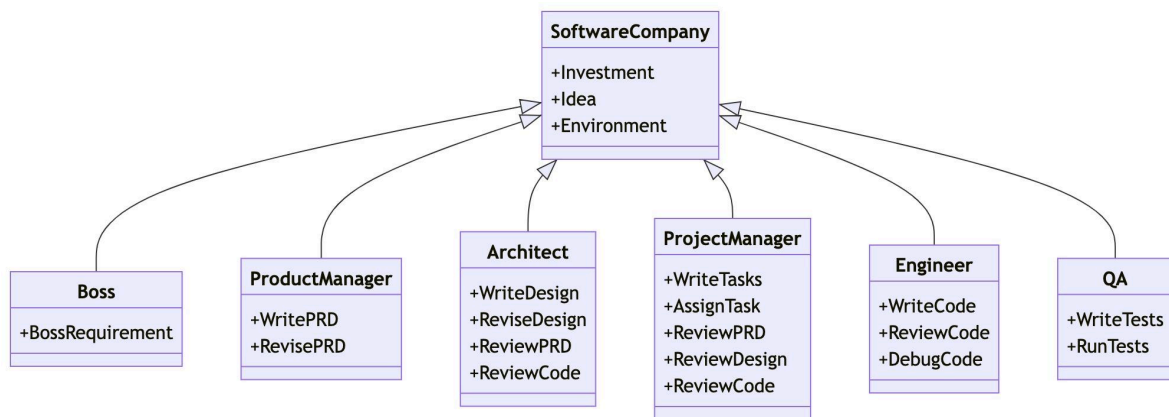
## MetaGPT

Ahora hablemos de MetaGPT: es open source y es uno de los mejores paquetes rankeados.

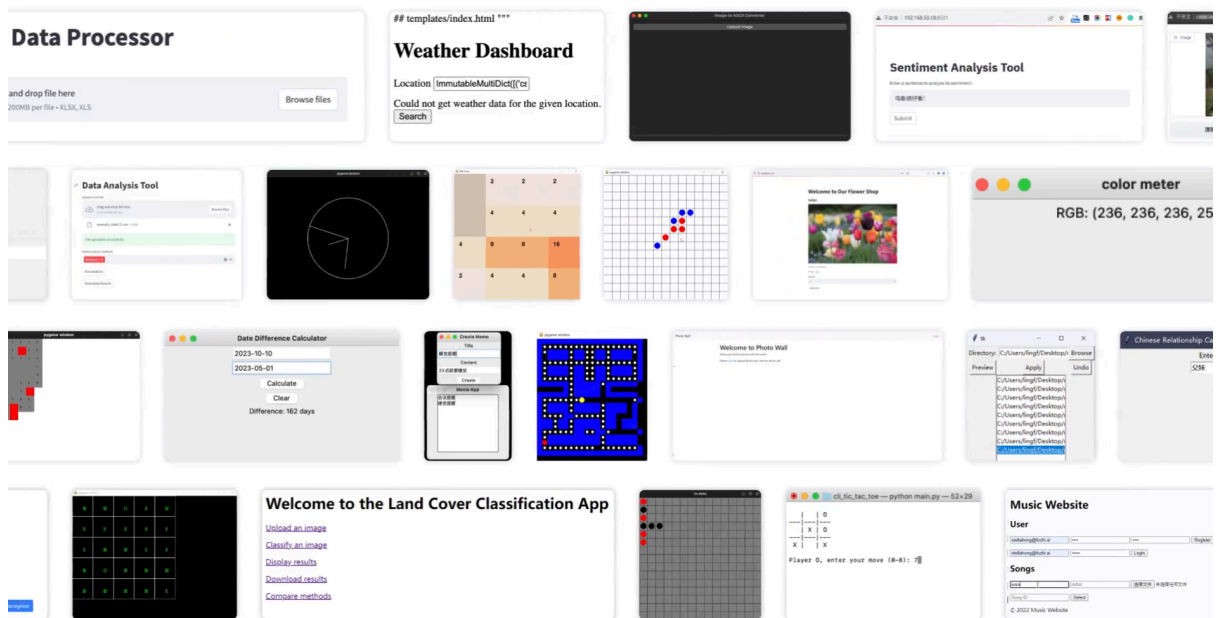
Con MetaGPT se construyó una aplicación que es Software Company Multi-Agent Schematic

1. MetaGPT recibe como entrada un requisito de una línea y produce historias de usuario, análisis competitivo, requisitos, estructuras de datos, APIs, documentos, etc. como salida.
2. Internamente, MetaGPT incluye a gerentes de producto, arquitectos, gerentes de proyecto e ingenieros. Proporciona todo el proceso de una empresa de software junto con SOPs (Procedimientos Operativos Estándar) cuidadosamente coordinados.

Podemos ver la estructura:



Podemos ver varias de las aplicaciones creadas por este sistema multi-agente:



En el caso del juego 2048 creado por este multi-agente el input al modelo fue:

metagpt "Create a 2048 game"

Es increíble el alcance que estos modelos pueden tener, desde tareas básicas hasta muy complejas. Que un modelo de lenguaje pueda realizar tareas en un juego como minecraft, me hace imaginar que a futuro se podría insertar estos modelos en un robot humanoide y generar un gran impacto en la industria de la tecnología.

## **Parte 2- Resolución de problema con un sistema multiagente**

### **Sistema Multiagente para una Clínica de Salud**

#### **Problemática**

La clínica de salud enfrenta desafíos para gestionar eficientemente los horarios de los pacientes, asignar recursos médicos de manera óptima y proporcionar un servicio personalizado y oportuno a los pacientes.

#### **Agentes Principales:**

1. **Agente de Recepción (AR):**
  - Función: Atender a los pacientes que llegan a la clínica, gestionar citas y registrar información básica del paciente.
  - Interacción:
    - Cliente: "Quisiera agendar una cita con el doctor Martínez."
    - AR: "Claro, ¿para qué fecha y hora le gustaría la cita?"
2. **Agente de Historia Clínica (AHC):**
  - Función: Gestionar y mantener actualizadas las historias clínicas de los pacientes.
  - Interacción:
    - Cliente: "He estado experimentando dolor en el pecho."
    - AHC: "¿Ha tenido algún episodio similar antes? ¿Cuándo comenzaron los síntomas?"
3. **Agente de Diagnóstico (AD):**
  - Función: Ayudar en el proceso de diagnóstico sugiriendo pruebas médicas y procedimientos.
  - Interacción:
    - Cliente: "Me siento muy cansado todo el tiempo."
    - AD: "Podría ser útil realizar un análisis de sangre completo y una evaluación de la función tiroidea. ¿Le gustaría programar esas pruebas?"
4. **Agente de Programación de Recursos (APR):**
  - Función: Gestionar la asignación de recursos como salas de consulta, equipos médicos y personal.
  - Interacción:
    - AR: "Necesitamos programar una ecografía para el paciente García."
    - APR: "Voy a verificar la disponibilidad de la sala de ecografías y al técnico correspondiente."
5. **Agente de Seguimiento (AS):**
  - Función: Realizar seguimiento post-consulta para asegurar el cumplimiento de tratamientos y proporcionar atención continua.
  - Interacción:
    - Cliente: "Me recetaron un nuevo medicamento, pero olvidé preguntar sobre los efectos secundarios."
    - AS: "Le enviaré información detallada sobre el medicamento y estaré disponible para responder cualquier pregunta adicional."

#### **Interacciones Entre Agentes:**

- El Agente de Recepción (AR) coordina con el Agente de Programación de Recursos (APR) para asignar citas y recursos según la disponibilidad.

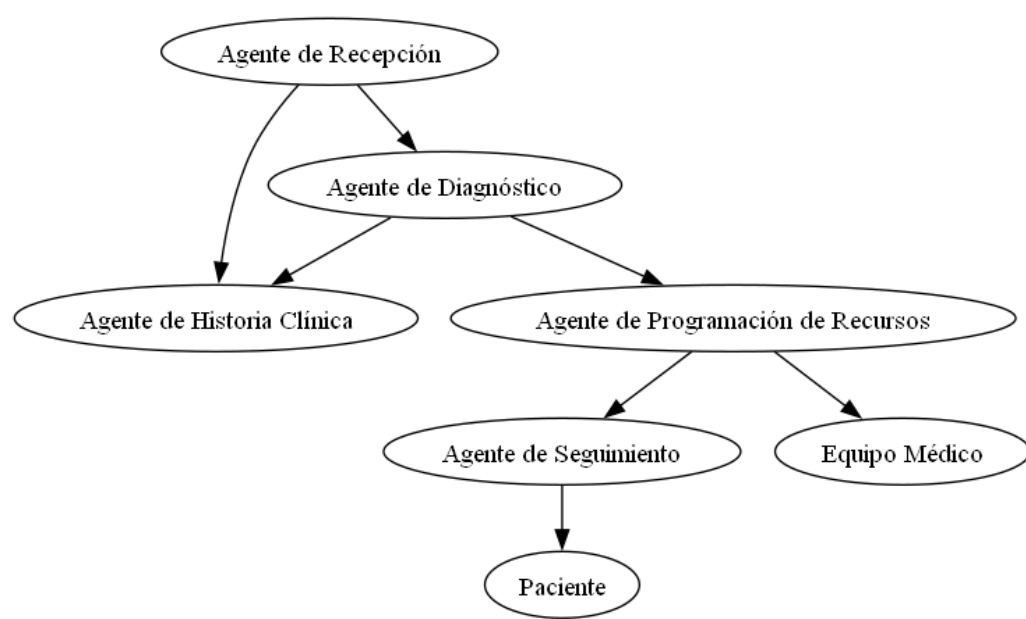
- El Agente de Diagnóstico (AD) consulta la Historia Clínica del paciente mantenida por el Agente de Historia Clínica (AHC) para obtener información relevante antes de sugerir pruebas médicas.
- El Agente de Seguimiento (AS) recibe actualizaciones de los diagnósticos y tratamientos realizados por el Agente de Diagnóstico (AD) para asegurar una atención continua.

### Conclusiones

El sistema multiagente propuesto para una clínica de salud aborda eficazmente la gestión de citas, la asignación de recursos médicos y el seguimiento de pacientes, mejorando así la eficiencia operativa y la satisfacción del paciente. La interacción entre los agentes garantiza un servicio personalizado y oportuno, mejorando la experiencia general del paciente en la clínica.

Este esquema proporciona una estructura básica para implementar un sistema multiagente en una clínica de salud, que puede adaptarse y ampliarse según las necesidades específicas de la institución médica.

### Esquema



En el repositorio de github [ESQUEMA](#) podemos observar un ejemplo de como se desarrollaría una consulta al sistema multiagente.

Para un paciente con los siguientes datos:

```

paciente = "Juan"

síntomas = ["dolor de cabeza", "fiebre"]

pruebas = ["análisis de sangre", "radiografía"]

tratamiento = "antibióticos"
  
```

El sistema multiagente devolvería lo siguiente:

```
Atendiendo al paciente: Juan
Actualizando historia clínica de Juan con los síntomas: ['dolor de cabeza', 'fiebre']
Sugiriendo pruebas para Juan basado en los síntomas: ['dolor de cabeza', 'fiebre']
Programando pruebas para Juan : ['análisis de sangre', 'radiografía']
Haciendo seguimiento a Juan sobre el tratamiento: antibióticos
```

Por cuestiones de tiempo y alcance de la materia el sistema multiagente se expresa como ejemplo y no está desarrollado en su totalidad. Pero tengo como objetivo poder desarrollarlo en un futuro ya que trabajo en el entorno de salud y es de mis metas poder ayudar a pacientes a tener mejor asistencia y tratamientos.



## **Referencias**

- <https://dl.acm.org/doi/abs/10.1145/3460268.3460275>
- <https://search.informit.org/doi/pdf/10.3316/informit.678065215092619>
- <https://voyager.minedojo.org/>
- <https://medium.com/@rohanbalkondekar/understanding-llm-agents-a-guide-to-creating-your-own-9390f02f3ff9>
- <https://towardsdatascience.com/decoding-auto-gpt-fae16ff1ee75>
- <https://medium.com/@rohanbalkondekar/understanding-llm-agents-a-guide-to-creating-your-own-9390f02f3ff9>
- <https://github.com/geekan/MetaGPT?tab=readme-ov-file>
- [https://github.com/augustofarias2/NLP/blob/main/Ejercicio\\_2/prueba\\_esquema.py](https://github.com/augustofarias2/NLP/blob/main/Ejercicio_2/prueba_esquema.py)