



# Poster y Paper

## **Equipo de Proyecto:**

Lucas Martin Chas Diaz

Gino Tombesi

Juan Ignacio Rottino

Valentino Gabriel Rocca

Juan Ignacio Varengo Bossi

Angel Nahuel Villegas

# OSIRIS

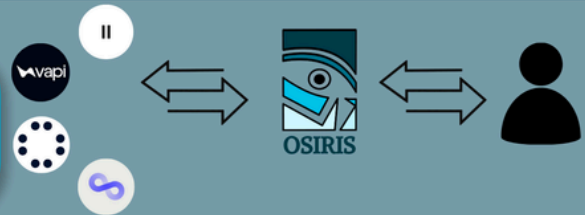


## ¿Qué es?

Plataforma web multi-tenant que centraliza la planificación, ejecución y análisis de campañas de voz operadas por proveedores de IA, incorporando un agente propio que sugiere estrategias de distribución de llamadas sujetas a aprobación humana

## ¿Qué ofrece?

Interfaz única para administrar campañas, agentes y contactos de múltiples proveedores de voz.



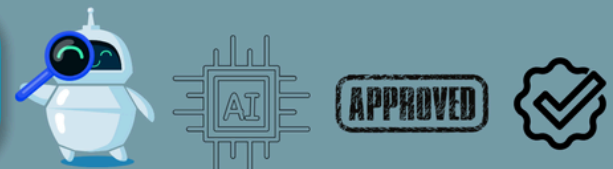
Dashboard en tiempo real con métricas de negocio, más allá de las métricas técnicas del proveedor.

Registro histórico con transcripciones, grabaciones y análisis conversacional automatizado.

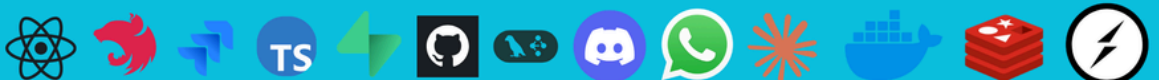


Distribución de llamadas basada en datos históricos, con reglas de reintento y niveles de concurrencia.

Recomendaciones del Strategy Agent siempre sujetas a aprobación humana antes de ejecutarse.



### Stack Tecnológico



**Lucas Martin Chas Diaz**  
luquitaschasdiaz3@gmail.com  
**Valentino Gabriel Rocca**  
roccavalentino5@gmail.com

**Equipo**  
**Gino Tombesi**  
ginotombesi@gmail.com  
**Angel Nahuel Villegas**  
nahuelvillegas4@gmail.com

**Juan Ignacio Rottino**  
juan.rottino@gmail.com  
**Juan Ignacio Varengo**  
juanivarengo@gmail.com



# OSIRIS

**Chas Díaz, Lucas Martín – Tombesi, Gino Paolo – Rottino, Juan Ignacio – Rocca, Valentino Gabriel – Varengo Bossi, Juan Ignacio – Villegas, Ángel Nahuel**

*Universidad Tecnológica Nacional, Facultad Regional Córdoba*

## Abstract

*Las empresas que operan agentes de voz con Inteligencia Artificial, como Retell AI, VAPI o Bland AI, enfrentaron un ecosistema operativo fragmentado, con gestión de campañas distribuida entre múltiples interfaces y ausencia de métricas unificadas orientadas al negocio. Para abordar esta problemática se desarrolló OSIRIS, una plataforma web multi-tenant orientada a centralizar la administración, ejecución y análisis de campañas de voz basadas en IA, sin reemplazar a los proveedores existentes. El desarrollo se llevó a cabo bajo metodología Scrumban, con una arquitectura de tipo Ports & Adapters que desacopló la lógica de negocio de los proveedores de voz específicos. Se implementó un backend en NestJS dividido en una API HTTP y un worker asíncrono basado en BullMQ, un frontend en React con Vite, y una capa de datos multi-tenant en Supabase (PostgreSQL) con seguridad a nivel de fila. El resultado principal fue un Strategy Agent, construido con LangGraph, capaz de proponer estrategias de distribución de llamadas —francas horarias, concurrencia y reglas de reintento— a partir de datos históricos y de contacto, sujetas a aprobación humana antes de su ejecución. La plataforma permitió la carga masiva de contactos, la ejecución orquestada de campañas, el registro histórico de llamadas con transcripciones y grabaciones, y la visualización de métricas operativas en un dashboard en tiempo real.*

## Palabras Clave

Producto, planificación de voz, inteligencia artificial, agentes conversacionales, multi-tenant, arquitectura de adaptadores, contact center, LangGraph, automatización de campañas.

## Introducción

Las empresas que gestionan campañas de atención telefónica automatizada mediante Inteligencia Artificial recurren habitualmente a proveedores especializados como Retell AI, VAPI o Bland AI. Sin embargo, la operación de estas campañas suele encontrarse fragmentada entre múltiples interfaces de proveedor, sin una capa que centralice la configuración, la ejecución y, sobre todo, las métricas de negocio —tasas de conversión, costos reales, resultados por segmento— por

encima de las métricas puramente técnicas que exponen dichos proveedores.

OSIRIS surge como respuesta a esta problemática: una plataforma web multi-tenant que se sitúa como capa de planificación por encima de los proveedores de voz existentes, sin intención de reemplazarlos ni de desarrollar un motor conversacional propio. Su propósito es dotar a equipos de operaciones de contact centers y áreas comerciales (banca, salud, retail) de una herramienta única para administrar campañas, aplicar estrategias de distribución de llamadas y obtener visibilidad unificada del rendimiento, incorporando además un agente de inteligencia artificial propio orientado a sugerir mejoras en la ejecución de las campañas.

## Elementos del Trabajo y Metodología

El desarrollo del proyecto se organizó bajo un enfoque híbrido basado en Scrumban, combinando la planificación iterativa de Scrum —sprints de dos semanas con reuniones de seguimiento y retrospectivas— con la visualización de flujo continuo propia de Kanban. La gestión de tareas se realizó en Jira, la documentación se centralizó en Google Drive, y el versionado de código se gestionó mediante Git y GitHub, registrándose a lo largo del proyecto más de 150 commits organizados en ramas por funcionalidad e integrados mediante pull requests revisados por pares.

La solución se estructuró como un monorepo (pnpm workspaces) compuesto por tres aplicaciones y un paquete de tipos compartidos. El backend se dividió en dos servicios independientes desarrollados en NestJS y TypeScript: una API HTTP (api-v2), responsable de la autenticación, la gestión de proyectos, agentes, campañas y

contactos, y un worker asíncrono, encargado del procesamiento de colas mediante BullMQ y Redis para tareas de mayor duración, como la generación de estrategias, la ejecución de campañas y el procesamiento de eventos de llamada. La comunicación en tiempo real hacia el frontend se implementó mediante Socket.IO.

El frontend se desarrolló en React 18 con Vite, utilizando componentes de Tremor y Recharts para la visualización de dashboards. La autenticación y la persistencia de datos se resolvieron mediante Supabase, aprovechando PostgreSQL con seguridad a nivel de fila (Row Level Security) para garantizar el aislamiento de datos entre organizaciones (workspaces) en el esquema multi-tenant.

Para evitar el acoplamiento con un proveedor de voz específico, se aplicó una arquitectura de tipo Ports & Adapters: toda integración externa debe implementar una interfaz común (IProviderAdapter) que valida la firma de los webhooks mediante HMAC-SHA256 y normaliza los eventos de llamada a un formato interno (NormalizedCallEvent), de modo que el resto del dominio —métricas, dashboard, persistencia— permanece independiente del proveedor utilizado. Sobre esta base se implementó el primer adaptador, correspondiente a Retell AI, encontrándose en desarrollo la incorporación de un segundo adaptador para VAPI.

El componente de inteligencia operativa, denominado Strategy Agent, se construyó con LangGraph y modelos de lenguaje de Anthropic y OpenAI, estructurado como un grafo de nodos secuenciales (recopilación de contexto, propuesta de estrategia, definición de reglas de fallback, planificación de lotes y validación del plan). La salida del agente se valida contra un esquema estricto (Zod) antes de persistirse, y su aprobación, edición o rechazo quedan sujetos a un usuario humano autorizado, en línea con el requerimiento de no dejar decisiones críticas sin intervención humana.

## Resultados

El desarrollo alcanzó una versión funcional de OSIRIS que cubre el flujo completo de operación de una campaña de voz. Se implementó la gestión de proyectos como unidad organizativa dentro de cada workspace, junto con la gestión de agentes de voz, incluyendo la provisión de agentes y números telefónicos a través de Retell AI y la asignación de voces disponibles por proveedor.

Sobre esta base, se desarrolló el módulo de campañas, que permitió su creación, edición, pausa, reanudación y cierre, así como la carga masiva de contactos mediante archivos CSV, con una vista previa de validación antes de su confirmación definitiva y persistencia del historial de cargas.

El resultado de mayor complejidad fue la implementación del Strategy Agent, capaz de generar un plan de ejecución —estrategia, distribución de llamadas por franja horaria y zona horaria, asignación porcentual entre agentes, nivel de concurrencia y reglas de reintento automático en oleadas sucesivas— a partir de los contactos y agentes disponibles en la campaña. Dicho plan quedó sujeto a aprobación, edición o rechazo por parte de un usuario autorizado antes de su ejecución.

Una vez aprobado el plan, el módulo de ejecución (wave-orchestrator) orquestó el disparo de llamadas respetando las ventanas horarias y niveles de concurrencia definidos, mientras que el procesamiento asíncrono de eventos de llamada permitió registrar en tiempo real el estado, la duración, el costo, el sentimiento detectado y la transcripción de cada interacción, junto con el almacenamiento de sus grabaciones.

Finalmente, se desarrolló un dashboard por campaña con métricas operativas en tiempo real —llamadas realizadas, completadas y fallidas, costo estimado y resultados por agente—, alimentado por eventos normalizados independientes del proveedor de voz utilizado, en línea con el diseño de adaptadores planteado en la metodología.

## Discusión

Los resultados obtenidos evidencian que OSIRIS cumple con su objetivo central de centralizar la planificación de campañas de voz con IA sin depender de un único proveedor, gracias a la arquitectura de adaptadores implementada. A diferencia de las interfaces nativas de proveedores como Retell AI, VAPI o Bland AI, que exponen métricas técnicas orientadas al procesamiento (latencia, tokens, duración), OSIRIS prioriza métricas de negocio y centraliza en una única herramienta la configuración, ejecución y análisis de campañas que de otro modo se gestionarían de forma dispersa.

La incorporación del Strategy Agent constituye el principal diferencial frente a plataformas de gestión de campañas convencionales, al proponer de forma automatizada una distribución de esfuerzo de llamado basada en datos históricos y de contacto, sin asumir por ello el criterio conversacional del agente de voz, que permanece bajo configuración de cada proveedor. La exigencia de aprobación humana sobre las recomendaciones generadas responde a una decisión de diseño deliberada, orientada a que la plataforma asista a los equipos de operaciones sin reemplazar su responsabilidad sobre decisiones sensibles.

Se identifican como líneas de trabajo futuro la incorporación de adaptadores adicionales (VAPI, Bland AI) sobre la interfaz ya definida, la ampliación de las métricas de negocio expuestas en el dashboard y la evaluación de resultados de la estrategia en escenarios de uso con mayor volumen de campañas concurrentes.

## Conclusión

OSIRIS demuestra ser una solución viable para centralizar la operación de campañas de voz basadas en Inteligencia Artificial, resolviendo la fragmentación operativa que enfrentan hoy los equipos que dependen de múltiples proveedores. La arquitectura de adaptadores adoptada garantiza que la incorporación de nuevos proveedores no

requiera modificar la lógica de negocio ya desarrollada, mientras que el Strategy Agent aporta valor diferencial al automatizar decisiones de distribución de llamadas manteniendo el control humano sobre su aprobación final.

Se espera continuar el desarrollo incorporando adaptadores para proveedores adicionales, profundizando el análisis conversacional de las transcripciones y evaluando la puesta en producción de la plataforma bajo un modelo de suscripción por organización.

## Referencias

- [1] NestJS, "Documentation," [Online]. Disponible: <https://docs.nestjs.com/>. Último acceso: julio 2026.
- [2] Meta Platforms Inc., "React Documentation," [Online]. Disponible: <https://react.dev/>. Último acceso: julio 2026.
- [3] Supabase Inc., "Supabase Documentation," [Online]. Disponible: <https://supabase.com/docs>. Último acceso: julio 2026.
- [4] Taskforce.sh Inc., "BullMQ Documentation," [Online]. Disponible: <https://docs.bullmq.io/>. Último acceso: julio 2026.
- [5] LangChain Inc., "LangGraph Documentation," [Online]. Disponible: <https://www.langchain.com/langgraph>. Último acceso: julio 2026.
- [6] Retell AI, "Retell AI Documentation," [Online]. Disponible: <https://docs.retellai.com/>. Último acceso: julio 2026.
- [7] Scrum.org, "What is Scrum?," [Online]. Disponible: <https://www.scrum.org/resources/what-is-scrum>. Último acceso: julio 2026.

## Datos de Contacto

Chas Díaz, Lucas Martín. UTN, Facultad Regional Córdoba. E-mail: [luquitaschasdiaz3@gmail.com](mailto:luquitaschasdiaz3@gmail.com)

Tombesi, Gino Paolo. UTN, Facultad Regional Córdoba. E-mail: [ginotomesi@gmail.com](mailto:ginotomesi@gmail.com)

Rottino, Juan Ignacio. UTN, Facultad Regional Córdoba. E-mail: [juan.rottino@gmail.com](mailto:juan.rottino@gmail.com)

Varengo Bossi, Juan Ignacio. UTN, Facultad Regional Córdoba. E-mail: [juanivarengo@gmail.com](mailto:juanivarengo@gmail.com)

Villegas, Ángel Nahuel. UTN, Facultad Regional Córdoba. E-mail: [NahuelVillegas4@gmail.com](mailto:NahuelVillegas4@gmail.com)

Rocca, Valentino Gabriel. UTN, Facultad Regional Córdoba. E-mail: [roccavalentino5@gmail.com](mailto:roccavalentino5@gmail.com)

|                                                                         |      |                                                                |           |
|-------------------------------------------------------------------------|------|----------------------------------------------------------------|-----------|
| AÑO                                                                     | 2026 | CURSO Y NRO.DE GRUPO                                           | 5K4 - G13 |
| NOMBRE DEL SISTEMA / PROYECTO                                           |      |                                                                |           |
| OSIRIS                                                                  |      |                                                                |           |
| CATEGORÍA (Solución De Negocio / Producto / Proyecto De Impacto Social) |      |                                                                |           |
| Producto                                                                |      |                                                                |           |
| HERRAMIENTAS Y TECNOLOGÍAS UTILIZADAS                                   |      |                                                                |           |
| ÁMBITO DE APLICACIÓN                                                    |      | NOMBRE Y VERSIÓN                                               |           |
| ENTORNO DE DESARROLLO                                                   |      | Visual Studio Code 1.134.0                                     |           |
| REPOSITORIOS Y VERSIONADO                                               |      | GitHub                                                         |           |
| PROGRAMACIÓN                                                            |      | TypeScript, NodeJS, NestJS, React, Vite                        |           |
| BASE DE DATOS                                                           |      | Postgres via Supabase, Redis como cola                         |           |
| COMUNICACIÓN INTERNA                                                    |      | Discord 10.0.262, Whatsapp 2.3000.1045554756                   |           |
| CAPACITACIÓN                                                            |      | Documentacion RetellAI, Vapi, Youtube                          |           |
| PRUEBAS DE SISTEMA                                                      |      | Pruebas manuales: Navegador<br>Pruebas unitarias: Jest, Vitest |           |
| GESTIÓN DEL PROYECTO                                                    |      | Jira 2026                                                      |           |
| DOCUMENTACIÓN                                                           |      | Google Drive                                                   |           |
| MODELOS                                                                 |      | Canva, Inkscape                                                |           |