WHITE PAPER

# The Architecture of a Real Voice Agent

*Production voice AI for the Indian phone network*

PREPARED FOR
CTOs, principal engineers, and platform architects

PREPARED BY
GRX10 Solutions Pvt Ltd
Bengaluru, Karnataka, India

DATE
April 2026

grx10.com · prem@grx10.com

INTRODUCTION

# Most voice AI is not optimized for true human-like conversations

Most voice AI products on the market today are not voice products.

They are text products with a microphone strapped to one end and a speaker strapped to the other. A caller speaks; their audio is transcribed to text; the text is sent to a language model; the response text is synthesised back to audio; the audio is played back. Four hops, four vendors, four points of failure, and somewhere between ~500 and 1000 milliseconds of latency before the agent says a single word.

This document describes a different architecture: a single multimodal speech model that hears, reasons, and speaks in one process — what the industry now calls speech-to-speech or unified voice — and the engineering work required to make that architecture actually run on the Indian phone network at telephony bandwidth.

> *This paper is our attempt to describe what we believe to be state of the art voice ai for production systems for Indian telephony. Although we don't name suppliers or frontier models available today, we will name the patterns, the trade-offs, and the design decisions, and explain why each one matters for an Indian deployment. The goal is to give a technical buyer enough to evaluate the platform without giving a competitor enough to copy it.*

SECTION ONE

# Why the cascaded stack has run out of road

## 1.1 The four-hop pipeline

The conventional voice AI architecture has been remarkably stable for five years:

```
Audio in  →  STT  →  LLM  →  TTS  →  Audio out
```

Every vendor in the market — including the developer platforms (V, R), the orchestration aggregators (B, Sy), and the TTS-rooted conversational suites (E) — runs some variant of this pipeline. Some co-locate components in the same data centre to shave milliseconds. Some let the customer bring their own STT, LLM, or TTS provider. Some swap one component for an open-source model to cut cost. The shape of the pipeline does not change.

Independent benchmarks place a well-tuned cascaded stack at 450–750 ms end-to-end in production, with best-in-class optimised configurations reaching the 500–800 ms range only with aggressive streaming and tight component co-location.

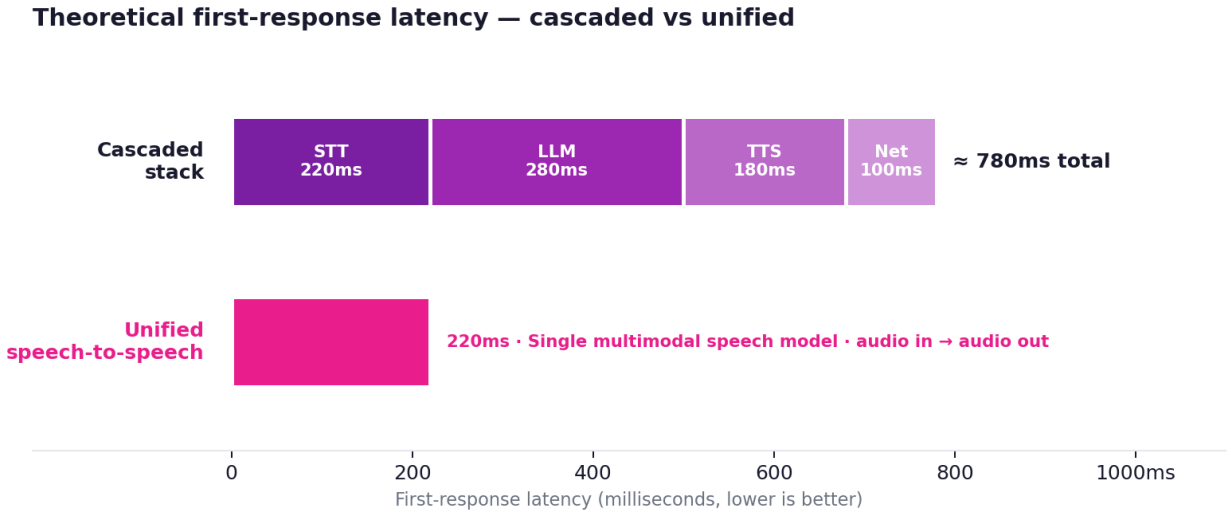



*Fig 1 · Theoretical first-response latency — cascaded stack (≈ 780 ms across STT, LLM, TTS, and network) versus unified speech-to-speech (≈ 220 ms in a single forward pass).*

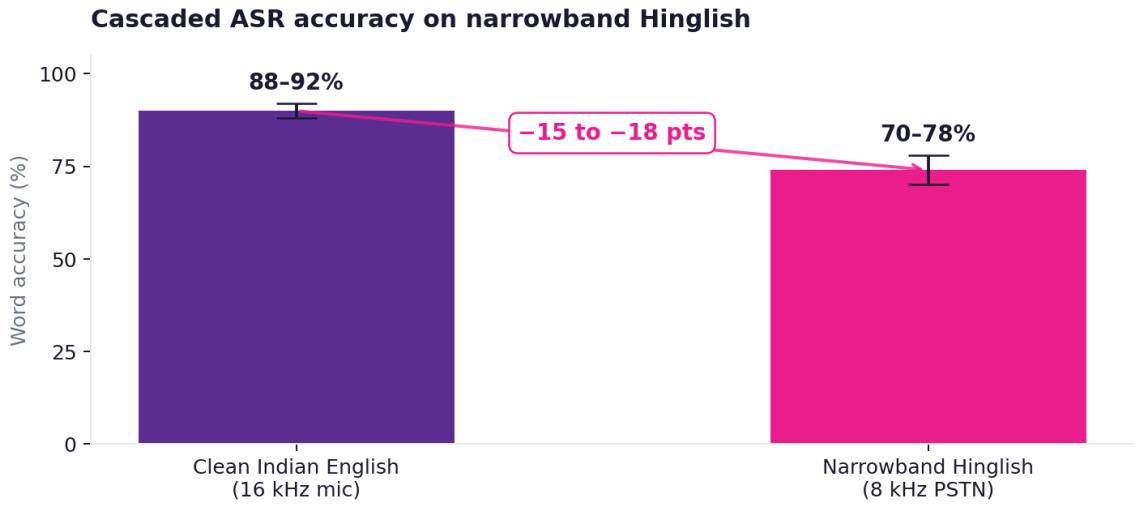
| Stage | Typical latency |
|---|---|
| STT (streaming first token) | 100–400 ms |
| LLM (time-to-first-token) | 200–800 ms |
| TTS (time-to-first-byte) | 40–500 ms |
| Network + buffering | 50–150 ms |

These numbers add up. They also do not include the prosodic cost — every hop is a one-way translation. The transcript step throws away tone, hesitation, urgency, sarcasm, and the half-second of silence that meant the caller was thinking. The LLM never sees that the caller's voice cracked when they said "my mother is in the hospital." The TTS step is a synthesis problem in isolation — the model generating the spoken reply has no idea what the caller actually sounded like.

## 1.2 Where the pipeline breaks for India

The cascaded stack has a second, less obvious problem: every component must be tuned for the input distribution. For India, that distribution is hostile.

| Constraint | Impact on cascaded stack |
|---|---|
| Narrowband audio | PSTN delivers 8 kHz μ-law, not the 16–48 kHz that most ASR models train on. Global ASR drops from 88–92% accuracy on clean Indian English to 70–78% on narrowband Hinglish. |
| Code-switching | 60% of online communication among Hindi speakers is code-switched. A single sentence — "sir aap ka loan approve ho gaya, please share your PAN card" — uses two languages, two scripts when transcribed, and one continuous prosodic line. Most STT engines pick a language at session start and stay there. |
| 22 official languages | Accents vary every 200 km. A Mumbai caller and a Madurai caller are mutually intelligible to humans and adversarial to most STT models. |
| Regulatory load | TRAI DLT registration, DPDP data residency, mandatory AI self-disclosure at the start of the call. None of these are solved at the model layer. |




*Source: published India voice-AI guides and IIT-Delhi Hinglish ASR studies.*

*Fig 2 · Cascaded ASR accuracy drops 15–18 points moving from 16 kHz mic-quality Indian English to 8 kHz PSTN Hinglish. Source: published India voice-AI guides; IIT-Delhi Hinglish ASR studies.*

> *The cascaded stack is not just slow for India. It is structurally mismatched to the input.*

## 1.3 What "speech-to-speech" actually means

A native speech-to-speech model takes raw audio in and emits raw audio out, in a single forward pass, with no intermediate text representation. The model is trained on aligned audio pairs — caller turns and agent turns — and learns to produce a vocal response conditioned on the acoustic features of the input, not just the words.

The architectural diagram collapses to:

```
Audio in  →  Single multimodal speech model  →  Audio out
```

Three things change.

**First, latency falls structurally.** The four hops become one. There is no STT-to-LLM token boundary to wait on, no LLM-to-TTS prompt assembly, no synthesis startup time on a fresh sentence. Public unified-architecture benchmarks land at 200–250 ms end-to-end. Our internal target is sub-second first-word latency on Indian PSTN — conservative compared to what the architecture allows, because telephony adds its own irreducible network and codec overhead that no model can compress away.

**Second, the model hears tone.** Hesitation, urgency, and emotional cadence are now first-class features the model can attend to. A caller who pauses mid-sentence to think gets a different reply than a caller who blurts out the same words. The model can match register: firm with an agitated caller, warm with a confused one.

**Third, code-switching becomes free.** A single multilingual speech model trained on Hinglish data does not have to "switch" — it has only ever known the code-switched distribution. Hinglish in produces Hinglish out. There is no Hindi-prompt-to-English-LLM-to-Hindi-TTS detour, no language dropdown, no translation tax.

> *This is a whole different category of system – not just a tuning improvement.*

SECTION TWO

# The architecture

## 2.1 High-level shape

GRX10 Voice AI is built around three concentric layers and a patent-filed bridge.

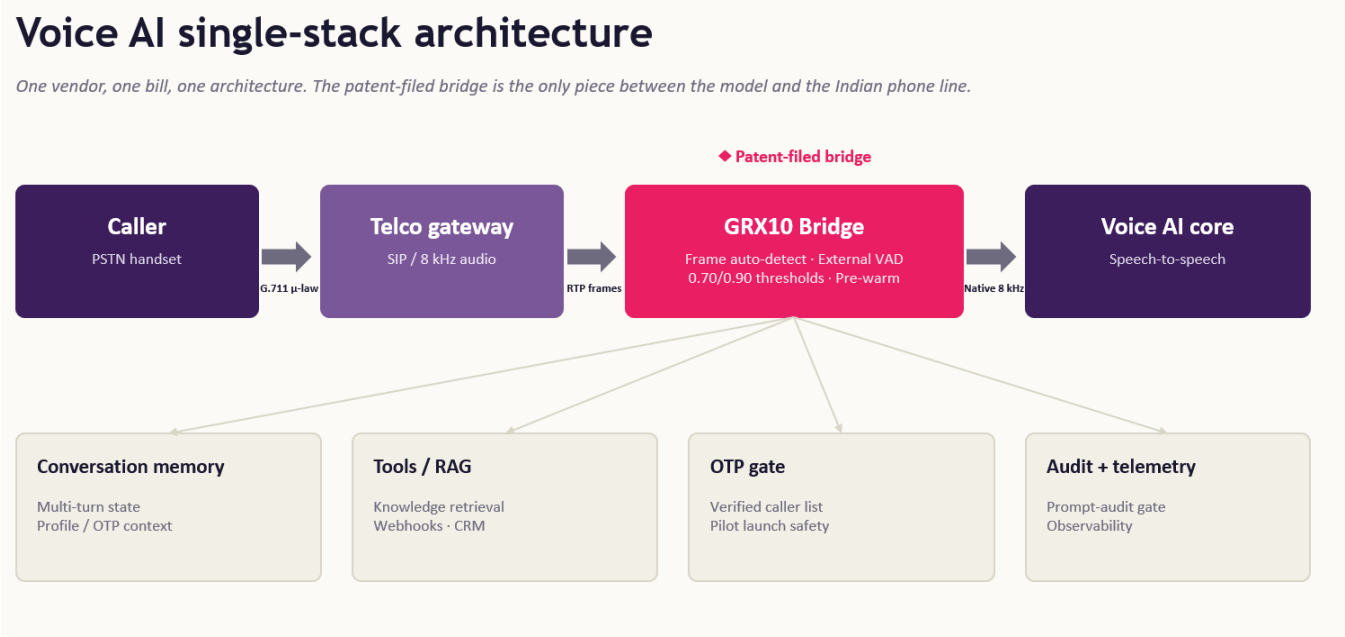



*Fig 3 · GRX10 Voice AI single-stack architecture. The patent-filed bridge sits between the carrier-handed audio stream and the multimodal speech model, with four supporting services (memory, RAG, OTP gate, audit and telemetry) running orthogonally.*

Three things are deliberate about this shape:

1. The customer never sees the telephony layer. Phone numbers, SIP credentials, dialplans, codec negotiation — none of it surfaces in the console. The customer creates an agent, picks a voice, uploads a knowledge base, and dials.
2. The voice gateway is owned end-to-end. It is not an orchestration shim that rents a TTS API and an STT API. It is the audio path itself.
3. The customer surface is a separate process with its own database, auth, and billing. A bug in the realtime path cannot take down the dashboard, and vice versa. Customers in a live call do not lose audio because someone deployed a billing migration.

## 2.2 The realtime audio path

The hot path — the path a packet of audio actually travels — is short by design. From the caller speaking to the model receiving audio there are exactly three stages: PSTN to PBX, PBX to gateway, gateway to model. Return path is the inverse.

We will not detail the codec transformations, frame sizes, or specific carriers. What matters is the principles:

- No transcription on the hot path. The model receives audio, not text. We do not run STT in line with the call.
- One model, one connection. The voice gateway holds a persistent realtime connection to a single multimodal speech model for the duration of the call.

- Pre-warm before the phone rings. The model session, system prompt, and greeting are loaded before the SIP call dials out. The first audio frame is ready the instant the caller picks up.
- Direct PSTN-bandwidth audio. We do not upsample 8 kHz telephony audio to 16 kHz to feed a 16 kHz-trained model. Every upsample-then-process-then-downsample cycle is latency, distortion, or both.

> *The result is an audio path that has fewer hops than the transport of a normal phone call between two humans. There is no inherent reason a voice AI should be slower to respond than a human on the same line.*

## 2.3 Why we do not run cloud STT in line

A reasonable engineer reading the above will ask: if the model takes audio directly, what about transcripts? You still need them for review, search, compliance, QA.

We do — but not synchronously and not in the audio path. After the call ends, both audio streams (caller and agent) are uploaded, mixed, and submitted to a separate batch transcription job that returns role-attributed, timestamped segments with confidence scores. This decoupling matters for two reasons:

1. Live transcription configurations on multimodal speech models measurably degrade output audio quality. Asking the model to emit a transcript while it is also generating speech changes its decoding behaviour, and the spoken response gets worse. The buyer hears a less natural voice in exchange for a real-time transcript that no human is reading mid-call. Bad trade.
2. Batch transcription can use a larger, more accurate model than would be feasible to run streaming. The transcript that lands in the customer dashboard is more accurate than what a streaming STT could have produced live, at lower cost.

SECTION THREE

# Voice activity detection — the hardworking, load-bearing piece

## 3.1 Why off-the-shelf VAD does not work on the phone

Speech models that ship with built-in voice activity detection — the logic that decides when the caller has stopped talking and the model should reply — are tuned on microphone-quality audio, typically 16 kHz or higher, with a clean noise profile. We tested these built-in detectors extensively against Indian PSTN audio at 8 kHz μ-law. They did not reliably detect speech.

This is a training-distribution mismatch. The detector was never shown narrowband telephony in training, so it does not fire on it.

## 3.2 What we do instead

We run a telephony-tuned voice activity detector locally, in the gateway, next to the audio path — not inside the model. The detector runs at native PSTN bandwidth (no upsampling), uses a lightweight on-device model, and emits explicit start-of-speech and end-of-speech events that the speech model treats as authoritative.

This decision has three downstream consequences:

| Capability | How it works |
|---|---|
| Controlled barge-in threshold | When the AI is speaking, we raise the speech-detection threshold so the model's own audio leaking back into the phone line cannot trigger a self-interrupt loop. When the AI is quiet, we lower the threshold so a real human voice clears it easily. The thresholds are per-agent tunable; defaults reflect a year of production tuning. |
| Silence as signal | Silence on a phone call means something. A 600 ms pause is the caller thinking; a 4-second pause is the caller waiting for us to take the next turn; a 12-second pause is the caller having walked away. Built-in VAD inside a closed model gives us none of these affordances. |
| Replaceable detector | When a better narrowband detector ships, we swap it without touching the model. The architecture does not couple speech understanding to speech detection. |

> *This is the kind of detail that gets missed in a feature comparison. It is also the difference between a voice agent that feels alive on a phone call and one that constantly speaks over its caller.*

SECTION FOUR

# Multilingual without translation tax

## 4.1 The wrong way: language as a configuration field

In a cascaded stack, "multilingual" usually means: the customer sets a language code at agent creation time; STT, LLM, and TTS each load the corresponding model variant. Switching languages requires changing the configuration. Code-switching mid-call requires either a language-detection layer that re-routes mid-utterance, or — more often — a degraded experience where the caller is silently forced into the agent's configured language.

This is why most platforms list "Hindi support" but stumble on real Hinglish. The platform supports Hindi and supports English. It does not support a single utterance that contains both.

## 4.2 The right way: native multilingual at the model level

Our approach:

- **The speech model is multilingual at the audio level.** It was trained on code-switched speech as a natural part of its data distribution. Hinglish is not a configuration; it is a class of input the model already understands.
- **The agent is instructed to mirror the caller.** A persistent instruction in every agent prompt directs the model to reply in the same language and register the caller used, including mid-sentence code-switching.
- **Regional accent inference is automatic.** Destination phone numbers are parsed at dial time to infer a likely regional accent — Mumbai, Delhi, Chennai, Bangalore, Hyderabad, Kolkata — for India; analogous inference for US, UK, UAE, and a long fallback list. The accent inference is a default that the caller can always override by saying so.
- **Voice personality is matched at runtime.** The chosen voice's gender and timbre are described to the model alongside the system prompt so word choice matches vocal texture.

The customer never thinks about language. They write the agent's purpose in English, and a Tamil-speaking caller in Madurai has a Tamil conversation with that agent. A Mumbaikar gets a Mumbai-accented reply. A Bangalore engineer who code-switches between Kannada, Hindi, and English in the same sentence gets a reply in the same code-switched register.

> *The buyer-visible knob is off. Multilingual is not a feature. It is a property.*

SECTION FIVE

# The hidden cost of stitched vendor stacks

This is the section that most buyers, and most platform vendors, do not want to discuss honestly.

## 5.1 What "voice AI platform" usually means

When a US developer platform (V, R) says it is a voice AI platform, what it actually offers is orchestration. The platform takes audio in over its WebSocket; routes the audio to a customer-configured STT provider; pipes the transcript to a customer-configured LLM provider; pipes the response text to a customer-configured TTS provider; and routes the resulting audio out via a third-party telephony carrier.

The buyer's invoice arrives from:

3. The platform vendor (orchestration fee, often $0.05/min)
4. The STT vendor (per-minute or per-character)
5. The LLM vendor (per-token, hard to predict)
6. The TTS vendor (per-character or per-minute, depends on plan tier)
7. The telephony carrier (per-minute, varies by destination)
8. Sometimes a recording storage vendor (S3 or equivalent)
9. Sometimes a separate observability vendor

Independent analyses confirm this: on V, the all-in cost is $0.18–$0.33/min despite the $0.05/min advertised rate. On R, real costs run $0.07–$0.31/min depending on configuration. On E's flagship conversational AI surface, low-latency TTS is gated behind the $1,320/month Business plan.

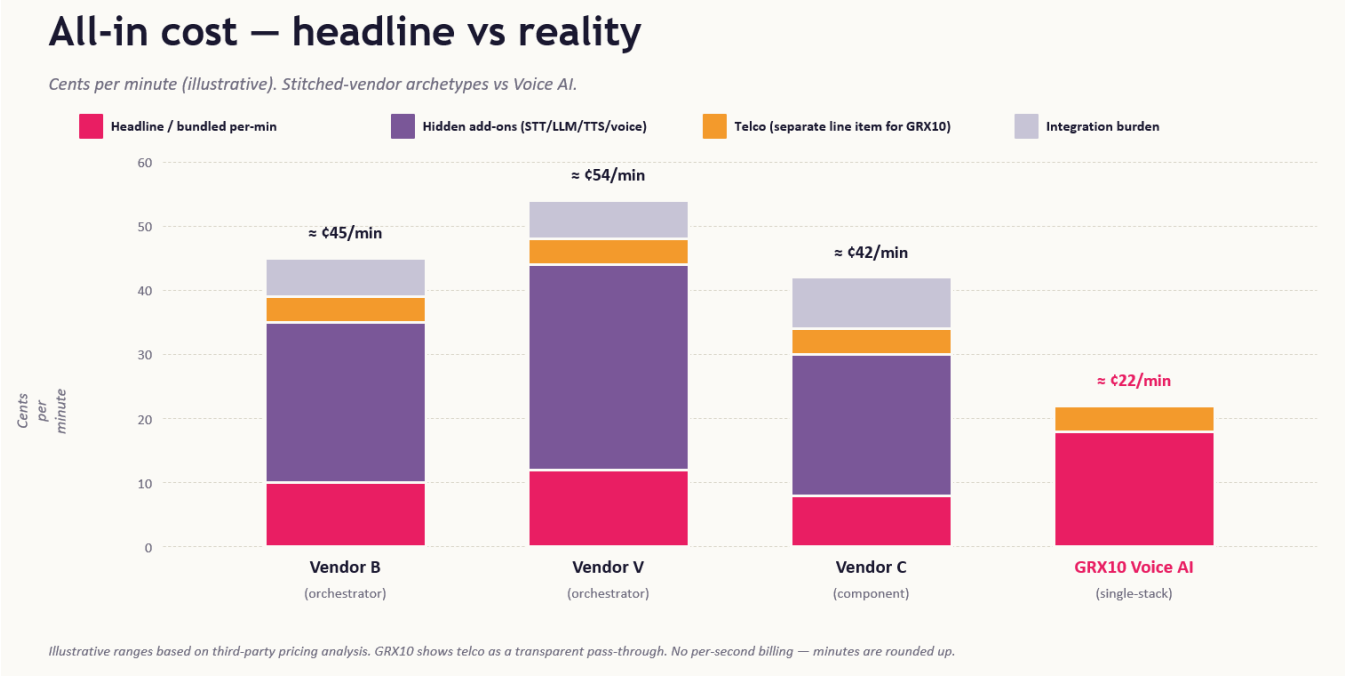



*Fig 4 · All-in cost — headline vs reality. Stitched-vendor archetypes vs Voice AI. Sources: third-party Vapi pricing breakdowns (Telnyx, Dograh); Retell pricing analysis (CheckThat); ElevenLabs and Synthflow pricing reviews.*

## 5.2 The architectural cost

The financial cost is the easy part of this story. The architectural cost is harder to see and harder to fix.

Every additional vendor in the realtime audio path is:

| Cost | What it does to the system |
|---|---|
| Another network hop | Audio leaves the platform, travels to the STT provider, the response travels to the LLM provider, then to the TTS provider, then back. Even with co-location, each hop adds 20–80 ms of round-trip time. |
| Another point of failure | The STT provider has an incident. The LLM provider hits rate limits. The TTS provider rolls out a model update that changes voice characteristics. Each is its own status page, its own SLA, its own 3 a.m. page. |
| Another seam where context is lost | The STT step throws away tone. The LLM step throws away interruption patterns. The TTS step has no idea what the caller's voice sounded like. |
| Another invoice to reconcile | The customer's finance team has to match six separate bills against one production product. Anomaly detection across vendors is something the customer builds, badly. |

For a US developer-buyer building a custom voice product, the BYO model is reasonable. It buys flexibility. For an Indian SMB or mid-market buyer running a sales floor, a collections team, or an inbound support line, the BYO model is a tax — a tax in latency, in operational load, in finger-pointing when something breaks.

## 5.3 The unified-vendor model

GRX10 Voice AI is a single product with a single bill. The customer signs one contract, configures one agent, gets one set of phone numbers, sees one usage dashboard, and calls one support number when something breaks. There is no STT vendor, no LLM vendor, no TTS vendor, no telephony vendor for the buyer to think about.

This is not a packaging trick. The architectural decision to use a single multimodal speech model is what makes the unified-vendor model possible. With a cascaded stack, the unified bundle would still hide six providers behind it. With a single-model architecture, there is genuinely one realtime model session, owned by one vendor, with one observability surface.

> *The procurement implication: one throat to choke. The technical implication: the audio path is shorter. They are the same decision viewed from two angles.*

SECTION SIX

# Knowledge, guardrails, and prompt engineering

## 6.1 Tiered context retrieval

Most voice AI platforms hand customers a flat vector store: upload your documents, the system embeds them, retrieves the top-k chunks per turn, stuffs them into the LLM context. This is standard RAG.

It is also wrong for telephony. Telephone conversations rarely require fine-grained semantic matching across a corpus. They require the right document, surfaced at the right moment, with enough context to be useful but not so much that the model wastes tokens on irrelevant chunks.

Our retrieval is tiered, not flat:

| Tier | When it fires |
|---|---|
| Always-in-context summaries | Every uploaded document gets an automatic summary at upload time. All summaries live in the agent's persistent context within a fixed token budget. The model always knows what knowledge exists. |
| Keyword-relevant chunks on demand | When the caller's last utterance contains terms that match a document, the relevant chunks are pulled in for that turn. |
| Expanded window for deep dives | When a conversation drifts deep into one topic, the chunk window for that document widens to give the model room to reason. |

The result is a knowledge layer that performs better on telephone-shaped conversations than naive vector RAG, with lower per-turn token cost.

## 6.2 Guardrails in the same language as the caller

Every agent has a list of topics it must not discuss — set by the customer, enforced by the prompt. The interesting engineering choice is not the list itself but the failure mode: when the agent must refuse, it refuses in the same language the caller spoke, with the agent's persona intact, never with the tell-tale "I can't help with that as an AI" break-character response.

This composes with the multilingual layer. A Marathi speaker who asks an out-of-scope question gets a polite Marathi refusal in the agent's voice. A Hinglish speaker gets a Hinglish refusal. The guardrail layer never strips the language and persona.

## 6.3 Auto-prompt with versioned, immutable revisions

Customers describe their agent's purpose in one or two sentences. A separate model expansion turns that into a full system prompt with role, persona, knowledge grounding, guardrails, and tone. The customer can edit the result. Once they lock it, that exact prompt becomes a versioned, immutable agent revision. Old versions stay callable so in-flight share-link traffic does not break when an admin re-tunes.

> *Voice agents drift in production in ways that text agents do not. A small change to a prompt that reads fine on paper can change the agent's pace, register, or barge-in behaviour. Versioning is not a nice-to-have; it is the only way to roll back safely when a Tuesday-afternoon edit makes the bot sound robotic.*

SECTION SEVEN

# Observability and the realtime ops surface

A voice AI platform without observability is a customer-support nightmare waiting to happen. The conversations are gone the moment the call ends. The customer wants to know what happened on the call, why the bot said that, and whether it was actually their fault for ambiguous prompt language.

Our observability surface, briefly:

| Surface | What it gives the operator |
|---|---|
| Structured logs on every hot path | Every audio frame, every model event, every barge-in decision, every billing decision is logged with structured JSON for after-the-fact debugging. |
| Per-agent versioned prompt history | Every change is auditable. We can replay what the agent looked like on the day of the call. |
| Mixed audio + role-attributed transcript | Customers hear the call and read the conversation in the same view, per call. |
| Live usage metering | Per-minute usage is visible in the dashboard, reconcilable against the customer's own call records, and updated continuously rather than batched at the end of the billing cycle. |
| Boot-time environment validation | The service refuses to start with a misconfigured secret rather than starting in a half-broken state. No silent-half-running production. |

> *This is unglamorous. It is also what a serious enterprise buyer evaluates after they get past the demo.*

SECTION EIGHT

# Telephony for India

## 8.1 Managed SIP, included by default

Every customer is auto-assigned managed SIP trunk capacity at signup, with TCP-default and a UDP fallback. No admin handholding, no carrier paperwork, no separate procurement cycle. The customer can dial out and receive calls from the moment they create their first agent.

For India-bound calls, this means we own the carrier relationship, the codec negotiation, the rate per minute, and the interconnect quality. The customer does not need to know what any of that means.

## 8.2 Bring-your-own SIP for buyers who need it

Some customers — typically larger or regulated — need to bring their own carrier. Reasons we have seen in practice:

- They have negotiated a lower per-minute telco rate than ours and want to keep the savings.
- A regulator requires calls to originate from a carrier they own the contract with.
- They have an existing PSTN contract that runs for two more years and they do not want to dual-pay.

Our PBX accepts their carrier credentials at agent-creation time and routes through their trunk. The realtime model, the gateway, and the customer dashboard are unchanged. BYO SIP is a configuration toggle, not a different product.

## 8.3 OTP-gated dialing for new accounts

Voice AI platforms are an obvious target for fraud: sign up with a stolen card, exhaust free-trial credits, spam a million numbers. The conventional defence is rate limiting after the fact. Ours is structural: on the entry-tier plan, an account can only dial numbers it has verified by OTP. The verification is a one-time cost per destination, and it makes outbound spam architecturally impossible without owning every phone number the spammer wants to reach.

Higher tiers lift the restriction once the customer is verified by other means.

## 8.4 Transparent per-minute billing with separate telco line item

Platform and telco usage are billed per minute, in line with the rest of the voice AI market and the underlying telco rate cards we pass through. Where we differ is in transparency, not in increment.

The platform fee and the telco pass-through are shown as separate line items, so the customer's finance team can reconcile the telco portion against the carrier invoice independently rather than chasing it inside an opaque blended rate. There is no bundled-minutes trap: the customer is not paying a fixed monthly minute commitment that rolls over to nothing if unused, and is not penalised by overage rates if they exceed it. Usage is metered as it is consumed.

We are explicit that this is not per-second billing. The economics of voice AI — model inference cost, telco interconnect, recording storage — do not yet move at sub-minute granularity in any honest way, and we would rather charge in whole minutes than invent a per-second rate that obscures the underlying cost structure.

SECTION NINE

# What we built vs. what we stand on

A single-model speech-to-speech architecture is the thesis. It does not, by itself, get a phone call across the line. The work that gets a phone call across the line — auto-detecting telephony frame formats, controlling the model's turn boundaries from outside the model, surviving acoustic echo without false interruptions, hiding model cold-start behind ringing — is engineering we did, not architecture we inherited.

This section separates the two.

## 9.1 What we stand on

Three categories of components are not novel and we make no claim that they are:

| Component | What it provides |
|---|---|
| Frontier-lab multimodal speech model | A single, native audio-to-audio model is the substrate. We do not train it. We do not fork it. We use it as the realtime engine. |
| Open-source PBX with audio-streaming hook | A standard, widely-deployed open-source PBX provides SIP termination, dial planning, and a TCP audio-streaming hook into our gateway. |
| Standard infrastructure | Object storage for recordings. A relational database for agents, calls, and billing. A payment processor for India-domestic charges. None of this is differentiated and we do not claim it is. |

> *A buyer should be skeptical of any voice AI vendor that pretends to have built every layer themselves. Engineering judgement is partly about what not to rebuild.*

## 9.2 What we built — and have filed a patent on

What sits between the telephony layer and the model is a gateway we built. The non-obvious engineering inside that gateway — and the design decisions that make a single-model speech architecture actually run on an 8 kHz line in production — is the subject of a provisional patent we filed in April 2026 with the Indian Patent Office (GRX10 Solutions Private Limited, inventors Prem Kumar Arora and Damodaran Shanmugam).

The patent covers the bridge itself, in outcome terms:

| Bridge mechanism | What it solves |
|---|---|
| External voice-activity detector at native telephony bandwidth | The model is told when the caller starts and stops speaking by a detector that lives outside the model and is tuned to narrowband telephony audio. The model's built-in detector — trained on microphone audio — is bypassed entirely. Load-bearing decision behind every other choice in this section. |
| Frame-format auto-detection | Telephony streams arrive in multiple wire formats depending on the codec the carrier negotiates. Rather than configuring per trunk, the gateway classifies each incoming frame at the transport layer and decodes accordingly. New deployments and re-negotiated codecs work without operational intervention. |
| Asymmetric, echo-tolerant interrupt handling | Two different speech-detection threshold sets — permissive when the agent is silent, restrictive (with a longer required run-length) when the |

| | |
|---|---|
| | agent is producing output — plus a short configurable grace period after every agent turn that rejects the immediate echo tail. Result: callers can interrupt naturally; the agent never interrupts itself. |
| Pre-warmed first response | The model session, the system prompt, and the agent's first audio response are loaded into a buffer before the SIP INVITE is sent. The instant the call connects, the buffered greeting plays. First-response latency at pickup is bounded by telephony bridging time, not model inference time. This is what eliminates the conspicuous "AI dead air" at the front of every cascaded-stack call. |
| Continuous keepalive on the audio channel | Telephony channels with idle inactivity timeouts will tear themselves down during natural agent silence. The gateway emits format-correct silence frames at a fixed cadence shorter than the channel's inactivity timeout, in both directions, so the channel remains healthy through long pauses without leaking those frames into the model's input. |
| Prompt-audit gate before outbound dialing | No outbound call is initiated unless the agent's locked prompt version has passed a content-audit check. Result: an unsafe agent revision cannot dial, even if it is the active version. |

The patent describes claim language at the level of methods and systems; we do not reproduce the claim numbers in this document. The disclosed innovations together address three problems that are well-known to anyone who has tried to put a frontier speech model on the Indian phone network: the sample-rate mismatch between the model's native VAD and 8 kHz PSTN, the ambiguity of telephony frame formats, and the acoustic-echo false-interrupt failure mode that breaks otherwise capable models in production.

## 9.3 Why this matters for buyers

A technical buyer evaluating voice AI vendors should ask two questions:

1. Did you train the speech model? The honest answer for almost every vendor in the market — including us, including most of the platforms we compete with — is no. The frontier speech models are trained by labs with the data and compute to do so.

2. What did you build between the model and the phone line? This is where vendors differ, and where the work that determines whether a call sounds natural lives.

> *We have a defensible answer to the second question, and a filed patent that documents it. We do not claim a moat on the model. We claim a moat on the bridge.*

SECTION TEN

# What we do not do, and why

A serious technical document should also describe what is deliberately out of scope.

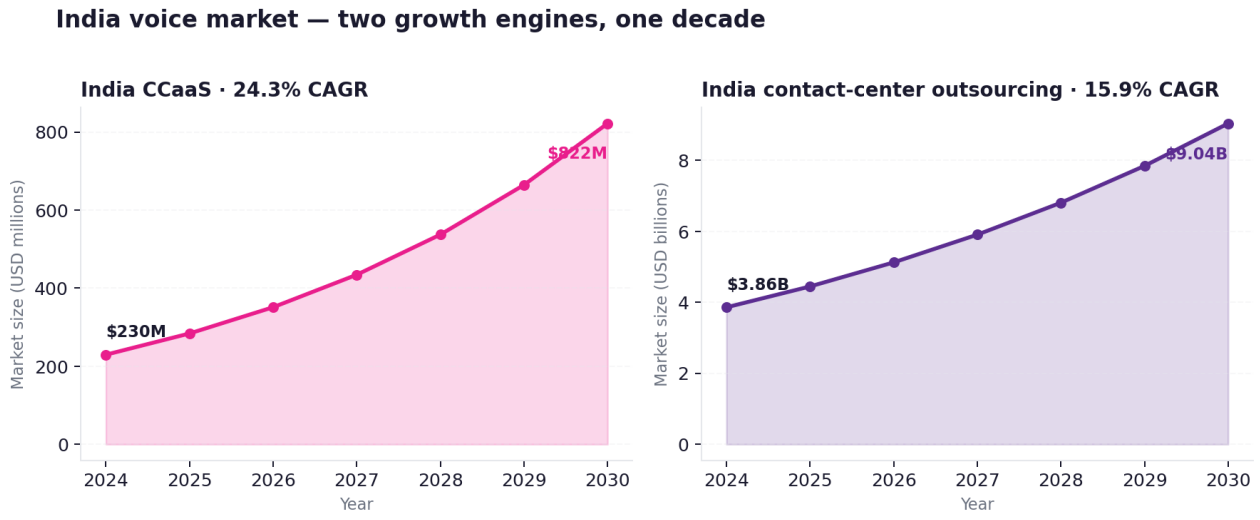
| Deliberately out of scope | Reason |
|---|---|
| No custom voice cloning at launch | The prebuilt voices in the underlying multimodal speech model are good enough for launch. Adding cloning means a dependency on a separate voice model, which breaks the "one model, one connection" guarantee that anchors our latency story. Smaller surface that works > larger surface that compromises the architecture. |
| No realtime in-call human-to-human translation | A different product surface — different VAD topology, different source/target mapping, different turn model. Worth building eventually; not at launch. |
| No sub-three-month feature catch-up to the underlying model's roadmap | The speech model evolves quickly. We track it; we do not fork it. Each new model capability becomes available to our customers within the upgrade cycle, not after a six-month re-engineering of our stack. |
| No on-prem, single-tenant deployment at launch | Will exist for enterprise on-request. Standard offering is multi-tenant cloud with India data-residency. |

> *The negative space is intentional. Voice AI is a market where most vendors ship every feature their largest customer asks for, and end up with a product that does many things badly. We would rather do fewer things well at launch and earn the right to expand.*

SECTION ELEVEN

# Why this matters now

The market moment is sharply defined. Three forces are converging.




Source: Grand View Research · India CCaaS and contact-center outsourcing outlooks, 2024–2030.

*Fig 5 · India voice market — two growth engines, one decade. CCaaS India growing at 24.3% CAGR; contact-center outsourcing at 15.9% CAGR. Source: Grand View Research.*

| Force | Implication |
|---|---|
| Multimodal speech models have crossed the production threshold | Until 2025, single-model speech-to-speech was a research demo. As of 2026, multiple frontier-lab speech models run real production workloads. The architectural debate is over; the deployment race is starting. |
| India's contact-center market is growing at 15.9% CAGR | CCaaS at 24.3% CAGR — from $229.5M in 2024 to a projected $821.7M by 2030. Voice is the largest revenue segment. Most of the growth is voice that has not yet been automated. |
| Global vendors discover India-specific complexity only after deployment | Narrowband audio, code-switching, regulatory load, accent variance — these are not features that appear on a US product roadmap. India-native platforms have a 24–36 month window to build a stack that is correct for the geography before global vendors close the gap. |

> *GRX10 Voice AI is built for that window. Single-model audio architecture, telephony-tuned voice detection, native code-switching, India-managed SIP, transparent per-minute billing with telco as a separate line item, OTP-gated launch safety. None of these is a roadmap item. They are the product as of day one.*

SECTION TWELVE

# How to evaluate this against a competitor

A short checklist for a technical buyer comparing platforms.

| Question | What a good answer sounds like |
|---|---|
| 1. Audio path — unified or cascaded? | If the vendor cannot answer, it is cascaded. |
| 2. Does the vendor own the realtime model session, or route to a third-party API? | Latency and observability follow from the answer. |
| 3. All-in per-minute cost on a representative Indian-language outbound campaign? | Including STT, LLM, TTS, telephony, recording storage, and any per-character or per-token overage. Get one number. Compare to the headline rate. |
| 4. How does the vendor handle a single sentence containing two languages? | Listen for "the user can configure the language" (does not handle code-switching) versus "the model handles it natively" (right answer). |
| 5. What does the platform do on built-in voice-detector failures at 8 kHz? | A vendor that has not encountered the question has not deployed at scale on Indian PSTN. |
| 6. Post-call transcript latency and format? Does live transcription degrade audio? | Right answer: "we transcribe post-call to protect live audio quality." |
| 7. What happens to in-flight calls when a prompt is edited? | If the vendor cannot answer, agent prompts are not versioned. |
| 8. How does the bundled-minutes plan work — if there is one? | A fixed monthly minute commitment that does not roll over and overage-bills above the cap is a smoothing mechanism for the vendor's revenue, not a pricing model for the buyer. Vendors that bill straight per-minute usage at a published rate are easier to reason about. |

> *Most vendors will fail at least four of these. That is not because they are dishonest. It is because the cascaded architecture cannot answer some of them well, and the BYO operational model cannot answer others.*

SECTION THIRTEEN

# Closing

Voice AI is at the architectural inflection point that text models passed in 2022 and image models passed in 2023. The cascaded stack will continue to ship for two more years on momentum, the way SQL replaced punch cards on momentum. The unified single-model architecture will replace it because the latency math, the prosodic math, and the operational math all point the same direction.

We have built GRX10 Voice AI on the right side of that transition, for a market — India — where the architectural advantage compounds with regulatory, linguistic, and economic advantages that the global vendors cannot easily copy. The technical work that gets a phone call to feel like a phone call, in Hinglish, on an 8 kHz line, in under a second, is not glamorous. It is the work.

> *If you are evaluating voice AI for India, the right question is not which vendor has the best demo. It is which vendor has built the architecture that matches the geography. We are happy to walk through ours in detail, under NDA, with whatever depth your engineering team requires.*

ABOUT GRX10

# Who built this

GRX10 is an Indian deep-tech company building AI infrastructure for voice, language, and lead-management workloads in the Indian and emerging markets. Voice AI is its production voice agent platform.

## Contact

**prem@grx10.com** · **www.grx10.com** · Bengaluru, Karnataka, India

## Patent

A provisional specification covering the telephony-bridge methods and systems described in Section 9 was filed by GRX10 Solutions Private Limited with the Indian Patent Office in April 2026. Inventors: Prem Kumar Arora and Damodaran Shanmugam. Patent pending.

## Sources and disclosures

This document is informational and does not describe specific implementation details, third-party dependencies, model versions, or carrier relationships. All competitor references are anonymised by initial. Architectural and latency claims are theoretical and based on published benchmarks and documented architectural patterns; production performance varies with network, carrier, and deployment configuration. Pilot references describe sector and use-case shape only.