Part 3 — What the engine sends you

← Contents · Previous: Part 2E — App actions · Next: Part 4 — Reference

The engine talks to the client app in three ways:

Channel	Direction	Used for
Webhooks	engine → client app's webhook URL (POST)	Everything that happened: calls, blocks, recordings, call-outs, agents, media, app actions
The can-call check	engine → client app's can-call URL (GET), waits for the answer	Ask before a call: is this account allowed to call now? (optional)
Agent desk signals	engine → the agent's browser, through LiveKit	Ring, ring cancelled, connecting

Both URLs and the shared secret are set by the Ringmaster owner for the client app (Appendix).

3.1 Webhooks

Every webhook

```
POST https://app.example.co.tz/hooks/ringmaster
Content-Type: application/json
X-Ringmaster-Event: call.ended
X-Ringmaster-Delivery: f267e572-f9d2-4a8d-831a-a3bec700cd44
X-Ringmaster-Timestamp: 1794588244
X-Ringmaster-Signature: 5c1e9a0d3f…   (64 hex characters)

{
  "id": "f267e572-f9d2-4a8d-831a-a3bec700cd44",
  "type": "call.ended",
  "createdAt": "2026-11-13T16:59:31.053Z",
```

```
  "data": { … }
}
```

Part	Description
`X-Ringmaster-Event`	Same as `type`
`X-Ringmaster-Delivery`	Same as `id`; unique per event, also on retries — **drop repeats by this id**
`X-Ringmaster-Timestamp`	Unix seconds when this attempt was sent
`X-Ringmaster-Signature`	Lower-case hex HMAC-SHA256 of `"<timestamp>.<raw body>"` with the webhook secret
`createdAt`	When the event happened (attempts later keep it)
`data`	The event's content (below). Every event about an account carries `accountRef` (the account's `externalRef`) — except `call.block` and `call.scheduled`, which carry `callId`

Checking the signature

Check it on the **raw body bytes**, before parsing; refuse timestamps more than 5 minutes off.

```
import hmac, hashlib, time

def valid(secret: str, headers, raw_body: bytes) -> bool:
    ts = headers["X-Ringmaster-Timestamp"]
    if abs(time.time() - int(ts)) > 300:
        return False
    expected = hmac.new(secret.encode(), ts.encode() + b"." + raw_body,
hashlib.sha256).hexdigest()
    return hmac.compare_digest(expected, headers["X-Ringmaster-Signature"])
```

```
String expected = HexFormat.of().formatHex(Mac.getInstance("HmacSHA256") {{ init(new
SecretKeySpec(secret.getBytes(UTF_8), "HmacSHA256")); }}
        .doFinal((timestamp + "." + rawBody).getBytes(UTF_8)));
boolean ok = MessageDigest.isEqual(expected.getBytes(UTF_8), signature.getBytes(UTF_8));
```

Answering, retries, order

- Answer **2xx within a few seconds**, then do the work (queue it). Any other answer, or no answer, is retried: **1 min, 5 min, 30 min, 2 h, 6 h, 12 h**, then every 12 h, until **24 h** after the event; after that the event is marked failed and not sent again.
- Events are stored in the same database step as the change they report, so none is lost when the engine restarts.
- **Order is not guaranteed** across retries: `call.ended` can arrive before a late `call.block`. Use `createdAt`, and treat each event as "the latest I know", not as a step in a strict sequence.
- The same event can arrive twice (a 2xx that got lost on the way back). Keep the `id`s you processed.
- The owner can send a `test.ping` (`ringmaster-admin client-app test-webhook <app>`) to check the URL and the secret.

Events

Type	When	Main `data`
`call.started`	A call begins (incoming picked up, or dialling out)	callId, type, direction, from, to
`call.answered`	The person answered a call-out / OTP / agent call	same as `call.started`
`call.block`	A caller's choice or answer worth acting on (menu, collect, poll, pick, ask, request, app action)	callId, block, type, …
`call.ended`	The call is over	the full call summary
`call.missed`	A caller gave up waiting, or nobody answered them	callId, from, to, result
`call.scheduled`	A caller booked a callback	callId, calloutId, at, to, said
`voicemail.received`	A voicemail is stored	callId, recordingId, durationSec, from
`recording.ready`	A recorded live part (agent talk) is stored	same
`summary.ready`	Transcript and summary of a recording are ready	callId, summary, sentiment, language, fields, unanswered
`ai.unanswered`	The AI could not answer some caller questions	callId, questions
`action.requested`	A flow asks the client app to do one of its actions	actionId, name, callId, params, wait, timeoutAt
`request.failed`	A Send request delivery kept failing for 24 h	callId, block, attempts, status, error
`callout.progress`	Every 30 s while a call-out runs	counts
`callout.paused`	The engine paused a call-out	counts + reason

Type	When	Main `data`
`callout.completed`	Everybody was called (or it was stopped)	counts
`callout.recipient_results`	Batches of people who are done	results[]
`agent.status`	An agent's status or presence changed	agentRef, status, online, departments
`media.ready` / `media.failed`	An uploaded audio file finished processing	mediaId, durationMs / error
`test.ping`	Sent by the owner to test the setup	—

The samples below are real events, shortened only where marked.

`call.started`


```
{ "id": "32c49361-fa61-441c-9c6d-9fcb0add6227", "type": "call.started", "createdAt": "2026-11-13T16:58:37.622Z",
  "data": { "callId": "4e2e8a2d-4a1e-487e-b684-bd7e5cfee244", "accountRef": "softphone", "type": "INCOMING",
            "direction": "INBOUND", "from": "+255745051250", "to": "+255677000001" } }
```


Use it for a "live calls" screen or a screen pop (look up `from` in the client app's contacts).

`call.answered`


```
{ "id": "c8365f5d-d368-47b9-a6d2-910ba5e9523e", "type": "call.answered", "createdAt": "2026-11-13T14:41:06.810Z",
  "data": { "callId": "fff4aa3f-8ed5-44e6-947c-c82bb3628c41", "accountRef": "softphone", "type": "BROADCAST",
            "direction": "OUTBOUND", "from": "+255677000003", "to": "+255745051250" } }
```


Only for outgoing calls (incoming calls are answered by the engine at once).

`call.block`

Sent when a caller does something the client app may act on during the call. `data.type` says which block:

`type`	Extra fields	Example

`menu`	`option` (the option's id), `by` (`key` / `speech`)	`{"callId": "763f…", "block": "main", "type": "menu", "option": "jitoe", "by": "key"}`
`collect`	`saveAs`, `value` (`••••` when the block is `sensitive`)	`{"block": "acct", "type": "collect", "saveAs": "account", "value": "1234"}`
`poll`	`question` (saveAs), `answer` (key), `label`, `by`	`{"block": "rate", "type": "poll", "question": "rating", "answer": "5", "by": "key"}`
`pick`	`id`, `label` of the chosen item	`{"block": "product", "type": "pick", "id": "acetic", "label": "Acetic Acid"}`
`ask`	`saveAs`, `text` (the transcribed answer)	`{"block": "problem", "type": "ask", "saveAs": "problem", "text": "Simu yangu haiwaki"}`
`request`	`outcome` (the exit taken), `status` (HTTP); late callbacks add `late: true`, `reply`	`{"block": "look", "type": "request", "outcome": "success", "status": 200}`
`app_action`	`name`, `outcome`; `late: true` when the answer came after the call moved on	`{"block": "pay_1", "type": "app_action", "name": "pay", "outcome": "paid"}`


```
{ "id": "a8156b02-36c6-4484-bb04-8f8b61b723c7", "type": "call.block", "createdAt": "2026-11-13T16:18:05.966Z",
  "data": { "callId": "763f750d-cf31-4265-a03c-7540627d55df", "block": "main", "type": "menu", "option": "jitoe", "by": "key" } }
```


`call.ended`

`data` is the full call summary, exactly as `GET /v1/calls/{callId}` returns it (Part 2C, G2).


```
{ "id": "f267e572-f9d2-4a8d-831a-a3bec700cd44", "type": "call.ended", "createdAt": "2026-11-13T16:59:31.053Z",
  "data": {
    "callId": "4e2e8a2d-4a1e-487e-b684-bd7e5cfee244", "accountRef": "softphone", "accountId": "adb72fac-66a2-471b-a437-035fe008197d",
    "type": "INCOMING", "direction": "INBOUND", "from": "+255745051250", "to": "+255677000001",
    "status": "ENDED", "result": "HANDLED_BY_AGENT", "endReason": "AGENT_HUNG_UP", "endedBy": "AGENT", "handledBy": "AGENT",
    "startedAt": "2026-11-13T16:58:37.599172Z", "answeredAt": "2026-11-13T16:58:37.599172Z", "endedAt": "2026-11-13T16:59:31.034198254Z",
    "durations": { "ringSeconds": null, "totalSeconds": 54, "scriptSeconds": 1, "waitSeconds": 16, "talkSeconds": 35, "holdSeconds": 0, "aiSeconds": 0 },
```

```
    "keys": null, "lastBlock": "desk", "outcomes": [],
    "usage": { "pstnSeconds": 54, "trunkSeconds": 54, "waitSeconds": 16, "aiSeconds": 0,
"ttsChars": 0, "sttSeconds": 0 },
    "providerCost": { "tzs": 25.61, "usd": 0.0, "items": [ { "provider": "AT", "tzs": 25.61,
"usd": 0.0, "lines": [ "…" ] } ] },
    "recorded": false,
    "script": { "versionId": "e183c7f1-3137-4848-95c1-11a7e8bef17a", "version": 1, "ref":
"desk-test" },
    "calloutId": null, "hangupCause": null, "sipStatus": null, "meta": null } }
```


`recorded: true` means a `voicemail.received` / `recording.ready` (and later `summary.ready`) is coming or came.

`call.missed`

A caller waited for an agent and left (`ABANDONED`) or nobody answered within the waiting room's give-up time (`MISSED`). Use it for a "call back" list.


```
{ "id": "07c440f3-50b1-4918-8338-cfd2b8e17bf2", "type": "call.missed", "createdAt": "2026-11-
13T16:28:17.449Z",
  "data": { "callId": "44f41241-e356-46d1-86aa-dcd4f0f82c12", "accountRef": "softphone",
            "from": "+255745051250", "to": "+255677000001", "result": "ABANDONED" } }
```


`call.scheduled`

A caller asked to be called later (Schedule call block or the AI's `schedule_call`) and a one-person call-out was booked.


```
{ "id": "75974f9d-dc08-4fcf-834d-dea89c3bb037", "type": "call.scheduled", "createdAt": "2026-
11-13T14:36:38.302Z",
  "data": { "callId": "ca10b2f7-a5a3-41bb-be8b-2e486b2b5dbf", "calloutId": "510fca7b-b226-
46ad-b80a-0082ab4ab0d5",
            "at": "2026-11-13T17:41+03:00", "to": "+255745051250", "scriptRef": "account-
callback",
            "said": "After five minutes, baada ya dakika tano." } }
```


Field	Description
`at`	Booked time, with the account's offset
`scriptRef`	The Outgoing flow that will play

Field	Description
`said`	The caller's words (absent for a fixed time)
`calloutId`	Show or cancel it with the call-out API (Part 2C)

A new booking by the same caller for the same flow replaces the old one (the old call-out is cancelled).

`voicemail.received` / `recording.ready`

```
{ "id": "3a73b6ee-36f4-4c4e-82ef-599145ed63a1", "type": "voicemail.received", "createdAt":
"2026-11-13T16:46:20.647Z",
  "data": { "callId": "355760f8-51c0-47c8-b3b0-49c3612e4b96", "accountRef": "softphone",
            "recordingId": "3eb839d7-bac0-49da-a870-7aba6fa77b6e", "kind": "VOICEMAIL",
"durationSec": 11, "from": "+255745051250" } }
```

`recording.ready` has the same shape with `kind: "CALL"` (the recorded talk with an agent). Fetch the audio with `GET /v1/calls/{callId}/recording`.

`summary.ready`

```
{ "id": "0f86f71d-8bec-4aff-a634-2fca99f4c3a0", "type": "summary.ready", "createdAt": "2026-
11-13T16:46:27.191Z",
  "data": { "callId": "355760f8-51c0-47c8-b3b0-49c3612e4b96", "accountRef": "softphone",
"language": "sw",
            "summary": "Mpigaji amesema huo ndio ujumbe wake na ametoa shukrani nyingi.",
"sentiment": "positive",
            "fields": {}, "unanswered": [] } }
```

The full text: `GET /v1/calls/{callId}/transcript`.

`ai.unanswered`

Questions callers asked an AI Agent that its knowledge did not answer — material for the business's knowledge base.

```
{ "id": "2099a5dd-ceab-4609-b32b-a167ea91d39e", "type": "ai.unanswered", "createdAt": "2026-
11-13T09:30:36.154Z",
  "data": { "callId": "d6f2ef46-2775-4696-8cac-e4e1e9e2133a", "accountRef": "softphone",
            "questions": ["Je, miadi ya kesho saa 1 usiku inaweza kupangwa?", "Amini,
```

```
wanipi."] } }
```

`action.requested`

A flow (App action block, AI tool) or a campaign fallback asks the client app to run one of its registered actions. Answer it with `POST /v1/calls/{callId}/action-result` (Part 2E, K3).

```
{ "id": "9c1d…", "type": "action.requested", "createdAt": "2026-11-20T09:12:30Z",
  "data": { "actionId": "c3a9e1f0-…", "name": "pay", "callId": "4b1e…", "accountRef": "duka-juma", "block": "pay_1",
            "params": { "amount": "45000", "phone": "+255712345678", "wallet": "M-Pesa" },
            "wait": true, "timeoutAt": "2026-11-20T09:14:30Z",
            "meta": { "client": { "orderId": "ORD-88" } } } }
```

Field	Description
`params`	Filled in: templates replaced by the call's values (`{caller}` → the number)
`wait` / `timeoutAt`	`true`: the caller is waiting; answer before `timeoutAt` or the flow takes its `timeout` exit
`meta.client`	The call's `meta` (from the call-out person, OTP request…)

A **campaign fallback** (after a person's last failed try) has no call to wait on: `data` is `{actionId, name, params, wait: false}`, and `params` also carries `to` (the person), `calloutId`, `accountRef` and `reason` (their last result, e.g. `NO_ANSWER`). Nothing waits for an answer.

`request.failed`

A Send request block in "send and continue" mode delivers in the background with retries (30 s, 2 min, 10 min, 30 min, 1 h, 2 h, 4 h…). After 24 h of failures it gives up and tells the client app:

```
{ "id": "…", "type": "request.failed", "createdAt": "2026-11-14T09:12:30Z",
  "data": { "callId": "4b1e…", "block": "notify_crm", "attempts": 9, "status": 503, "error": "Service Unavailable" } }
```

`callout.progress` · `callout.completed` · `callout.paused`

Same shape: the call-out's counts. `progress` every 30 s while it runs; `completed` once at the end (also after `stop`); `paused` whenever it is paused (by you or by the engine), with `reason`.

```
{ "id": "2d0dd47c-4f31-4b9f-a523-b3cd7e486f33", "type": "callout.completed", "createdAt":
"2026-11-13T14:41:34.770Z",
  "data": { "calloutId": "510fca7b-b226-46ad-b80a-0082ab4ab0d5", "externalRef":
"schedule:account-callback:+255745051250",
            "status": "COMPLETED", "total": 1, "byStatus": { "DONE": 1 }, "byResult": {
"ANSWERED": 1 }, "byOutcome": {},
            "meta": { "client": { "said": "After five minutes, baada ya dakika tano.",
"scheduledBy": "ca10b2f7-a5a3-41bb-be8b-2e486b2b5dbf" } } } }
```


`reason` (paused)	Why
`ACCOUNT_SUSPENDED` / `CLIENT_SUSPENDED`	The account or the client app was suspended
`VERSION_REVOKED`	Its flow version was stopped
`FEATURE_OFF`	The owner took the client app's `CALLOUTS` feature away
`CAN_CALL:<reason>`	The can-call check said no (3.2)
`PAUSED_BY_CLIENT` / your own text	You paused it (`POST …/pause`); the webhook is sent for these too

The engine never resumes a call-out it paused.

`callout.recipient_results`

People who are finished, in batches (about every 30 s), so the client app can update its contacts.


```
{ "id": "50c42336-a708-4289-a307-6a4dd6ceefae", "type": "callout.recipient_results",
"createdAt": "2026-11-13T14:42:44.692Z",
  "data": { "calloutId": "510fca7b-b226-46ad-b80a-0082ab4ab0d5",
            "results": [ { "e164": "+255745051250", "result": "ANSWERED", "attempts": 1,
"callId": "fff4aa3f-8ed5-44e6-947c-c82bb3628c41",
                          "answeredSeconds": 28, "keys": "1234", "outcomes": null } ] } }
```


`agent.status`

Every status or presence change, for a wallboard.


```
{ "id": "19908d3b-3d76-4473-a18b-9b100ad5ea80", "type": "agent.status", "createdAt": "2026-11-
13T16:59:31.077Z",
  "data": { "accountRef": "softphone", "agentRef": "asha", "status": "AVAILABLE", "online":
true,
```

```
            "departments": ["c09c1657-d18a-4a32-94c7-4ffa24e038c3"] } }
```

`media.ready` · `media.failed`

```
{ "id": "facd42e5-8867-4b10-ae19-cb7ecfe02578", "type": "media.ready", "createdAt": "2026-11-13T09:46:09.038Z",
  "data": { "mediaId": "eebb855e-2a03-4af7-a8c8-301f85877308", "accountRef": "softphone", "durationMs": 24192 } }
```

`media.failed`: `{"mediaId", "accountRef", "error"}`.

`test.ping`

Not in the usual envelope:

```
{ "type": "test.ping", "deliveryId": "d451ae97-ea90-4039-9098-0363acbdeda7", "clientAppId": "bd74035b-bdb6-450d-885b-222e83719906",
  "at": "2026-11-13T15:19:12.682569166Z" }
```

Signed like every webhook; answer 2xx.

3.2 The can-call check

Optional. When the client app has a **can-call URL**, the engine asks it before calls, so the client app can apply its own rules (the business has credit, its plan allows calls now, a number is blocked):

```
GET https://app.example.co.tz/hooks/can-call?accountRef=pmk-chemicals&type=INCOMING&from=%2B255745051250&to=%2B255677000010
X-Ringmaster-Timestamp: 1794588244
X-Ringmaster-Signature: <hex HMAC-SHA256(secret, "<timestamp>.")>      (empty body)
```

Query	Description
`accountRef`	The account
`type`	`INCOMING` · `BROADCAST` · `CAMPAIGN` · `AGENT_CALL`
`from` / `to`	E.164 (`from` is empty for call-outs)

Answer (bare, or inside the usual envelope's `data`):

```
{ "allowed": true, "maxMinutes": 10, "reason": null }
```

Field	Description
`allowed`	`false` refuses the call (missing = `true`)
`maxMinutes`	Optional: end this call after this many minutes (e.g. the business's remaining credit)
`reason`	Your short code, kept with the refusal

Time limit: connect 0.7 s, answer 1.5 s. When the URL does not answer in time or fails:

Call type	When the check fails	When `allowed: false`
`INCOMING`	**Allowed** (a caller is never refused because the client app is down)	Refused: caller hears the service is unavailable; `result: REJECTED`
`BROADCAST` / `CAMPAIGN`	**Refused**	The call-out is paused (`CAN_CALL:<reason>`, `callout.paused`)
`AGENT_CALL`	**Refused**	`403 CAN_CALL_REFUSED` to `POST /v1/calls/agent`

For call-outs the check runs once per batch of people, not per person.

3.3 Agent desk signals

Sent by the engine into the agent's **lobby room** (LiveKit data messages, JSON) while their browser is connected (Part 2D, I12).

`type`	When	Fields	What the desk does
`ring`	A caller is offered to this agent	`callId`, `from`, `department`	Show "Incoming call", play the ring, offer Accept / Decline
`ring_cancel`	The offer ended	`callId`, `reason`	Stop ringing, hide the card
`connecting`	The agent's **mobile** answered the ring (ring mode with mobiles)	`callId`, `via: "mobile"`	Show "On a call on your mobile"

`ring_cancel.reason`	Means
`answered_elsewhere`	Another agent accepted it
`caller_left`	The caller hung up or left the waiting room
`timeout`	Nobody pressed Accept within the department's ring time

`ring_cancel.reason`	**Means**
`declined`	This agent declined
`not_joined`	The agent accepted but their browser never joined the call room (15 s); the caller went back to the queue

When the agent changes their own status while ringing (Away, Offline), the ring is cancelled without a signal: the desk already knows.

```
{ "type": "ring", "callId": "8a5e282d-93a1-443b-81f3-4eb7294a0639", "from": "+255745051250", "department": "Mapokezi" }
{ "type": "ring_cancel", "callId": "8a5e282d-93a1-443b-81f3-4eb7294a0639", "reason": "timeout" }
```

A ring is sent once the caller has heard everything before the waiting music, so the agent never meets a caller who does not yet know they are being connected.

Revision #1
Created 13 November 2026 18:52:05 by Admin Qbit
Updated 13 November 2026 18:52:33 by Admin Qbit