# Ringmaster Voice Engine: Flows & Actions Reference

> The single action system used by **inbound numbers**, **campaigns** and (Phase 3) **AI**. Companion to `SPEC.md` and `ARCHITECTURE.md`. **[VERIFY]** = confirm with Africa's Talking (AT) in the PoC.

---

# 1. Concepts

A **flow** is a small graph of **nodes**. Each node does one thing and has named **outcomes** that point to the next node (or `end`).

| Runs in | Starts when | Flows |
|---|---|---|
| Inbound number | A call arrives | `open` flow + `closed` flow (chosen by business hours) |
| Campaign | The recipient answers | One campaign flow |
| AI (Ph 3) | AI hands over / calls a tool | Uses the same `connect`, `webhook`, `sms` actions |

Customers normally don't edit nodes. Textify templates generate flows:

- **Direct**: `play greeting → connect department → voicemail`
- **Menu**: `play greeting → menu → connect department(s) → voicemail`
- **AI** (Ph 3): `ai → connect department → voicemail`
- **Campaign screen**: `menu(message + keys) → actions`

An advanced flow editor may come later; this format already supports it.

## 1.1 How AT actions map

| AT action | Used for |
|---|---|
| `Play` | Every message callers hear (uploaded audio, generated text, hold music) |
| `GetDigits` | Menus, collecting digits, keys during waiting |

| AT action | Used for |
|---|---|
| `Record` | Voicemail |
| `Dial` | Anything **live**: into LiveKit (`sip:{callId}@…`) for agents (and AI in Ph 3) |
| `Say` | **Not used** (predictable cost, consistent voice) |

Rule: **one-way listening =** `Play`, **keys =** `GetDigits`, **live two-way =** `Dial` **→ LiveKit**.

---

# 2. Message (what callers hear)

One shape everywhere:

```
{ "source": "audio", "media": "med_123" }
{ "source": "text",  "text": "Karibu {name}", "lang": "sw", "voice": "female", "tier": "basic" }
```

- `audio`: an uploaded/recorded file from the audio library (must be `READY`).
- `text`: turned into an audio file **when the flow is saved** (or **before dialing** for campaigns with contact variables). Never synthesized during a live call.
  - `tier`: `basic` (self-hosted light model, fixed cost) or `premium` (OpenAI/Azure, cost shown before saving).
  - Cached by `hash(renderedText + lang + voice + tier)`.
  - Swahili normalization first: numbers, currency, dates, times to words.
  - Max 500 characters.
- At call time the engine renders `<Play url="https://voice-api…/media/{id}.mp3?t=…"/>`.
- Safety net: if a file is missing at call time → play system file `sys_sorry_try_later` and hang up; alert.

---

# 3. Variables

Written as `{name}` inside message text and webhook params.

| Group | Variables | Available |
|---|---|---|
| System | `{caller}`, `{number}`, `{callId}`, `{date}`, `{time}`, `{key}` (last key), `{businessName}` | Everywhere |
| Contact | Any recipient column: `{name}`, `{amount}`, `{appointment_id}`… | Campaign flows only |

| Group | Variables | Available |
| --- | --- | --- |
| Collected | `{input.<saveAs>}` from `collect` | After that node |
| Webhook result | `{result.<field>}` from a **sync** `webhook` | After that node (Ph 2) |

Missing variable at runtime → empty string + warning in the call trail (validation prevents most cases).

---

# 4. Node types (complete list)

Common fields: `type`, `on` (outcome → node id or `"end"`), optional `label` (for stats).

## 4.1 `play`

Play a message.

```
{ "type": "play", "message": {…}, "on": { "next": "menu" } }
```

Outcomes: `next`. XML: `Play`.

## 4.2 `menu`

Message + wait for **one key**.

```
{ "type": "menu", "message": {…},
  "timeoutSec": 8, "retries": 2,
  "invalidMessage": { "source": "audio", "media": "sys_invalid_option" },
  "on": { "1": "sales", "2": "support", "0": "menu", "noInput": "support", "invalid":
"support" } }
```

Outcomes: keys `0–9`, `*`, `#`, `noInput`, `invalid` (after retries). XML: `GetDigits numDigits=1` wrapping `Play`.

## 4.3 `collect`

Ask for **several digits** and save them.

```
{ "type": "collect", "message": {…}, "saveAs": "account",
  "minDigits": 6, "maxDigits": 10, "finishOnKey": "#", "pattern": "^[0-9]{6,10}$",
```

```
  "timeoutSec": 10, "retries": 2,
  "on": { "ok": "lookup", "failed": "agent" } }
```

Outcomes: `ok`, `failed`. XML: `GetDigits`. Saved as `{input.account}`.

# 4.4 `connect`

Put the caller through **live**.

```
{ "type": "connect",
  "target": { "department": "dep_support" },        // or {"agent": "agentRef"} | {"external":
"+2557…"} | {"ai": {…}} (Ph 3)
  "ringTimeoutSec": 30,
  "wait": {
    "music": { "source": "audio", "media": "sys_hold_default" },
    "comfortEverySec": 45,
    "comfortMessage": { "source": "text", "text": "Tafadhali endelea kusubiri, mhudumu
atakupokea punde.", "tier": "basic" },
    "maxWaitSec": 300,
    "keys": { "1": "voicemail" },
    "announcePosition": false,     // later
    "announceWaitTime": false,     // later
    "callbackKey": null            // later: e.g. "2" = request a callback
  },
  "whisper": null,                  // later: message only the agent hears
  "record": true,
  "on": { "noAnswer": "vm", "voicemail": "vm", "callback": "cb_ok" } }
```

(The comfort message reads: "Please keep waiting, a staff member will answer you shortly.")

Outcomes: `noAnswer` (max wait reached / nobody available), plus any outcome named in `wait.keys`. Answered = the call is live; the flow ends there.

How it runs: see §6 "Waiting loop". `external` (forward to a mobile) is allowed but off by default (extra outbound leg cost).

# 4.5 `voicemail`

```
{ "type": "voicemail", "message": {…}, "maxSec": 60, "beep": true, "on": { "next": "end" } }
```

XML: `Play` + `Record`. Emits `voicemail.received`.

# 4.6 `webhook`

Call the customer's saved endpoint.

```
{ "type": "webhook", "endpoint": "ep_clinic", "mode": "async",
  "params": { "event": "appointment_confirmed", "id": "{appointment_id}", "phone": "{caller}",
"key": "{key}" },
  "on": { "next": "thanks" } }
```

- `async` (Ph 1): queued in the outbox, retried up to 24 h; outcome `next` immediately.
- `sync` (Ph 2): wait ≤ `timeoutMs` (default 3000). Response JSON fields become `{result.*}`; an optional `"say"` field is played next (generated on demand, then cached). Outcomes `success`, `failure` (timeout / non-2xx).
- Request: `POST` JSON `{event params…, meta: {callId, campaignId, accountRef, node, timestamp}}`, headers `X-Voice-Signature` (HMAC with the endpoint secret), `X-Voice-Idempotency-Key` (= callId + node).
- HTTPS only; private/internal IP ranges blocked (SSRF); never breaks the call.
- Emits `call.action_triggered` to Textify (for logs/dashboard).

# 4.7 `branch`

Decide by condition (engine-side, instant).

```
{ "type": "branch",
  "cases": [
    { "if": { "hours": "closed" }, "go": "closedMsg" },
    { "if": { "var": "result.status", "equals": "paid" }, "go": "paidMsg" },
    { "if": { "var": "input.account", "matches": "^99" }, "go": "vip" }
  ],
  "on": { "default": "menu" } }
```

Operators: `equals`, `in`, `matches` (regex), `exists`, `hours: open|closed`. First match wins.

# 4.8 `sms`

Ask Textify to send an SMS/WhatsApp to the caller ("we've sent you the payment link").

```
{ "type": "sms", "channel": "sms", "template": "payment_link", "params": { "amount":
"{amount}" },
  "on": { "next": "thanks" } }
```

Engine emits `message.requested` → **Textify sends and bills** (it owns messaging and templates).

## 4.9 `optout`

Add `{caller}` to the account's do-not-call list. Outcome `next`.

## 4.10 `hangup`

Optional goodbye message, then end.

```
{ "type": "hangup", "message": { "source": "text", "text": "Asante, kwaheri." } }
```

(The goodbye reads: "Thank you, goodbye.")

## 4.11 `ai` (Phase 3)

```
{ "type": "ai",
  "instructions": "Jibu maswali kuhusu bei na saa za kazi.",
  "knowledgeBase": true,
  "tools": { "endpoints": ["ep_orders"], "sms": ["payment_link"] },
  "handoff": { "department": "dep_support" },
  "maxMinutes": 10,
  "on": { "handoff": "agent", "ended": "end" } }
```

(The instructions read: "Answer questions about prices and working hours.")

Runs via `Dial` → LiveKit → OpenAI Realtime (ARCHITECTURE Appendix A). The AI uses the account knowledge base, can call endpoints/sms as tools, and hands over to a department (`handoff` outcome → usually a `connect`).

---

# 5. Examples

## 5.1 Menu answer mode (generated by the template)

```
{
  "version": 7,
```

```
  "settings": { "record": true, "recordingNotice": true },
  "open": {
    "start": "greet",
    "nodes": {
      "greet": { "type": "play", "message": { "source": "audio", "media": "med_karibu" },
"on": { "next": "menu" } },
      "menu":  { "type": "menu", "message": { "source": "text", "text": "Bonyeza 1 mauzo, 2
huduma kwa wateja.", "tier": "basic" },
                 "on": { "1": "sales", "2": "support", "noInput": "support", "invalid":
"support" } },
      "sales":   { "type": "connect", "target": { "department": "dep_sales" },   "wait": {
"music": { "source": "audio", "media": "sys_hold_default" }, "maxWaitSec": 180 }, "on": {
"noAnswer": "vm" } },
      "support": { "type": "connect", "target": { "department": "dep_support" }, "wait": {
"music": { "source": "audio", "media": "sys_hold_default" }, "maxWaitSec": 300, "keys": { "1":
"voicemail" } }, "on": { "noAnswer": "vm", "voicemail": "vm" } },
      "vm": { "type": "voicemail", "message": { "source": "text", "text": "Acha ujumbe baada
ya mlio.", "tier": "basic" } }
    }
  },
  "closed": {
    "start": "closedMsg",
    "nodes": {
      "closedMsg": { "type": "play", "message": { "source": "text", "text": "Tumefunga kwa
sasa. Tunafungua saa mbili asubuhi.", "tier": "basic" }, "on": { "next": "vm" } },
      "vm": { "type": "voicemail", "message": { "source": "text", "text": "Acha ujumbe baada
ya mlio.", "tier": "basic" } }
    }
  }
}
```

(The Swahili messages read: "Press 1 for sales, 2 for customer service." · "Leave a message after the tone." · "We're closed right now. We open at eight in the morning." Tanzanian time "saa mbili asubuhi" is 08:00.)

# 5.2 Campaign: appointment reminder

```
{
  "start": "msg",
  "nodes": {
```

```
    "msg": { "type": "menu",
             "message": { "source": "text", "text": "Habari {name}, una miadi kesho saa
{time_slot}. Bonyeza 1 kuthibitisha, 2 kuongea na mhudumu, 3 kupata ujumbe wa SMS, 9
kujiondoa.", "tier": "premium" },
             "on": { "1": "confirm", "2": "agent", "3": "sms", "9": "out", "noInput": "msg",
"invalid": "msg" } },
    "confirm": { "type": "webhook", "endpoint": "ep_clinic", "mode": "async",
                 "params": { "event": "confirmed", "id": "{appointment_id}", "phone":
"{caller}" }, "on": { "next": "thanks" } },
    "thanks": { "type": "hangup", "message": { "source": "text", "text": "Asante,
tumethibitisha miadi yako.", "tier": "basic" } },
    "agent":  { "type": "connect", "target": { "department": "dep_reception" }, "wait": {
"music": { "source": "audio", "media": "sys_hold_default" }, "maxWaitSec": 120 }, "on": {
"noAnswer": "vm" } },
    "vm":     { "type": "voicemail", "message": { "source": "audio", "media": "med_vm" } },
    "sms":    { "type": "sms", "channel": "sms", "template": "appointment_details", "params":
{ "id": "{appointment_id}" }, "on": { "next": "smsOk" } },
    "smsOk":  { "type": "hangup", "message": { "source": "text", "text": "Tumekutumia ujumbe.
Kwaheri.", "tier": "basic" } },
    "out":    { "type": "optout", "on": { "next": "bye" } },
    "bye":    { "type": "hangup", "message": { "source": "text", "text": "Umeondolewa kwenye
orodha. Kwaheri.", "tier": "basic" } }
  }
}
```

(The Swahili messages read: "Hello {name}, you have an appointment tomorrow at {time_slot}. Press 1 to confirm, 2 to talk to a staff member, 3 to get an SMS, 9 to opt out." · "Thank you, we've confirmed your appointment." · "We've sent you a message. Goodbye." · "You've been removed from the list. Goodbye.")

---

# 6. Waiting loop (`connect` to a department)

Waiting happens **on AT**, not in LiveKit. Reasons: keys work while waiting, no trunk cost while waiting, announcements are just files.

```
connect node:
  1. engine marks call WAITING_AGENT, adds it to the department's waiting order (oldest first)
```

```
  2. rings available agents in their browsers (LiveKit lobby data message)
  3. returns XML:  <GetDigits numDigits="1" timeout="1" callbackUrl=".../wait/{callId}">
                      <Play url="{hold segment ~10–15 s}"/>
                   </GetDigits>
AT calls back after each segment (or on a key):
  - agent accepted?       → <Dial phoneNumbers="sip:{callId}@sip.voice…" record="true"/>   → live
  - key pressed?          → outcome from wait.keys (e.g. voicemail)
  - waited ≥ maxWaitSec?  → outcome noAnswer
  - else                  → next segment: music, or comfortMessage every comfortEverySec
```

- Agent accepts mid-segment → their screen shows "Connecting caller…" until the segment ends (≤ ~15 s; keep segments short).
- Agent becomes available (or finishes a call) → engine rings them with the oldest waiting caller.
- Later features slot into the same loop: position announcement (pre-generated number files), estimated wait, callback request, whisper, wrap-up time.
- **[VERIFY]** GetDigits-wrapping-Play interrupt behaviour, timeouts, callback count limits. Fallback if AT can't do this: dial into LiveKit immediately and play hold music via LiveKit Ingress (no keys while waiting).

In-call hold/transfer by the agent happens inside LiveKit (hold music via LiveKit Ingress).

---

# 7. Validation rules (on save; errors shown in Textify, no approvals)

1. Every outcome points to an existing node or `end`; `start` exists.
2. ≤ 50 nodes per flow; menus ≤ 12 options; ≤ 3 retries.
3. Variables exist in scope (contact columns only in campaigns; `{input.x}` only after its `collect`; `{result.x}` only after a sync `webhook`).
4. `connect` targets, endpoints, media, sms templates belong to the same account; media `READY`; text generation succeeded.
5. Message text ≤ 500 characters.
6. Inbound numbers must have both `open` and `closed` flows (template provides defaults).
7. `ai` nodes only when Phase 3 is enabled for the account.

Runtime guards: max **25 node visits per call** (loop guard → system goodbye + hangup); max call length per flow (default 30 min).

---

# 8. Versioning

- Every save → new immutable version (`flow_versions`). The number points to its current version.
- Calls store the version they ran.
- **Running campaigns are pinned** to the version they started with; edits apply to new campaigns only.
- Rollback = point the number to an older version.

---

# 9. Execution (engine)

- Each AT callback URL carries `callId` + node: `/hooks/at/{secret}/step/{callId}/{nodeId}` (and `/wait/{callId}` for the waiting loop).
- Call **context** (variables, retries, visit count) stored as JSON on the call row; no in-memory sessions.
- One small Java class per node type: `NodeExecutor.execute(node, ctx) → Xml | NextNode`. Engine-side nodes (`webhook` async, `branch`, `sms`, `optout`) run instantly and continue to the next node within the same callback.
- **Never fail the caller**: any exception → `Play sys_sorry_try_later` + hang up; log + alert. Never return 5xx to AT.
- **Call trail**: each node visited, key pressed and action result is recorded in `call_events` → per-node stats ("60% pressed 1, 12% pressed 9").
- **Test call**: `POST /v1/numbers/{id}/test-call {toE164}` or `/campaigns/{id}/test-call` runs the flow against the customer's own phone (contact variables from a sample row).

---

# 10. Cost per node

| Node | Cost |
|---|---|
| `play`, `menu`, `collect`, `voicemail`, `hangup` | Call minutes only (on AT) |
| Text messages | Basic: free · Premium: characters × rate, known at save / campaign creation |
| `connect` (waiting) | Call minutes only (still on AT) |
| `connect` (talking) | Minutes + SIP trunk 8.61/min; `external` = extra outbound leg |
| `webhook`, `branch`, `optout` | Free |
| `sms` | Textify's SMS/WhatsApp price |
| `ai` | AI per-minute price (Ph 3) |

# 11. Phase availability

| Feature | Phase |
| --- | --- |
| `play`, `menu`, `collect`, `connect` (department/agent), `voicemail`, `webhook` async, `branch`, `sms`, `optout`, `hangup` | 1 |
| Waiting: music, comfort message, max wait, keys while waiting | 1 |
| Basic voice tier | 1 |
| Premium voice, `webhook` sync, campaigns | 2 |
| `ai` node, knowledge base, AI tools | 3 |
| Position / wait-time announcements, callback request, whisper, wrap-up, `connect.external` on by default, advanced flow editor | Later |

Revision #2
Created 8 November 2026 20:57:56 by Admin Qbit
Updated 13 November 2026 19:07:23 by Admin Qbit