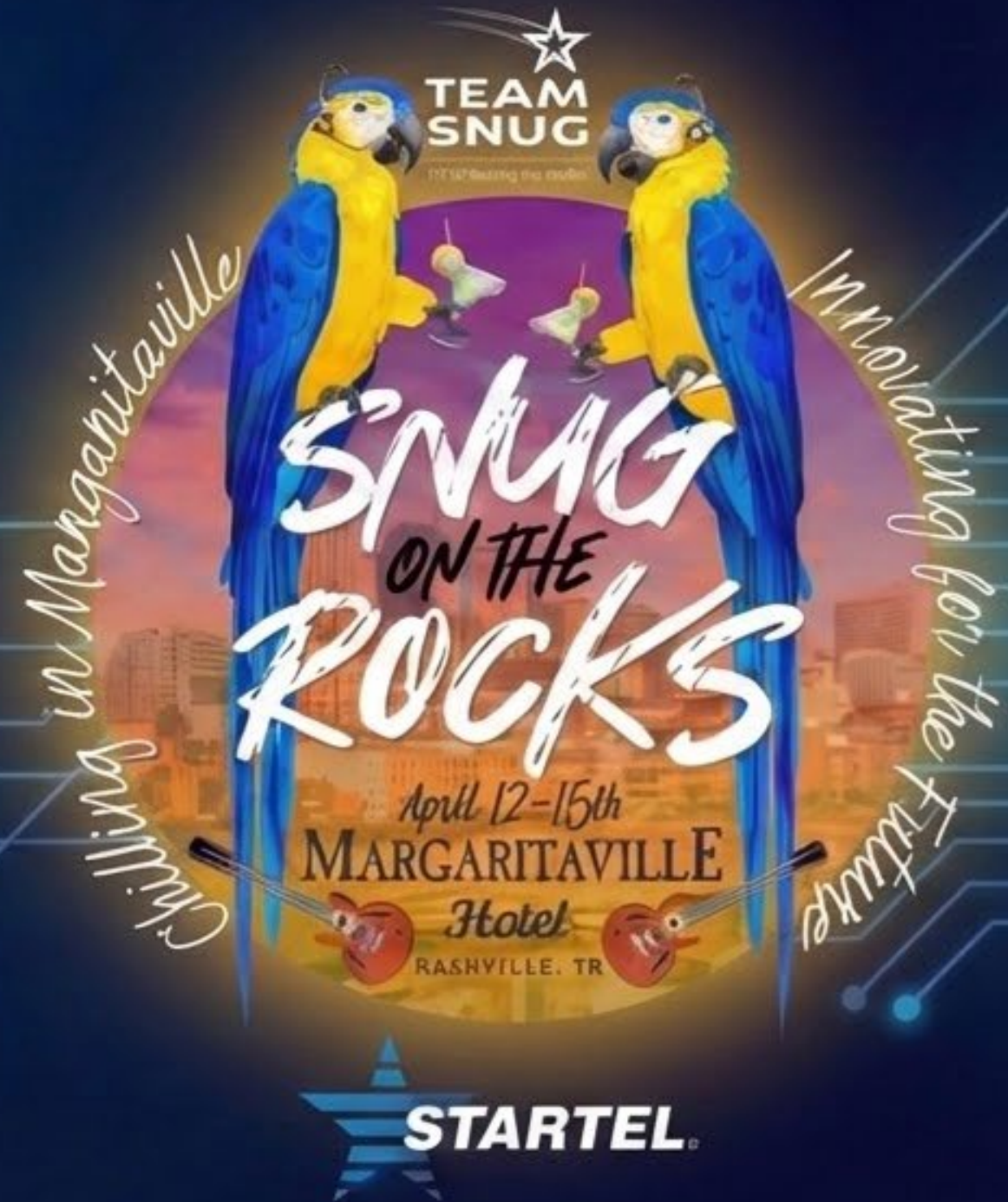


# How the API Transformed Our Operations



**Chad Roberts**  
Chief Operating Officer

From managing manual fragments to building resilient, automated platforms



# What is an API?

An API (Application Programming Interface) is a set of rules and protocols that allows different software applications to communicate and exchange data with one another.



# What Does That Mean To The Non-Software Engineers Like Me



**API = Connection to the App Store**

**Startel = Think of Like Your iPhone**

We all have the same base system. What's different is what we build around it. We stopped asking what Startel could do, and started asking what we could build.

# Our systems worked. They just didn't work together.



Agents were forced to act as the **integration layer**.



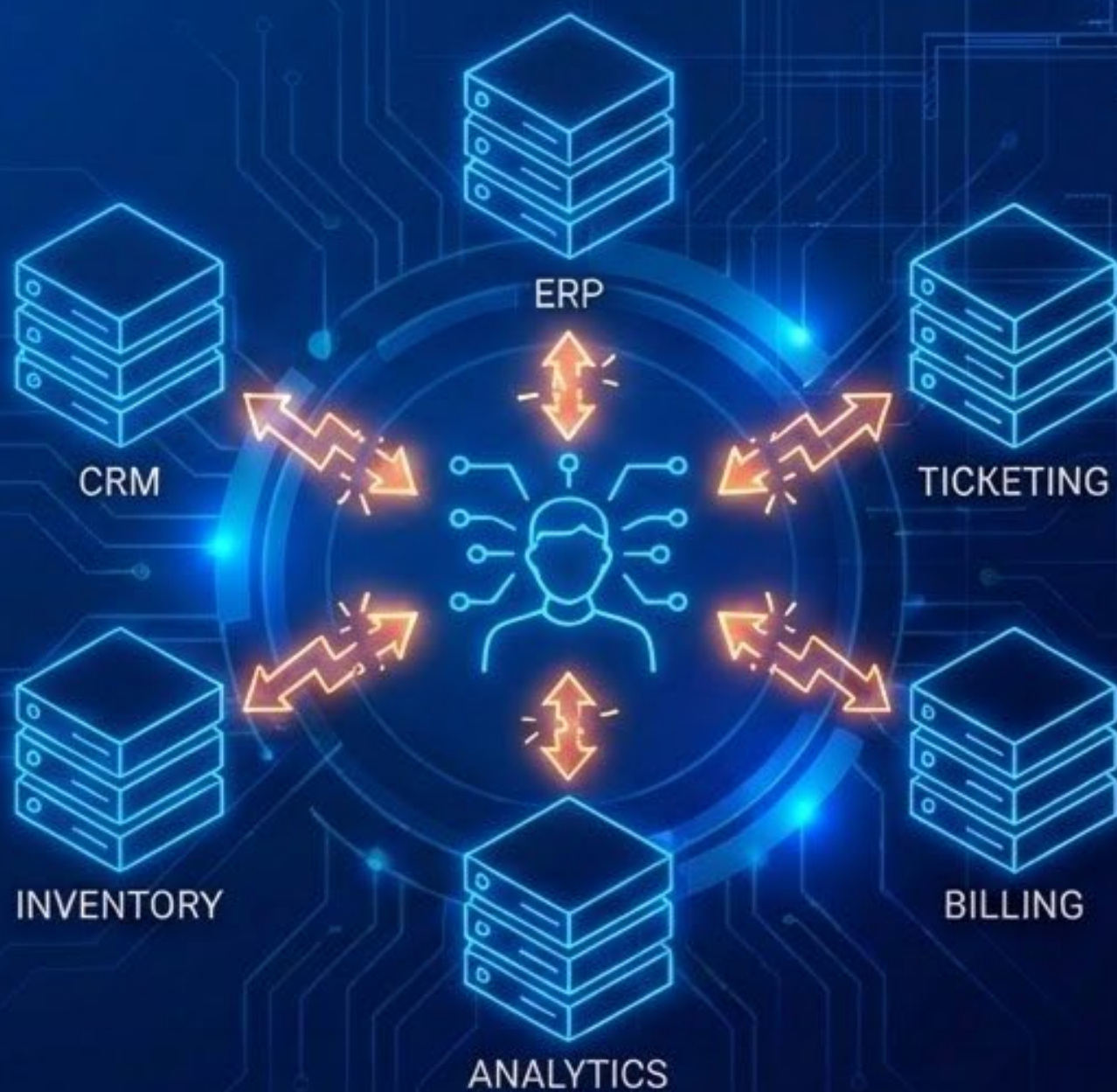
Constant **context switching**.



Manual data **re-entry** across platforms.



**Judgment calls** made without full operational visibility.



**"We had good people doing their best inside a broken flow."**

# We moved the **complexity** out of the agent's head and into **the system**.

## People Move Data

- Manual decisions fill the gaps.
- Agents navigate multiple, disconnected systems.
- Execution is inconsistent and relies on memory.

## Systems Move Data

- APIs directly connect systems **behind the scenes**.
- Agents operate in a single, **guided environment**.
- Real-time data guarantees consistent outcomes.

**We didn't make  
agents faster.**

**We removed what  
slowed them down.**

**11 Consolidated Queues**

**43 Distinct Queues**

Primary Queue A  
Primary Queue B

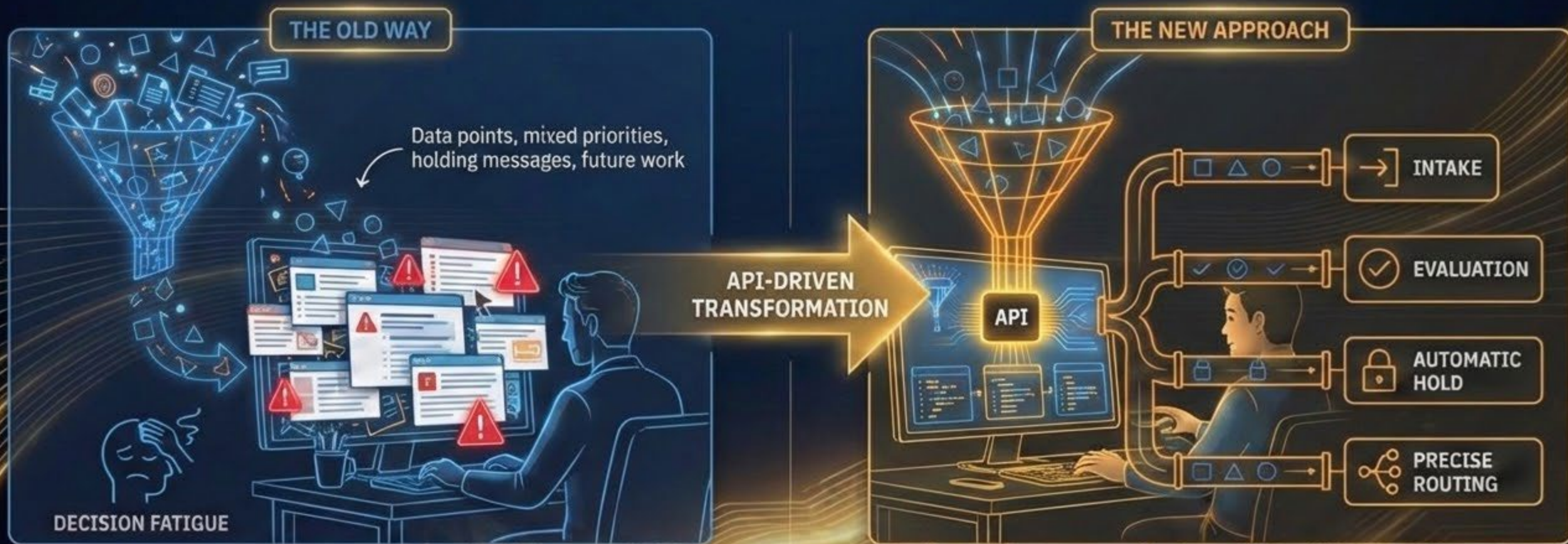
Queues heavily  
consolidated:  
From 43 down to 11.

Training is no longer a  
bottleneck; it is integrated  
into standard onboarding.

**85% of call volume**  
is now in only  
**two queues.**

We didn't improve training—we removed an entire category of it. We redefined what 'ready' looks like.

# We stopped managing work manually and started managing when work appears. We're no longer just handling work. We're shaping it.



Everything lived in one place. Agents manually sorted future work, holding messages, and mixed priorities, causing severe decision fatigue.

We separated intake from processing. Custom API applications evaluate the data, hold future work automatically, and route it to the exact place at the precise time it is needed.

# Identifying Your First API Opportunity



**Start with your current workflow.** Look relentlessly for manual steps, data re-entry, and constant system switching.



**The Litmus Test:** If a person is acting as the bridge between systems, that's your very first API opportunity.



**Ask clients:** "What happens after we send you data? Where does it go?" (Turns it into an operational conversation, not a technical one).



**Think past  
just client  
integrations**

**We didn't look for innovation. We looked for irritation.**

# Startel & API Integration: The Intelligent Message Journey

## 1. STARTEL INTAKE



Raw Message

Raw caller information captured.

## 2. API PROCESSING



Intelligent parsing, enrichment, and categorization.

## 3. STARTEL: ORGANIZED & READY



Polished Message

Polished caller information queued for precise dispatch.

# Case Study: Eliminating Administrative Overhead at Scale



## CONTEXT

### Large Property Management Company

A multi-property client with **260** locations spanning **7** states. Each required distinct message access, billing, and reporting.

## THE OLD WAY

**260** distinct sub-accounts = **260** setups,  
**260** maintenance points, **260** points of failure.  
Every new property multiplied complexity.

## THE NEW APPROACH

**1** Single Account  
A custom portal utilizes API data to provide location-based access, automated billing, and segmented reporting from one central hub.

We built around data, not accounts. We eliminated an entire category of operational overhead.

# Case Study: Large Hospital Cyberattack Outage

## THE SCENARIO

A sudden, complete telephony, network, EPIC, and all internal systems outage for a massive healthcare system.



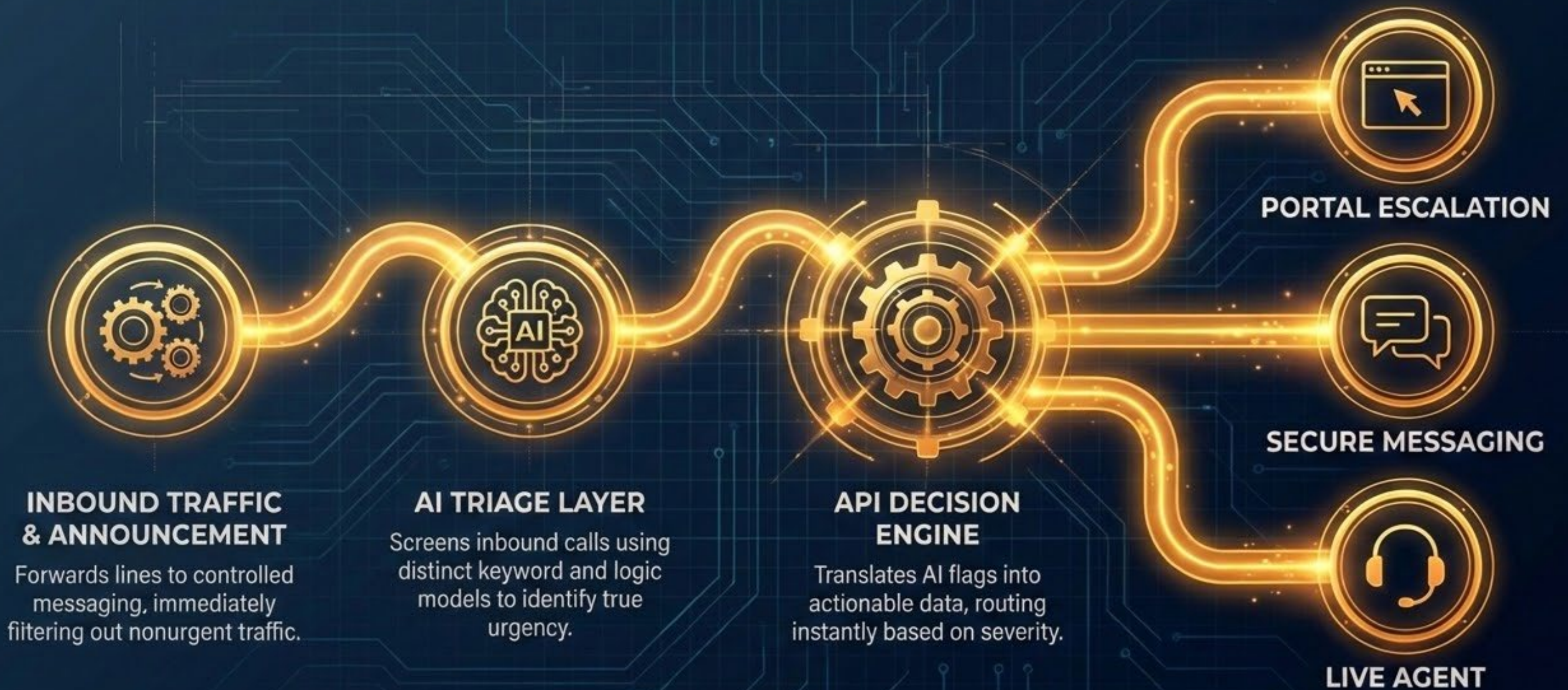
## THE THREAT

- Thousands of inbound calls per day still requiring handling.
- Patients needing urgent escalations with zero delay.
- Internal healthcare staff completely locked out of their outbound communication systems.

## THE MANDATE

Rebuild the entire communication flow externally, in real-time, without access to the client's phone system.

# HOSPITAL OUTAGE WORKFLOW



# 10 DAY OUTAGE METRICS



**78,338**

TOTAL CALLS  
HANDLED



**2,435**

URGENT CALLS DIGITALLY  
ESCALATED TO PORTAL



TASLiNQ UTILIZED  
TO PLACE

**7,263**

CALLS



**12,317**

INBOUND CALLS  
SCREENED BY AI



**ONLY 231**

INBOUND CALLS REQUIRED  
LIVE AGENT ESCALATION



STARTEL'S  
AI PLATFORM ABSORBED

**24,099.5**

MINUTES (401.66 HOURS)



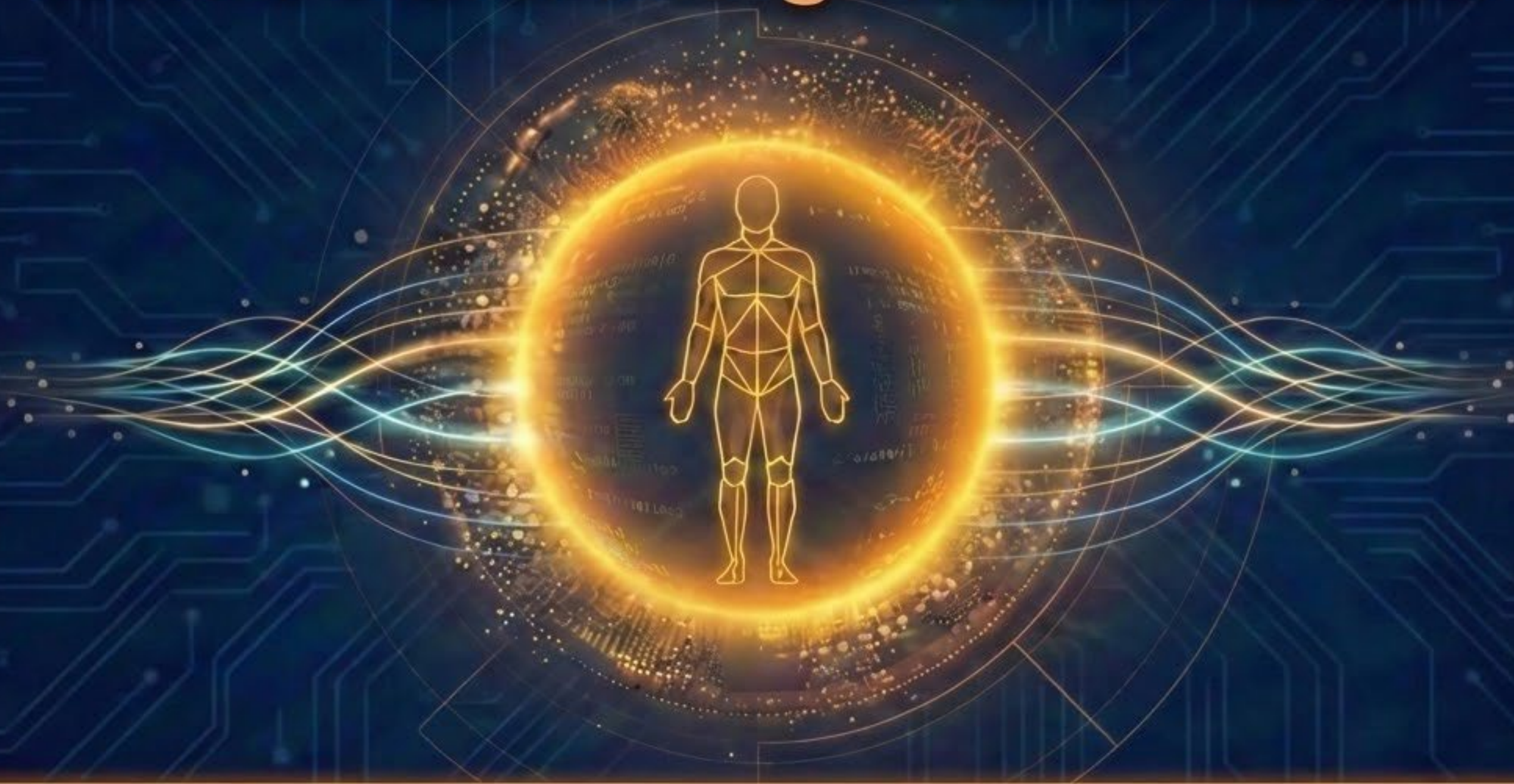
**1,449**

TASLiNQ & PORTAL USERS  
ENABLED FOR OUTBOUND  
CALLING VIA CELLULAR

**So what does an API mean to me today,  
beyond the technical definition?**



# The API Is the Engineer in Action



**An API is only as powerful as the person designing it.  
The technology enables it. The Engineer defines it.**

**At some point, every growing system reaches a decision.**



**Do we need an API Engineer in house?  
And if so, what does the right hire look like?**

# SOFTWARE ENGINEER

Wanted: Experience & Qualifications

[Apply Now](#)

[Save Job](#)

## PROFESSIONAL EXPERIENCE & LEADERSHIP (The Role & Operational Ownership)

### SENIOR DEVELOPER & SYSTEMS ARCHITECT

- Extensive .NET development of robust, scalable applications.
- Full-stack expertise across backend & modern frontend.
- Proven track record of ground-up application lifecycle.
- Translate business needs into complete technical solutions.
- Comfortable acting as sole developer, taking full project ownership.

### OPERATIONAL LEADERSHIP

- Self Directed and highly independent.
- Requirements gathering, system design, troubleshooting, ongoing improvements.

## TECHNICAL QUALIFICATIONS & PERFORMANCE (Beyond Code & Reliability Under Fire)

### ARCHITECTURE & DATABASE MASTERY

- Deep understanding of software architecture (maintainable, scalable).
- Advanced SQL Server experience (design, indexing, tuning).

### SYSTEM RELIABILITY & INCIDENT RESPONSE

- Reliable and responsive under pressure.
- Ability to be on call, handle critical issues, deliver rapid resolutions.

**Not just someone who can code, but someone who can design systems around operations. – The ultimate qualification.**

And if there's one thing I'd leave you with...

“don't start with the technology. Start with your workflow, follow the friction, and build from there.”

Thank you.

