CEA Comité Español de Automática
XLVII JORNADAS DE AUTOMÁTICA CÓRDOBA 2026
UNIVERSIDADE DA CORUÑA

Jornadas de Automática

Sistema de control por lenguaje natural para planta de manufactura

Daniel Quesada-Lobo[a], Francisco Blanes-Noguera[b,*], Juan Carlos Brenes-Torres[a]

[a] *Instituto Tecnológico de Costa Rica, Cartago, Costa Rica.*
[b] *Instituto de Automática e Informática Industrial, Universitat Politècnica de València, Camino de Vera, s/n, 46022, Valencia, España.*

To cite this article: Quesada-Lobo, D, Blanes-Noguera, F, Brenes-Torres, J. 2026. Natural Language Control System for Automated Manufacturing Plant. Jornadas de Automática, 47. https://doi.org/10.17979/ja-cea.2026.47.13862

Resumen

Gracias a la aparición y el auge de los Grandes Modelos de Lenguaje (LLMs), se han generado nuevos sistemas capaces de interactuar con procesos industriales a través de lenguaje natural, complementando las interfaces tradicionales y aumentando la versatilidad de los sistemas de control. Este artículo desarrolla un sistema que utiliza LLMs por medio de un agente autónomo, el cual puede controlar las diversas partes de una planta de manufactura automatizada por voz. El agente elaborado permite la interacción con una base de datos SQL, y tiene acceso a las entradas y salidas de los Controladores Lógicos Programables (PLCs) por medio del protocolo industrial OPC-UA. El sistema fue implementado y validado en una planta de manufactura en el Instituto de Automática e Informática Industrial de la Universitat Politècnica de València, en donde alcanzó una tasa de éxito del 98.5 % para instrucciones sencillas y 81 % para las complejas, con un tiempo medio de respuesta inferior a 12.5 segundos.

Palabras clave: Grandes modelos de lenguaje, Control de plantas de manufactura, Sistemas de manufactura inteligentes, Interfaces inteligentes, Automatización y diseño centrada en humanos, Controlador lógico programable, Protocolos de comunicación industrial.

Natural Language Control System for Automated Manufacturing Plant

Abstract

Thanks to the appearance and rise in popularity of Large Language Models (LLMs), new systems capable of interaction with industrial processes through natural language have started to emerge, complementing traditional interfaces and making control systems more versatile. This article develops a system that uses LLMs as part of an autonomous agent, which is capable of controlling different parts of an automated manufacturing plant by voice. The agent is able to interact with a SQL database, and can also access Programmable Logic Controller (PLC) inputs and outputs via the industrial protocol OPC-UA. Finally, the system was implemented and tested in a physical manufacturing plant within Instituto de Automática e Informática Industrial of Universitat Politècnica de València, where a success rate of 98.5 % was achieved for simple instructions and 81 % for complex ones, with an average response time less than 12.5 seconds.

Keywords: Large Language Models, Manufacturing plant control, Intelligent manufacturing systems, Intelligent interfaces, Human-centered automation and design, Programmable logic controllers, Industrial communication protocols.

1. Introducción

En los últimos años se han dado grandes avances en sistemas de procesamiento de lenguaje natural gracias a la aparición de los Grandes Modelos de Lenguaje (LLMs), los cuales se han utilizado en un sinnúmero de áreas. Desde simples asistentes virtuales hasta sistemas que interactúan con equipos especializados, se ha demostrado la gran ventaja de trabajar utilizando lenguaje natural en lugar de los métodos de interacción tradicionales (Sarkar et al., 2025).

*Autor para correspondencia: pblanes@ai2.upv.es

Un área emergente en el uso de LLMs son los sistemas de control, especialmente para equipos automatizados que requieren una interacción dinámica. De esta manera, es posible complementar las pantallas HMI (Interfaz Humano-Máquina) con sistemas en lenguaje natural que buscan y organizan la información de sensores o bases de datos, y a su vez son capaces de activar distintas partes de la planta de producción. Estos sistemas autónomos son conocidos como agentes de inteligencia artificial (Sarkar et al., 2025; Xia et al., 2024).

Basado en esto, resulta muy útil el desarrollo de un sistema de control por lenguaje natural en plantas industriales, para garantizar la mayor facilidad de uso para sus usuarios. Además, estos agentes permiten la interacción con distintos tipos de APIs (Interfaz de Programación de Aplicaciones), los cuales son escogidos de manera autónoma según las necesidades y requerimientos del usuario. Esta capacidad de planificar y tomar decisiones de manera autónoma proporciona grandes ventajas con respecto a sistemas tradicionales (Xia et al., 2024).

Este artículo busca el desarrollo e implementación de un sistema basado en LLMs para el control por voz de una planta de manufactura en serie automatizada de Festo, ubicada en el laboratorio 4.0 del Instituto de Automática e Informática Industrial (Instituto ai2) de la Universitat Politècnica de València (UPV).

Las principales contribuciones de este trabajo son: (1) proponer una arquitectura capaz de controlar una planta de manufactura por medio de voz en lenguaje natural, (2) centralizar el proceso de control con un agente autónomo capaz de interactuar con las diversas partes de la planta, (3) facilitar la interacción con sistemas industriales diversos sin introducir vulnerabilidades de seguridad, y (4) demostrar las ventajas de los LLMs para el control de procesos industriales en un entorno aplicado.

El resto del documento se organiza según se describe a continuación. En la sección 2 se realiza una breve revisión del estado del arte en el control de sistemas industriales por medio de LLMs, la sección 3 presenta el sistema actual utilizado en la planta de manufactura, para luego detallar la solución desarrollada para el nuevo control del sistema. Se presentan las pruebas de validación realizadas y el análisis de los resultados del sistema en la sección 4, y finalmente la sección 5 detalla las principales conclusiones del trabajo y recomendaciones para trabajos futuros.

2. Estado del arte

Existe una gran cantidad de ejemplos de sistemas de control implementados con agentes autónomos, y las áreas de aplicación resultan realmente diversas, desde finanzas y medicina hasta la robótica, vehículos autónomos y manejo de datos. Hernández et al. (2025) diseñan un sistema con un LLM visual para el diagnóstico de glaucoma a partir de imágenes. Menéndez et al. (2025) desarrollan un sistema de agente autónomo basado en LLMs para la el control de un robot que interactúa con humanos. Además, Merino-Fidalgo et al. (2025) implementan un sistema para pasar de lenguaje natural a la creación de árboles de comportamiento para un robot social. Sin embargo, el control de sistemas industriales en una planta de producción por lenguaje natural es un área bastante reciente que se encuentra en constante evolución para desarrollar nuevas técnicas cada vez más eficientes (Sarkar et al., 2025; Zhao et al., 2026).

Xia et al. (2025) presentan un agente capaz de generar el control de una planta de producción, generando acciones en tiempo real según las demandas del usuario. Este utiliza un modelo de múltiples agentes para generar un control más complejo, el cual se subdivide en tareas más sencillas. De esta forma, se demuestra la usabilidad de los agentes en áreas de control para procesos de automatización en entornos reales.

Un aspecto fundamental para controlar una planta de producción recae en la interacción con los Controladores Lógicos Programables (PLCs) presentes. Para lograr esto se utilizan protocolos estandarizados como OPC-UA (Open Platform Communication Unified Architecture), que facilitan esta tarea y permiten obtener datos y escribir valores en las variables de cada PLC. Los agentes pueden interactuar de esta manera, actuando como clientes, y así controlar directamente cada PLC según sea necesario (Velesaca and Holgado-Terriza, 2025). Hofmanna et al. (2025) y Xia et al. (2025) generan sistemas de control por medio de agentes, los cuales controlan directamente los PLCs a través de OPC-UA. Esto permite pasar de instrucciones de alto nivel en lenguaje natural a comandos específicos para cada PLC.

Otra parte fundamental en una planta de producción recae en el almacenamiento de los datos. Para acceder a una base de datos es común utilizar Structured Query Language (SQL), el cual consiste en instrucciones llamadas queries, las cuales permiten obtener, editar, eliminar y en general tener control sobre información específica de las tablas de la base de datos. Sin embargo, este es un lenguaje técnico que no siempre es fácil de interpretar, por lo que utilizar LLMs para pasar de lenguaje natural a SQL es sumamente útil (Mohammadjafari et al., 2025).

La interacción de un agente con una base de datos SQL permite traducir las instrucciones en lenguaje natural a SQL de manera autónoma, lo cual permite que el agente pueda obtener de manera eficiente y en tiempo real los últimos datos de producción. Sin embargo, es importante tener en cuenta que si no se cuenta con la seguridad adecuada el agente podría borrar registros importantes en caso de error, lo cual puede ser perjudicial para el proceso (Mohammadjafari et al., 2025). En San Emeterio de la Parte et al. (2026) se implementa un agente para generar la interacción con una base de datos SQL en un entorno real, lo cual se logra de manera exitosa.

A pesar de su gran utilidad, es importante reconocer las limitaciones y principales desafíos que enfrentan los agentes en el control de procesos de manera autónoma. La principal fuente de incertidumbre recae en que los LLMs son propensos a alucinar generando información falsa, lo cual puede causar problemas en la planta desde errores sencillos hasta pérdidas importantes. Adicionalmente, otro problema predominante es la dificultad en comprender su toma de decisiones, ya que es complejo encontrar el punto de fallo en todo el flujo del agente, e interpretar sus respuestas puede llegar a ser difícil. Además, el tiempo de procesamiento necesario puede llegar a ser elevado, ya que para generar acciones de control más complejas es necesario el uso de LLMs con mayor tamaño, lo cual aumenta la demanda computacional y los tiempos para tomar decisiones (Sarkar et al., 2025; Xia et al., 2024).

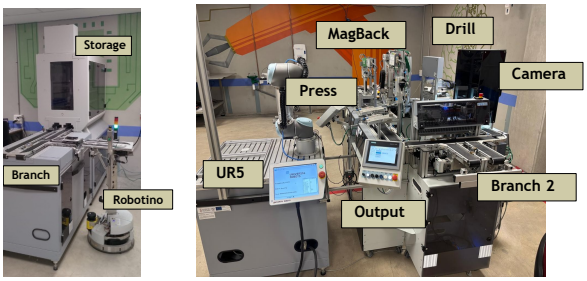
El sistema propuesto aborda estos desafíos mediante una arquitectura sencilla, la cual se enfoca en realizar las tareas más simples para generar el control de la planta. Además, se utilizó la herramienta LangSmith para aumentar la observabilidad del sistema al generar trazas para su respectivo análisis, permitiendo comprender todas las decisiones tomadas por el agente y reduciendo su incertidumbre. Esto permitió mejorar la trazabilidad del sistema durante su desarrollo y validación, pero no sustituye los mecanismos de control y seguridad en entornos industriales reales. De esta manera, el sistema resultó más rápido, confiable y ajustado a las necesidades, sin estar sobredimensionado, y permitiendo adiciones a futuro.

3. Materiales y métodos

3.1. Sistema actual

La célula de manufactura utilizada cuenta con ocho estaciones, las cuales se encuentran en serie y se comunican con un sistema centralizado MES (Sistema de Ejecución de Manufactura). Cada estación se encarga de una tarea distinta, desde liberar las piezas del almacén hasta realizar agujeros, colocar y presionar la cubierta, y finalmente despacharlas hacia un cliente. Cada estación cuenta con un PLC que envía datos por medio de un servidor OPC-UA, los datos de la orden completa son guardados en una base de datos en Microsoft SQL Server, y a su vez todo el proceso se monitorea en un sistema Ignition SCADA.

La planta está dividida en dos partes, las cuales son conectadas por medio de un robot móvil Robotino de Festo que transporta las piezas desde el almacén hasta las demás estaciones y de regreso. Además, para el despacho de las piezas se cuenta con un brazo robótico UR5 de Universal Robots que traslada las piezas terminadas hacia cajas listas para su entrega. La distribución de la planta se observa en la Figura 1.



(a) Isla principal (b) Isla remota

Figura 1: Distribución de la célula de manufactura. (a) La isla principal que incluye el almacén de la planta. (b) La isla remota que genera el ensamblaje.

La planta cuenta con dos bases de datos que están sincronizadas, debido a que el sistema MES4 utilizado solo permite acceso de forma local, mientras que Ignition se ubica en un servidor remoto. Además, se cuenta con una interfaz en un ordenador desarrollada en Ignition Vision, la cual permite controlar la planta de manufactura mediante acciones como la creación, consulta, activación o eliminación de órdenes de producción, y también permite revisar y editar los contenidos del almacén. Un diagrama de capas que detalla la comunicación de este sistema se muestra en la Figura 2.

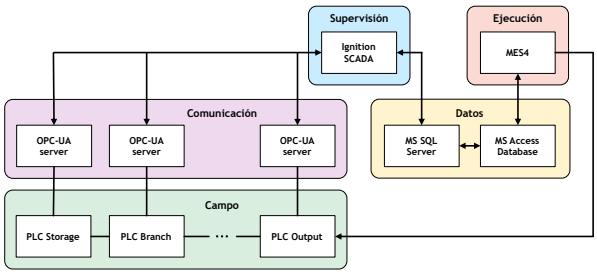


Figura 2: Arquitectura de la comunicación en el sistema actual.

Para generar una pieza nueva en esta interfaz es necesario seleccionar especificaciones de la orden a generar como el número de orden, tipo de pieza y el cliente deseado, las cuales muchas veces no resultan intuitivas. Además, cada acción debe ser realizada de manera individual, por lo que crear tres órdenes distintas implica acceder a esa ventana tres veces separadas y generar cada pieza una a la vez, lo cual puede tardar un tiempo considerable. Esto resulta evidente al editar los contenidos de las piezas en el almacén, ya que al haber 32 espacios posibles puede llegar a tardar más de cinco minutos en total, lo cual es un tiempo sin producción bastante elevado.

3.2. Arquitectura general de la solución

La solución propuesta se basa en un agente autónomo, el cual actúa como la unidad central del sistema y es capaz de llamar a herramientas y funciones que acceden a distintos servicios como la base de datos SQL y el servidor OPC-UA de cada PLC. Todo esto funciona gracias a la conexión con un LLM (GPT-4.1 mini) que interpreta el lenguaje natural que proviene del usuario, determina la acción a realizar, y luego redacta una respuesta acorde a lo solicitado, todo de manera autónoma y con la menor intervención posible. Este modelo de lenguaje se seleccionó tras una revisión de LLMs disponibles en la nube y algunas pruebas de funcionamiento, en donde se buscó un modelo sencillo que tuviera la capacidad de toma de decisiones y llamada a funciones externas necesarias para el control. Además, se priorizó tener un tiempo de respuesta corto, ya que esto es indispensable para una interacción en tiempo real, y a su vez contar con un bajo costo y un tamaño de contexto suficiente.

Aunado al agente, dos modelos distintos se encargan de pasar la voz del usuario a texto y las respuestas del agente en texto a voz. Para la primera acción se utilizó el modelo Whisper de OpenAI, mientras que para la salida de voz se usó Google Text-to-Speech. Esto permite que el usuario pueda solamente hablar y escuchar, sin necesidad de tener un teclado o mirar la pantalla, con el fin de poder trabajar en cualquier lugar de la planta de la manera más eficiente. Estos modelos se escogieron por su simplicidad de implementación y de uso, tomando en cuenta mantener el tiempo de respuesta al mínimo.

De manera adicional, la salida del modelo de voz a texto (STT o Speech-to-Text) pasa por una etapa con un LLM adicional (GPT-4.1 nano) que se encarga de corregir errores del STT como gramática y redacción, y eliminar errores ya sea por ruido de fondo, voces distintas o demás aspectos que dificulten su comprensión. Para su configuración se le brindó en su contexto una lista de las instrucciones y términos más

comunes para ayudar a interpretar los mensajes que recibe, lo cual corresponde a un método de few-shot prompting.

Esto se desarrolló tras observar errores usuales en interpretar los nombres de las estaciones o fallos al eliminar partes importantes del mensaje. Al introducir estas instrucciones base el modelo fue capaz de interpretar los mensajes de mejor manera, y se redujeron en gran medida la cantidad de fallos generados. Esto resulta crítico, ya que si el mensaje tiene errores o está incompleto el agente no sería capaz de interpretar la instrucción real y su resultado sería impredecible.

En la Figura 3 se puede observar la arquitectura general del sistema desarrollado, que contiene todas las partes descritas anteriormente.

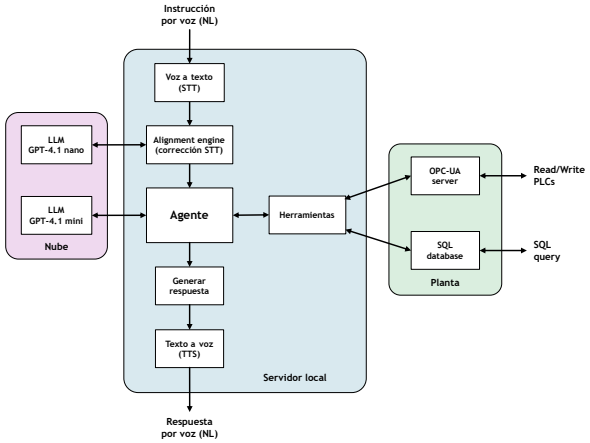


Figura 3: Arquitectura general del sistema de control. El bloque en celeste representa las partes implementadas en el servidor local, que se conectan a LLMs en la nube y a la planta de manufactura.

3.2.1. Interacción con la base de datos SQL

En lugar de permitir que el agente genere sus propias queries en SQL para consultar la base de datos, se definió una serie de queries preestablecidas las cuales el agente puede determinar y escoger la más útil según la instrucción del usuario. Esto se realizó para mejorar la confiabilidad del agente, eliminar la posibilidad de instrucciones inválidas y prevenir riesgos de seguridad en donde el agente podía eliminar o editar datos importantes no autorizados. Además, antes de ejecutar cualquier query sobre la base de datos se validan los parámetros utilizados para cada función, lo cual busca prevenir acciones destructivas como la inyección SQL.

El nuevo sistema es capaz de replicar lo que era posible en la interfaz anterior, además de tener funciones adicionales gracias a la adaptabilidad del agente. Se generaron un total de 12 funciones con queries a la base de datos, divididas en 5 para la lectura de información, 5 para editar diversos datos, y 2 para tareas varias necesarias durante el flujo de la interacción.

El agente puede decidir las funciones que debe llamar para resolver el problema de manera directa según los nombres y la descripción de cada función. En la Tabla 1 se muestra una comparación entre las acciones posibles en el nuevo sistema y en la interfaz anterior.

3.2.2. Interacción por OPC-UA

El agente puede conectarse directamente con el servidor OPC-UA de cada PLC de manera independiente a través de una librería en Python llamada *asyncua*, la cual permite leer y escribir datos en los tags de cada estación de manera asíncrona. El sistema establece la conexión con el servidor OPC-UA del PLC que requiere leer o escribir y se verifica que la conexión sea correcta, o en caso de fallos se indica un mensaje de error al usuario para que pueda revisar la estación.

La primera acción del agente es generar una consulta de los tags disponibles para el PLC de interés, tanto de las entradas como las salidas. Debido a que los nombres de estos no son intuitivos, se generó un diccionario de tags que el agente puede consultar para interpretar y decidir las entradas y salidas que debe utilizar. Se excluyeron del diccionario los tags de sistema y aquellos que pueden llevar a riesgos de seguridad, especialmente para acciones de escritura. Este se implementó directamente en Python como una función auxiliar, de manera que el agente recibe directamente los nombres interpretados al utilizar la función de consultar tags.

Las otras funciones que puede realizar el agente son leer y escribir a las variables asociadas a cada PLC, basado en los nombres proporcionados por el diccionario de tags. Esto le permite activar o desactivar las bandas, stoppers, barreras y demás aspectos básicos de la planta, pero por seguridad el agente no puede activar salidas que resulten peligrosas, como las garras en movimiento o el taladro. Además, el agente puede monitorear la planta gracias a la lectura de los tags para comprender las partes que se encuentran activas.

3.2.3. Otras funciones

Adicionalmente, se agregaron otras funciones que resultan útiles durante la utilización de la célula de manufactura. En primer lugar, el agente es capaz de activar o detener cada estación, lo cual permite poner todo en funcionamiento a partir de un único comando. Además de esto, cada estación permite trabajar en modo automático o manual, por lo que el agente puede cambiar entre estos según el tipo de acción a realizar.

Finalmente, se incluyó una función que identifica los sensores de presencia que hay en cada estación y lee su valor a través de OPC-UA. Esto permite obtener una lista de todas las piezas que se encuentran presentes en la planta, detallando la estación en la que están y su ubicación dentro de esta. De esta manera, es posible determinar si hay sectores en los que las piezas tarden mucho tiempo, lo cual permite alertar al usuario para revisar si existe un problema con la planta.

4. Resultados y análisis

Para evaluar el desempeño del sistema desarrollado se generó una lista de instrucciones a las cuales se sometió el agente. En total se utilizaron 43 instrucciones distintas, se realizaron tres repeticiones por cada instrucción, y a su vez fueron clasificadas en dos grupos: sencillas y complejas. Esto llevó a obtener datos para 66 pruebas con instrucciones sencillas y 63 pruebas con instrucciones complejas, para una muestra total de 129 datos. Todas estas pruebas fueron realizadas por un único evaluador, y luego se realizaron pruebas adicionales con 5 usuarios distintos para evaluar los resultados con otras voces y acentos.

Las instrucciones sencillas consisten de acciones que pueden solucionarse en una única interacción con el usuario, y

Tabla 1: Comparación entre las acciones posibles con el sistema anterior en Ignition y el nuevo sistema con agente autónomo.

Acción	Interfaz en Ignition	Sistema de agente autónomo
Crear órdenes inmediatas	Una a la vez	Múltiples simultáneamente
Cancelar o eliminar órdenes	Una a la vez	Múltiples simultáneamente
Editar piezas del almacén	Una a la vez	Múltiples simultáneamente
Avisar si no hay piezas disponibles en almacén	No	Sí
Calcular duraciones	No	Sí
Instrucciones condicionales	No	Sí
Consultar piezas terminadas	Todas	Según filtros (por fechas, clientes, etc.)
Crear órdenes futuras	Con hora fija	Con hora fija o relativa (dentro de cierto tiempo)
Revisar ubicación de las piezas	Estaciones con piezas	Partes de estaciones con piezas (mayor detalle)

que además no requieren más de 3 llamadas a funciones ni de un razonamiento complejo por parte del agente. Muchas de estas corresponden a aspectos que se podían realizar con la interfaz anterior. Por otro lado, las instrucciones complejas pueden requerir múltiples llamadas a funciones en secuencia, un análisis del contexto para tomar decisiones, o incluso consultas adicionales al usuario para aclarar detalles. La mayoría de estas instrucciones requieren de datos de distintas funciones, lo cual evalúa realmente la capacidad del modelo de escoger las herramientas adecuadas y no alucinar valores incorrectos.

4.1. Tasa de aciertos

Se definieron fallos posibles que el agente podía generar, como alucinar información falsa, llamar funciones incorrectas, indicar información no solicitada o errónea, problemas de comunicación, entre otros. Se anotó en cada prueba si el agente realizaba la instrucción de manera exitosa o no, y los resultados para ambos tipos de instrucciones se observan en la Tabla 2.

Tabla 2: Tasa de aciertos para instrucciones simples y complejas.

Instrucciones	Aciertos	Fallos	Tasa de éxito (%)
Simples	65	1	98.48
Complejas	51	12	80.95

El agente tuvo una tasa de éxito muy alta para las instrucciones sencillas, en donde la única instrucción con fallo resultó por llamar una función incorrecta. Sin embargo, cuando aumenta la complejidad resulta mucho más propenso a fallar, en donde se obtuvo 1 error por llamar funciones incorrectas, 8 por interpretar mal la salida de una función, y 3 por alucinar información falsa.

Para analizar esto con más detalle, se calculó la tasa de aciertos según los distintos tipos de instrucciones definidos, lo cual se puede observar en la Figura 4.

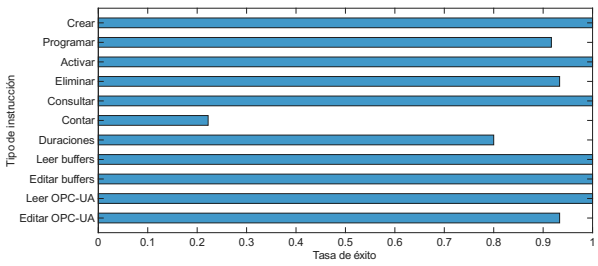


Figura 4: Tasa de aciertos según el tipo de instrucción.

Las instrucciones más falladas por el agente correspondieron a aquellas que requieren contar la cantidad de datos de cierto tipo, como por ejemplo "¿Cuántas órdenes terminaron en mayo?". El agente fue capaz de identificar las órdenes de manera correcta, pero se equivocó al contar el total de órdenes con esa característica. Este fallo se debe a una limitación intrínseca de los LLMs, en donde los resultados se dividen en tokens para su entendimiento, pero estos no suelen coincidir con los elementos de las listas. Esto genera que pueden contar múltiples veces un elemento si este se compone de múltiples tokens, o también agrupar varias órdenes en una sola al no estar clara la división entre órdenes para el sistema (Ball et al., 2024; Fu et al., 2023).

4.2. Tiempo de procesamiento

Otra métrica importante es el tiempo total de procesamiento de cada instrucción, ya que esto permite estudiar las instrucciones más lentas que limitan para una interacción en tiempo real. En la Figura 5 se muestra un diagrama de cajas y bigotes para ambos tipos de instrucciones.

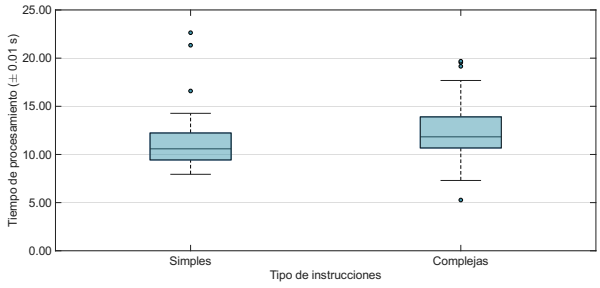


Figura 5: Tiempo de procesamiento transcurrido para instrucciones simples y para instrucciones complejas.

La mediana del tiempo de procesamiento para instrucciones simples resultó de 10.59 s, mientras que fue de 11.83 s para las complejas. Esto indica que el agente realmente requiere de un mayor tiempo cuando aumenta la complejidad de las instrucciones, lo cual es un resultado esperable. Sin embargo, se observan algunos puntos atípicos, los cuales corresponden a problemas de conexión que sucedieron durante la ejecución de la prueba, así como por errores en llamar alguna función, por lo que el agente lo volvió a intentar y tardó casi el doble del tiempo esperado.

4.3. Tiempo de inicialización

El tiempo necesario para la puesta en marcha del sistema fue medido para cada prueba realizada. Esto no depende del tipo de instrucción, por lo que se agruparon todos los resultados obtenidos. El sistema anterior tardaba aproximadamente 30 segundos en inicializar, mientras que con este nuevo sistema los tiempos no superan los 8 segundos. Esto es una gran mejora, lo cual se debe gracias a ser un sistema centralizado que permite cargar los datos conforme se requieren, a diferencia de antes en donde toda la información se cargaba y siempre estaba disponible para el usuario, lo cual muchas veces terminaba siendo innecesario.

Finalmente, en la Tabla 3 se muestra el promedio, desviación estándar y el percentil 95 obtenidos para la prueba del tiempo de inicialización, además del tiempo de procesamiento para instrucciones simples y complejas.

Tabla 3: Promedio, desviación estándar y percentil 95 para las pruebas realizadas.

Tiempo	Promedio (s)	Desv. est. (s)	P95 (s)
Inicialización	6.332	0.4413	7.216
Procesamiento simples	11.21	2.590	14.14
Procesamiento complejas	12.33	2.874	17.64

La comparación de los resultados obtenidos con los tiempos con la interfaz en Ignition resulta interesante. El sistema anterior tardaba entre 5 y 10 segundos en generar cada acción, lo cual resulta más rápido que el nuevo sistema. Sin embargo, tomando en cuenta que todas las acciones se debían realizar una tras otra, cuando se evalúan instrucciones más complejas que requieren de múltiples acciones, el sistema pasado resulta sumamente ineficiente, tardando más de 30 segundos en algunos casos. Esto demuestra la ventaja del sistema desarrollado.

5. Conclusiones

En el presente trabajo se desarrolló un sistema que permite generar el control de una planta de manufactura a partir de instrucciones en lenguaje natural por voz. Se demostró que el sistema es capaz de ejecutar instrucciones de manera acertada especialmente para comandos sencillos, y a su vez puede actuar de manera rápida para obtener una interacción en tiempo real. Esto ha mejorado considerablemente la facilidad de uso de la planta de manufactura al lograr trabajar de manera más rápida y personal, pudiendo adaptarse a cualquier solicitud del usuario.

La interacción por lenguaje natural con equipo industrial puede ser un gran complemento al uso de sistemas HMI tradicionales, gracias a sus ventajas en facilidad y adaptabilidad de uso. Sin embargo, el poco acceso a modelos de lenguaje locales, la falta de estandarización que dificulta su implementación, y los problemas de seguridad que pueden surgir son aún barreras que existen para su adopción total. Esto indica que para llevar este tipo de sistemas a entornos reales es necesario desarrollar capas intermedias que validen y supervisen las acciones generadas por el agente, reduciendo los riesgos de seguridad y mejorando la confiabilidad del control. De esta manera, puede ser posible desarrollar enfoques híbridos que permitan interactuar de ambas formas con la planta de manufactura.

Como líneas futuras de trabajo se planea la creación de una capa MCP (Model Context Protocol) para que el agente interactúe con esta en lugar de directamente con los PLCs o la base de datos, y así utilizar un protocolo estándar que sea escalable y que aumente la seguridad en las acciones realizadas. Además, se implementará el uso de LLMs locales para garantizar la privacidad de los datos, lo cual resulta crítico para toda empresa industrial. Finalmente, se plantea generar un sistema multi-agente que divida cada tarea de control y monitoreo en subagentes distintos, y un agente central que los gobierne y los llame según sea necesario. Esto permitiría no sobrecargar el agente actual con una gran cantidad de tareas, manteniendo una alta exactitud en seleccionar las herramientas adecuadas, y así garantizar una alta escalabilidad y confiabilidad en el control de la planta.

Referencias

Ball, T., Chen, S., Herley, C., 2024. Can we count on llms? the fixed-effect fallacy and claims of gpt-4 capabilities. Microsoft Research. DOI: 10.48550/arXiv.2409.07638

Fu, T., Ferrando, R., Conde, J., Arriaga, C., Reviriego, P., 2023. Why do large language models (llms) struggle to count letters? Association for Computing Machinery. DOI: 10.48550/arXiv.2412.18626

Hernández, J., Barrios, E. J., Diaz-Aleman, S. A. T., 2025. Aplicación de grandes modelos de lenguaje en diagnóstico de glaucoma. Jornadas de Automática, 46. DOI: 10.17979/ja-cea.2025.46.12085

Hofmanna, B., Piechuleka, N., Kreitleina, S., Frankea, J., Bründl, P., 2025. Beyond touch-based human-machine interface: Control your machines in natural language by utilizing large language models and opc ua. Manufacturing Review. DOI: 10.48550/arXiv.2510.11300

Menéndez, E., Garcia-Haro, J. M., Martínez, S., Balaguer, C., 2025. Integración modular de herramientas gestionadas por llms en el robot tiago++. Jornadas de Automática, 46. DOI: 10.17979/ja-cea.2025.46.12229

Merino-Fidalgo, S., Zalama, E., Gómez-García-Bermejo, J., Duque-Domingo, J., 2025. Generación y ajuste dinámico de behavior trees mediante llms para el control de un robot social. Jornadas de Automática, 46. DOI: 10.17979/ja-cea.2025.46.12071

Mohammadjafari, A., Maida, A. S., Gottumukkala, R., 2025. From natural language to sql: Review of llm-based text-to-sql systems. Cornell University. DOI: 10.48550/arXiv.2410.01066

San Emeterio de la Parte, M., Wang, Y., Martínez-Ortega, J. F., Martínez, N. L., 2026. A secure and efficient llm-based system for natural language query-to-sql translation in iot data management. IEEE Access. DOI: 10.1109/ACCESS.2026.3665980

Sarkar, S., Ghorbanpour, S., Gutiérrez, R. L., Guillén-Pérez, A., Gundecha, V., Babu, A. R., Naug, A., 2025. Review of llm based control. Hewlett Packard Labs at Hewlett Packard Enterprise. DOI: 10.13140/RG.2.2.28595.54567

Velesaca, H. O., Holgado-Terriza, J. A., 2025. Opc-ua in artificial intelligence: a systematic review of the integration of data mining and nlp in industrial processes. Manufacturing Review. DOI: 10.1051/mfreview/2025003

Xia, Y., Jazdi, N., Weyrich, M., 2024. Applying large language models for intelligent industrial automation. Universidad de Stuttgart. DOI: 10.17560/atp.v66i6-7.2739

Xia, Y., Jazdi, N., Weyrich, M., 2025. Control industrial automation system with large language model agents. IEEE International Conference on Emerging Technologies and Factory Automation. DOI: 10.48550/arXiv.2409.18009

Zhao, L., Liu, S., Xin, T., Tan, J., Wang, X., Li, Y., Bian, Z., Chen, Y., Kong, F., Bian, J., Qian, C., Zhang, Z., 2026. Ai agent in healthcare: applications, evaluations, and future directions. npj Artificial Intelligence. DOI: 10.1038/s44387-026-00076-4