









# MCP: Conecta tu IA con el Mundo Real



# El Problema de la IA Actual - Desconectada del Mundo Real

-  Los modelos de IA están "desconectados" del mundo real
-  Tienen conocimiento, pero no pueden acceder a datos actuales
-  No pueden acceder a datos en tiempo real
-  No pueden ejecutar funciones externas
-  Limitados a su entrenamiento inicial
-  Como estar en Matrix sin poder "enchufarse" al mundo exterior

# MCP: Cuando tu IA dice "I Know Kung Fu"

- 🥋 Crear un servicio MCP
- 🔌 MCP
- ⚡ Acceso seguro a datos en tiempo real
- 🛠 Ejecución de herramientas y funciones
- 🌐 Integración con APIs y servicios externos
- 💡 "I know Weather API", "I know Currency Exchange", "I know Database Queries"





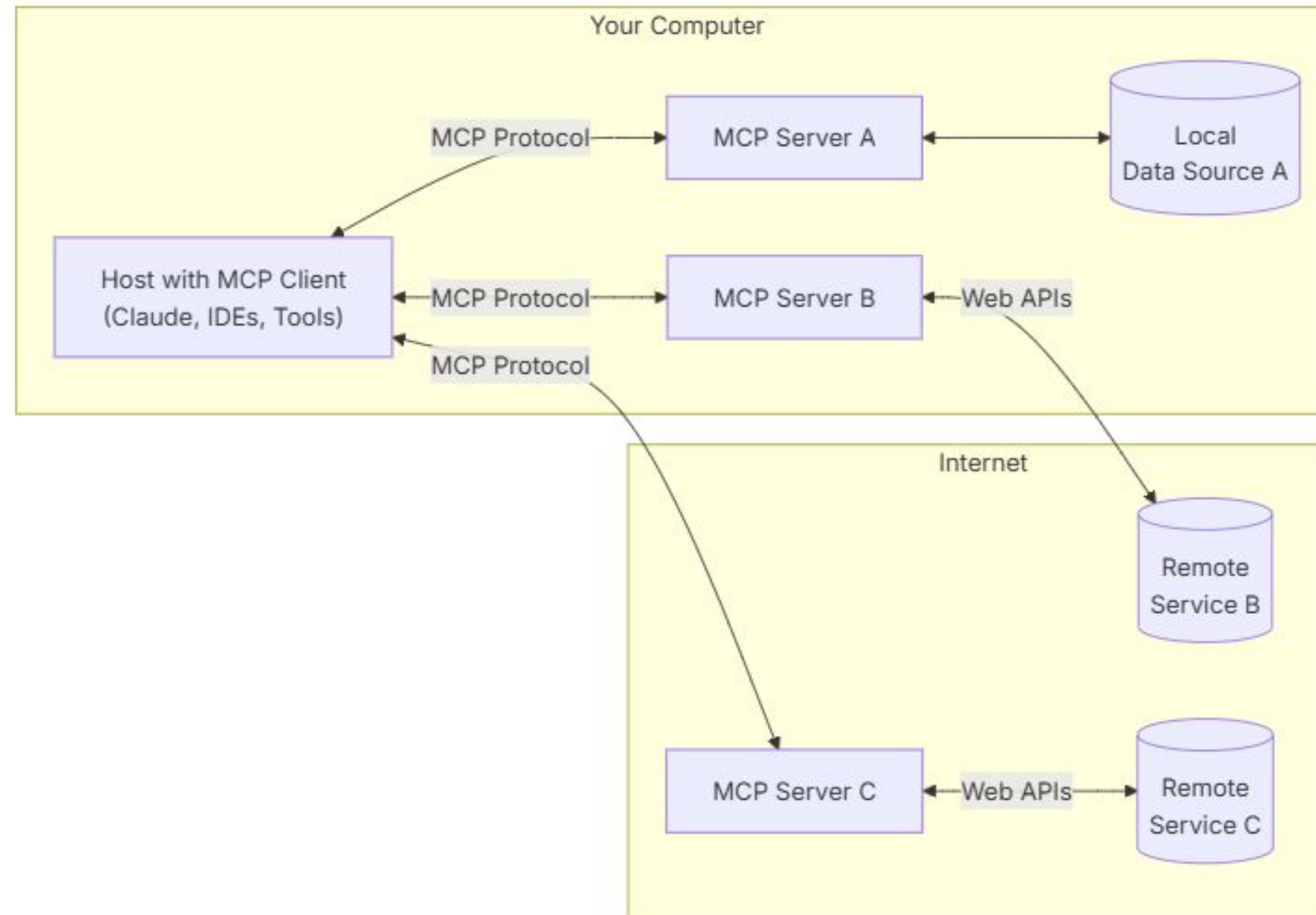
# Model Context Protocol - Definición

El **Model Context Protocol (MCP)** es un protocolo estándar abierto que permite a los modelos de IA acceder de forma segura a datos y herramientas externas.

Desarrollado por Anthropic, MCP actúa como un puente entre los modelos de lenguaje y las aplicaciones del mundo real.

-  **Protocolo estándar abierto** desarrollado por Anthropic
-  **Puente seguro** entre modelos de IA y aplicaciones externas
-  **Arquitectura cliente-servidor simple** y eficiente
-  **Seguridad integrada** para acceso controlado
-  Documentación oficial:  
<https://modelcontextprotocol.io/>
-  Anthropic:  
<https://www.anthropic.com/>
-  MCP Marketplace:  
<https://github.com/modelcontextprotocol/servers>

# Cómo Funciona MCP - Arquitectura



# MCP en Acción - Casos de Uso

- 🌤️ **APIs de Clima:** Información meteorológica en tiempo real
- 💵 **Conversión de Monedas:** Tipos de cambio actualizados
- 🌐 **Datos Geográficos:** Información de países y ciudades
- 📊 **Bases de Datos:** Consultas dinámicas a sistemas empresariales
- 🔍 **Web Scraping:** Extracción de datos de sitios web
- ✉️ **Automatización:** Envío de emails, notificaciones

Tiempo de castellon de la plana para el fin de

**E** Ejemplos

Voy a obtener la previsión meteorológica para Castellón de la Plana.

🔍 weather/get-weather ▾

He obtenido información meteorológica para Castellón de la Plana. Los datos muestran:

**Ubicación:**

- **Latitud:** 40°
- **Longitud:** -0.0625°
- **Elevación:** 42 metros sobre el nivel del mar







**Información técnica:**

- **Zona horaria:** GMT
- **Tiempo de generación:** 0.0008 ms

Sin embargo, parece que la respuesta no incluye la previsión meteorológica específica para el fin de semana. Los datos obtenidos son principalmente las coordenadas geográficas y información de ubicación de Castellón de la Plana.







Para obtener una previsión meteorológica más detallada del fin de semana, sería necesario acceder a un servicio meteorológico más completo que proporcione información sobre temperatura, precipitaciones, y condiciones climáticas específicas para los próximos días.

# ¿Por Qué Elegir MCP?

-  **Estándar Abierto:** No vendor lock-in
-  **Seguridad:** Acceso controlado y auditado
-  **Simplicidad:** Fácil implementación
-  **Escalabilidad:** Crece con tus necesidades
-  **Interoperabilidad:** Compatible con múltiples modelos
-  **Comunidad:** Respaldado por Anthropic y desarrolladores



# Demo Preview

-  Servidor MCP Completo con:
-  API de clima por ciudad
-  Conversión de monedas en tiempo real
-  Información detallada de países
-  Autenticación y seguridad (por ahora no)
-  Logging y monitoreo

```
67 // Función para obtener coordenadas usando Nominatim (OpenStreetMap)
68 async function getCityCoordinates(cityName: string): Promise<{ lat: number; lon: number; displayName: string }> {
69   const encodedCity = encodeURIComponent(cityName);
70   const url = `${NOMINATIM_API_BASE}/search?q=${encodedCity}&format=json&limit=1&addressdetails=1`;
71
72   const data = await makeRequest<NominatimResult[]>(url);
73
74   if (!data || data.length === 0) {
75     throw new Error(`Ciudad '${cityName}' no encontrada`);
76   }
77
78   const result = data[0];
79   return {
80     lat: parseFloat(result.lat),
81     lon: parseFloat(result.lon),
82     displayName: result.display_name
83   };
84 }
85
86 // Función para obtener datos meteorológicos usando Open-Meteo
87 async function getWeatherData(lat: number, lon: number): Promise<any> {
88   const url = `${OPEN_METEO_API_BASE}/forecast`;
89   const params = {
90     "latitude": lat,
91     "longitude": lon,
92     "daily": ["temperature_2m_max", "temperature_2m_min", "rain_sum", "precipitation_sum"],
93     "hourly": ["temperature_2m", "wind_speed_10m", "relative_humidity_2m", "precipitation", "precipitation_probability", "rain"],
94     "current": ["relative_humidity_2m", "temperature_2m", "precipitation", "rain", "wind_speed_10m"],
95     "timezone": "auto"
96   };
97 }
```



# "Welcome to the Real World" - Tu IA Conectada

- 🍁 Tu IA ahora puede "enchufarse" al mundo real
- 🧠 Conocimiento + Datos en tiempo real = IA Poderosa
- 🚀 Posibilidades infinitas de aplicación
- 💡 El futuro de la IA es conectada, no aislada







# ¡Muchas gracias por asistir!

---



# DATOS DE CONTACTO

[www.apicamp.org](http://www.apicamp.org)



Linkedin

[APICAMP](#)



Twitter

[APICAMP](#)



Instagram

[APICAMP](#)