Real-time API y Conector MCP

Las dos vías que llevan una integración más allá de «identificar y registrar»: la **Real-time API** te deja enterarte y **controlar la llamada mientras ocurre** (Call Popup, transferir, aparcar, conferencia, grabar); el **Conector MCP** deja que un **agente de voz IA** use tu CRM como herramienta durante la conversación.

Dirigido a desarrolladores e integradores

Vías cubiertas Real-time API · Conector MCP

Requiere credenciales de tu instalación

---

01 Panorama

Conexia Infinity ofrece cinco vías de integración; esta referencia cubre las dos que operan **en tiempo real**. Las otras tres se documentan aparte: **API Key** (consultar histórico, CDR, transcripción), **ARI** (originar llamadas / click-to-call) y **Webhook** (avisos HTTP sin mantener conexión).

VÍA	QUÉ RESUELVE	SENTIDO
Real-time API	Eventos en directo + control de la llamada (Call Popup, transferir, aparcar, conferencia, grabar)	Puente ↔ plataforma
Conector MCP	Que un agente de voz IA consulte y actúe sobre tu CRM durante la llamada	Plataforma → tu servidor MCP

CONVENCIÓN DE ESTA GUÍA

Los valores entre `<ángulos>` son **marcadores** que sustituyes por los de tu instalación (`<RT_HOST>`, `<EXT>`, `<TENANT>`, `<MCP_URL>`…). Las credenciales concretas y los endpoints de tu tenant te los facilita tu contacto de Conexia; no se publican aquí.

PARTE A

Real-time API — enterarse y controlar

02 Qué es y cómo conecta

La Real-time API es un canal de **eventos y control en directo**. A diferencia de un webhook (aviso puntual por HTTP), mantiene una **conexión persistente** por la que la plataforma te envía eventos de llamada en el instante en que ocurren, y por la que tú envías comandos de control. Es el motor del **Call Popup** y de los botones de acción del CRM.

Arquitectura recomendada

El patrón que respeta la seguridad: un **proceso puente en tu backend** mantiene la conexión con la plataforma, se autentica por cada agente y reenvía los eventos a la interfaz del agente (por WebSocket propio, SSE, etc.). Así el navegador nunca ve credenciales.

Transporte: conexión TCP / WebSocket sobre TLS.

Autenticación: por agente (credenciales de su extensión). Cada sesión corresponde a un agente.

Alcance: una sesión recibe **solo los eventos de las llamadas de ese agente** (privacidad por diseño). Para ver a todo el equipo, se usa la API Key, no una sesión Real-time compartida.

> **SESIÓN POR AGENTE**
>
> Tu puente debe mantener una sesión por agente conectado, no una global. Un panel de supervisión que vea al equipo completo se resuelve con la *API Key* (snapshot de todas las extensiones), no aquí.

03 Call Popup: identificar la llamada entrante

El caso estrella: **suena el teléfono y aparece al instante la ficha del contacto**. Lo resuelve la Real-time API, porque «se entera» de la llamada en el momento en que entra.

1. Tu puente está conectado con la sesión del agente y escuchando eventos.
2. Entra una llamada. La plataforma emite `event_call_started` con `state:1` (Ring In).
3. El campo `connectedNum` trae **el número de quien llama**. Con él, tu backend busca en el CRM.
4. Tu interfaz abre el **pop-up**: nombre, cuenta, últimas interacciones.
5. Al contestar llega `event_call_connected` (`state:3`); al colgar, `event_call_finished`. Con ambos marcas la actividad como atendida o perdida.

EVENTO QUE DISPARA EL POP-UP (RING IN)

```
{ "event": "event_call_started", "type": "event",
  "uid": "<CALL_UID>",           // id único de ESTA llamada; úsalo como clave
  "state": 1,                     // 1 = Ring In (entrante, sonando)
  "ctype": 1,                     // 1 = regular → connectedNum/Name válidos
  "number": "<EXT>",             // tu extensión / número local
  "connectedNum": "<NUM_LLAMANTE>", // ← quién llama: buscar en el CRM
  "connectedName": "",            // nombre según la centralita (si lo hay)
  "pbx_user": 1, "cdr_uid": "<CDR_UID>" }
```

CÓMO LEER LOS CAMPOS

`uid` es la clave de la llamada: todos los eventos siguientes (contestada, actualizada, colgada) lo repiten. `connectedNum` es el número del interlocutor; normalízalo (quita prefijos y espacios) antes de buscar.

Distinguir entrante de saliente

`1` Ring In — entrante → abre pop-up

`2` Ring Out — saliente → el CRM ya sabe a quién llama

`3` Connected — contestada

Timbre agrupado

Si suenan a la vez el teléfono de mesa, el softphone y el móvil del agente, la plataforma envía **un solo** `event_call_started`: es una única llamada. Tu pop-up aparece una vez, no tres. No tienes que deduplicar.

EL POP-UP NO DESCUELGA

La Real-time API no contesta la llamada por ti (eso es SIP, en el softphone/teléfono). El pop-up es señalización + contexto: muestra quién llama y ofrece acciones. Para datos históricos del contacto, complementa con la API Key tras el evento.

---

04 Control de la llamada en curso

Una vez identificada la llamada por su `call_id`, la Real-time API te da los botones de acción del CRM. Todos operan sobre el id de la llamada.

BOTÓN EN EL CRM	ACCIÓN	NOTAS
Colgar	`hangup`	Con el `call_id`.
Transferencia ciega	`transfer`	Al número indicado, sin consultar.
Transferencia atendida	`atxfer` → `hangup` / `atxfer_cancel`	Consulta antes de completar.
Aparcar	`park`	Emite un evento con la plaza asignada.
Añadir a alguien (conferencia)	`add_into_call`	Crea una conferencia dinámica.
Conferencia (crear / invitar)	`start_conf`	Estática o dinámica.
Silenciar / expulsar	`mute` / `kick`	Según permisos.
Grabar / pausar / parar	`cmd_record_start` / `_pause` / `_stop`	Requiere grabación instantánea habilitada.
Cambiar de dispositivo	`switch_device`	Pasar la llamada al móvil, softphone, etc.
Notas / asunto de la llamada	`cmd_call_metadata_update_notes` / `_subject`	No emiten evento.

EJEMPLO: TRANSFERIR LA LLAMADA ACTIVA

```
{ "action": "transfer", "call_id": "<CALL_ID>", "number": "<DESTINO>" }
```

SEGUIR EL PROGRESO

Iniciar una llamada o invitar a una conferencia genera un trabajo en segundo plano con eventos de estado (Calling → Ringing → Connected → Completed / Failed). Úsalos para pintar el estado en la interfaz y mostrar errores claros.

05 Presencia en vivo

Para reflejar en la interfaz del agente el estado de un teléfono sin sondear, el evento `extension_hint` avisa cuando cambia (Idle, In Use, Ringing, On Hold…). Ideal para un BLF reactivo.

TODO EL EQUIPO VS. UN AGENTE

Los *hints* de la Real-time API reflejan lo que ve la sesión del agente conectado. Para un panel que vea a **todo** el equipo, usa la API Key (snapshot de todas las extensiones).

PARTE B

Conector MCP — que la IA use tu CRM

06 Qué es el conector MCP

MCP (**Model Context Protocol**) es un estándar por el que un agente de IA accede a **herramientas** externas: funciones que invoca para leer o escribir datos. El conector MCP de Conexia Infinity permite que el **agente de voz IA** de la centralita llame a herramientas que **tú alojas** en tu propio servidor MCP.

Las herramientas **no se definen en la centralita**: viven en tu servidor. La plataforma solo apunta a su URL. Tú decides qué operaciones expones (buscar contacto, crear cita, consultar pedido…), con qué parámetros y con qué permisos.

POR QUÉ IMPORTA

Es la diferencia entre una IA que recita un guion y una que **trabaja con datos reales de tu negocio**: durante la llamada, el agente identifica a quien llama, consulta su ficha y actúa — directamente contra tu CRM.

07 Roles: quién invoca a quién

El punto que más confusión genera. En las demás vías, *tu sistema* habla con la plataforma. Con MCP el sentido se invierte: la plataforma invoca tu servidor.

```
Agente de voz IA (centralita) — el cliente MCP
  | invoca herramienta
Servidor MCP (lo alojas tú) — el proveedor
  | consulta / escribe
Tu CRM — contactos, tareas, estados
```

LA CENTRALITA ES EL CLIENTE MCP

Tú no «llamas» al MCP: publicas herramientas y el agente las invoca cuando su conversación lo requiere. En la plataforma solo registras el `Name` y la `URL` (`<MCP_URL>`) de tu endpoint.

08 Diseñar las herramientas del CRM

Lo que expongas en tu servidor MCP es lo que el agente podrá hacer. Casos habituales:

HERRAMIENTA (EJEMPLO)	QUÉ HACE	EN LA CONVERSACIÓN
`buscar_contacto`	Busca por teléfono / documento / nombre	«¿Con quién hablo?»
`estado_pedido`	Consulta el estado de un pedido o servicio	«¿Cómo va mi instalación?»
`agendar_cita`	Crea o mueve una cita	«Quiero cita para el martes»
`consultar_disponibilidad`	Devuelve huecos disponibles	«¿Qué horas tenéis libres?»
`crear_ticket`	Abre una incidencia	«Quiero reportar una avería»

BUENAS HERRAMIENTAS PARA VOZ

Acotadas (una operación clara cada una), con parámetros mínimos y explícitos y descripciones claras — el agente decide cuál usar leyendo esas descripciones. Devuelve datos concisos, legibles en voz alta. Evita operaciones ambiguas o de efecto masivo.

09 Flujo de una llamada con IA

1. **Entra la llamada** a un número enrutado al agente de voz. La centralita la atiende con la IA.
2. **El agente saluda y conversa** según su prompt.
3. **Necesita un dato** e **invoca una herramienta** de tu servidor MCP.
4. **Tu servidor MCP responde** con los datos del CRM; el agente los comunica en voz.
5. **Actúa**: agenda, abre ticket, informa… y si hace falta, **transfiere a un humano** o **cuelga** al resolver.

MCP NO REEMPLAZA A LAS OTRAS VÍAS

El MCP es la **capa de decisión de la IA**, no el transporte de la señalización. La detección y el encaminamiento siguen siendo de la centralita; el registro de actividad puede ir por Webhook o Real-time API; los históricos, por API Key. El MCP aporta el «cerebro» que consulta tu CRM durante la conversación.

10 Seguridad del servidor MCP

EL AGENTE ACTUARÁ SOBRE LO QUE EXPONGAS

Tu servidor MCP es una superficie de acción sobre tu CRM. Todo lo que publiques como herramienta, el agente podrá invocarlo durante una llamada. Trátalo con el mismo cuidado que una API pública.

Mínimo privilegio. Expón solo las herramientas imprescindibles; evita operaciones destructivas o de gran alcance.

Autenticación en tu lado. El servidor MCP debe autenticar y limitar quién invoca sus herramientas; no dependas de que «solo lo llama la centralita».

Solo HTTPS para el endpoint MCP.

Valida entradas. Los parámetros llegan de una conversación en lenguaje natural: normaliza y valida antes de tocar datos.

Registra y limita. Log de cada invocación y límites de frecuencia.

Cuida los datos sensibles. Devuelve solo lo necesario para la conversación; no expongas PII de más.

11 Cómo empezar

1. **Solicita credenciales.** Pide a tu contacto de Conexia el acceso Real-time (por extensión) y, para MCP, ten listo tu endpoint `<MCP_URL>`.
2. **Levanta el puente Real-time** en tu backend, con una sesión por agente, y prueba el Call Popup con `event_call_started`.
3. **Añade control** (transferir, grabar) sobre el `call_id` de la llamada activa.
4. **Para IA,** publica primero herramientas MCP de **solo lectura** (identificar contacto, consultar estado); cuando el comportamiento sea sólido, añade escritura con validación.

DISPONIBILIDAD

El agente de voz IA y el conector MCP forman parte del módulo de IA de la plataforma; su disponibilidad y opciones dependen de la versión y la licencia de tu instalación. Confírmalas con tu contacto de Conexia.

¿Buscas las otras vías? Consulta las referencias de **API Key**, **ARI** y **Webhook** en el índice de integraciones.

Documento para desarrolladores. Los valores entre `<ángulos>` son marcadores; los endpoints y credenciales de tu tenant los facilita tu contacto de Conexia. La disponibilidad de funciones depende de la versión y la licencia de la instalación.