Arquitecturas Cognitivas Computacionales y Metodologías Avanzadas de Ingeniería de Prompts: Un Análisis Técnico Profundo (2025-2026)

1. Introducción: La Bifurcación del Paradigma de Inferencia y la Era del Razonamiento

El panorama de la Inteligencia Artificial Generativa ha experimentado una transformación estructural fundamental durante el ciclo 2024-2025. La ingeniería de prompts, anteriormente considerada un arte empírico de "susurrar a la máquina" mediante heurísticas lingüísticas, ha madurado hacia una disciplina de ingeniería de sistemas rigurosa. Este cambio ha sido precipitado por la emergencia de dos desarrollos arquitectónicos críticos: la integración de capacidades de "Razonamiento Profundo" (System 2 Thinking) directamente en el proceso de inferencia y la expansión masiva de las ventanas de contexto hacia escalas de millones de tokens.

El análisis del estado del arte revela que la eficacia de las estrategias de prompting ya no es universalmente transferible. La arquitectura subyacente de cada modelo de frontera —sea la estructura densa optimizada para multimodalidad de Google Gemini 3.0, el enfoque de Mixture-of-Experts (MoE) de Moonshot Kimi k2, o el razonamiento híbrido de OpenAI GPT-5.2— impone restricciones y affordances únicas que dictan la estructura óptima de la entrada. La documentación técnica reciente y los benchmarks de la industria, como SWE-bench Verified y GDPval, sugieren que el rendimiento del modelo depende menos de la elocuencia del prompt y más de la estructuración topológica de la información y la definición precisa de restricciones.[1]

Este informe técnico disecciona las idiosincrasias operativas de las cinco arquitecturas dominantes, proporcionando una exégesis de los mecanismos de atención que validan estas estrategias. Además, se explora la transición hacia la optimización automatizada, donde marcos como DSPy y TextGrad están reemplazando la optimización manual mediante la aplicación de principios de descenso de gradiente textual y programación declarativa.

---

2. Idiosincrasias Arquitectónicas y Optimización Específica del Modelo

La interacción con los modelos de lenguaje grandes (LLMs) modernos requiere una comprensión profunda de cómo cada arquitectura procesa, prioriza y retiene la información. A continuación, se presenta un análisis granular de las estrategias de optimización para los modelos objetivo.

2.1. Anthropic Claude (Familia 4.5: Sonnet, Haiku, Opus)

La familia Claude 4.5, lanzada en el cuarto trimestre de 2025, se ha establecido como el estándar de oro para tareas de codificación y agentes autónomos de larga duración, operando bajo el marco de seguridad ASL-3.[1] La arquitectura de Claude presenta una sensibilidad única a la estructura jerárquica y a la delimitación semántica, lo que diferencia radicalmente su ingeniería de prompts de la de sus competidores.

Estructura de Entrada Preferida: El Paradigma de Delimitación XML

A diferencia de los modelos de la serie GPT, que históricamente han favorecido estructuras Markdown o JSON implícito, la familia Claude demuestra una mejora significativa en la adherencia a las instrucciones y en la calidad del razonamiento cuando se utiliza un etiquetado XML estricto. Esta preferencia no es meramente estética; se alinea con la forma en que el modelo ha sido entrenado para segmentar y atender a diferentes partes del contexto.

El uso de etiquetas semánticas actúa como un mecanismo de "enmascaramiento de atención suave" (soft attention masking). Al encapsular componentes críticos del prompt dentro de etiquetas como <context>, <instructions>, <thinking_process> y <output_format>, se reduce la "fuga" de atención entre las instrucciones del sistema y los datos ruidosos del usuario. La documentación técnica de Anthropic y los análisis de la comunidad de desarrolladores confirman que esta estructura permite al modelo analizar el prompt con mayor precisión, reduciendo errores de interpretación en tareas complejas.[5]

La Técnica del "Pre-fill" (Pre-llenado):
Una de las optimizaciones más potentes y específicas para Claude es la estrategia de pre-llenar la respuesta del asistente. Debido a la naturaleza autoregresiva de los LLMs, el primer token generado condiciona fuertemente la trayectoria de toda la respuesta. En la API de Claude, es posible inyectar texto en el rol assistant al final de la secuencia de mensajes. Por ejemplo, al forzar el inicio de la respuesta con <analysis>, se obliga al modelo a entrar inmediatamente en el modo de procesamiento analítico definido en el prompt del sistema, evitando preámbulos conversacionales innecesarios ("Claro, aquí está el análisis...") y alineando la salida con el esquema deseado desde el token cero.6

Manejo del Contexto y Persistencia en Ventanas de 200k+

Aunque Claude 4.5 soporta ventanas de contexto de hasta 1M de tokens (en preview) y 200k estándar, la gestión de la "memoria" en agentes de larga duración presenta desafíos de degradación de rendimiento y alucinación.

- **Edición de Contexto y "Ghost Context":** Para flujos de trabajo que se extienden

durante decenas de horas (como la refactorización de código en Claude Code), la acumulación de historial irrelevante puede degradar el razonamiento. Claude 4.5 introduce capacidades de edición de contexto y herramientas de memoria que permiten la eliminación activa de "contexto fantasma" (ghost context) —información de pasos de razonamiento fallidos o datos obsoletos. La estrategia óptima implica instruir al modelo para que utilice herramientas de gestión de memoria para "comprimir" su estado interno en artefactos persistentes antes de limpiar su ventana de contexto.[7]

- **Topología de la Información:** La evidencia sugiere que la recuperación de información ("needle in a haystack") en Claude es más robusta cuando los documentos de referencia se colocan *antes* de las instrucciones detalladas de la tarea dentro del prompt. Esto contrasta con otros modelos que sufren de un sesgo de recencia más pronunciado. Se recomienda estructurar el prompt de sistema inyectando primero la base de conocimiento en etiquetas <knowledge_base> y cerrando con las instrucciones operativas <task_definition>.[9]

Tokens Especiales y Delimitadores

El modelo responde a tokens y estructuras específicas que activan subrutinas de comportamiento:

- **Thinking Explícito:** Para tareas que requieren razonamiento complejo pero donde no se activa automáticamente el modo "Extended Thinking", se debe instruir al modelo para usar etiquetas como <antThinking> o simplemente <thinking>. Esto externaliza el monólogo interno, permitiendo al usuario (o al sistema evaluador) auditar la lógica antes de la generación de la respuesta final.[7]
- **Artefactos:** Claude está optimizado para generar contenido autónomo y reutilizable (código, documentos) dentro de "Artefactos". Los prompts deben solicitar explícitamente la creación de estos artefactos para tareas de generación de contenido sustancial, lo que activa una interfaz de usuario especializada en los entornos de chat de Anthropic.[7]

2.2. Google Gemini (3.0 Pro/Flash)

Google Gemini 3.0 representa un avance significativo hacia el razonamiento multimodal nativo y una eficiencia sin precedentes en el manejo de contextos masivos. Su arquitectura, diseñada desde cero para la multimodalidad y el contexto largo, exige un enfoque de prompting distintivo que prioriza la estructura sistémica sobre la narrativa conversacional.

Estructura de Entrada: Instrucciones de Sistema y Concisión

La documentación técnica de Gemini 3 enfatiza un principio rector: "Precisión sobre Persuasión". A diferencia de modelos anteriores que requerían "encantamiento" o role-play elaborado en el prompt del usuario, Gemini 3 responde mejor a instrucciones directas y funcionales ubicadas en el bloque de System Instructions.

- **Separación Estricta de Roles:** Google recomienda encarecidamente utilizar el campo

dedicado de system_instruction en la API para definir el rol, el tono, las restricciones de seguridad y el formato de salida. Esto ancla el comportamiento del modelo de manera mucho más efectiva que incluir estas directivas dentro del turno de usuario, donde pueden diluirse por el contenido de la consulta.[10]

- **Control de Verbosidad:** Por defecto, Gemini 3 exhibe una tendencia hacia la concisión extrema ("lacónico"). Si el caso de uso requiere explicaciones detalladas o un tono conversacional ("chatty"), esto debe ser una instrucción explícita en el sistema (e.g., "Actúa como un profesor universitario que explica conceptos con analogías detalladas"). De lo contrario, el modelo optimizará agresivamente para la eficiencia de tokens, lo cual es ideal para tareas de extracción de datos pero subóptimo para la tutoría.[12]

Manejo del Contexto Masivo (1M+ Tokens)

Gemini 3 destaca por su capacidad para ingerir repositorios de código completos, libros enteros o horas de video en una sola ventana de contexto. Sin embargo, la atención sobre este volumen de datos requiere una estructuración cuidadosa.

- **Estrategia de Anclaje Final (End-Anchoring):** A pesar de las mejoras en los mecanismos de atención, la física de los modelos de transformadores aún favorece la información ubicada al final de la secuencia (sesgo de recencia). Cuando se trabaja con contextos masivos (RAG o carga directa de documentos), la instrucción específica o la consulta del usuario debe colocarse estrictamente **al final** del prompt, *después* de todo el bloque de datos.
- **Puente Contextual:** Para maximizar la recuperación, es crucial utilizar frases de transición explícitas que conecten el bloque de datos masivo con la consulta final. Frases como "Basado exclusivamente en la información proporcionada anteriormente, responde a la siguiente pregunta..." actúan como un delimitador suave que reorienta la atención del modelo desde la ingesta pasiva hacia la generación activa.[10]

Multimodalidad Coherente

El prompting para Gemini 3 debe trascender el texto. Al ser un modelo multimodal nativo, las instrucciones pueden y deben referenciar diferentes modalidades como ciudadanos de primera clase. Un prompt optimizado entrelaza referencias a imágenes, video y texto (e.g., "Analiza la tendencia de ventas en el gráfico de la imagen 1, compárala con las proyecciones en el archivo de texto adjunto, y busca discrepancias en el minuto 5:30 del video proporcionado"). Esta capacidad de razonamiento cruzado se activa mejor cuando las modalidades están claramente etiquetadas y referenciadas en la instrucción textual.[13]

2.3. OpenAI GPT (5.2: Instant, Thinking, Pro)

Lanzado en diciembre de 2025, la serie GPT-5.2 introduce una distinción explícita en la experiencia de usuario y la API entre los modos de pensamiento "Instantáneo" y "Pensante"

(Thinking), redefiniendo las expectativas sobre latencia y profundidad de razonamiento.[2]

Estructura de Entrada: Markdown y Descomposición Estructural

GPT-5.2 mantiene la preferencia histórica de los modelos de OpenAI por el formato Markdown y JSON, pero muestra una sensibilidad mucho mayor a la estructura lógica de la tarea.

- **Prompting por Fases para "Extended Thinking":** Aunque el modo "Thinking" realiza Chain-of-Thought (CoT) automáticamente, los usuarios avanzados pueden activar un "Pensamiento Extendido" o profundo estructurando el prompt en fases explícitas. En lugar de solicitar una solución directa, el prompt debe exigir una derivación por etapas (e.g., "Fase 1: Enumera todas las suposiciones fundamentales. Fase 2: Deriva la fórmula teórica. Fase 3: Aplica los valores numéricos."). Esta estructura fuerza al modelo a dedicar más tiempo de computación (test-time compute) a la planificación antes de la ejecución, resultando en un rendimiento superior en tareas STEM y de codificación compleja.[15]
- **Especificidad Semántica y Castigo a la Ambigüedad:** A diferencia de GPT-4, que intentaba inferir la intención del usuario ante prompts vagos, GPT-5.2 tiende a tomar la ruta más eficiente o "perezosa" si la tarea no está estrictamente acotada. La ingeniería de prompts para este modelo requiere una especificación rigurosa de las restricciones y los formatos de salida para evitar respuestas genéricas. La ambigüedad se traduce directamente en una menor profundidad de razonamiento.[17]

Manejo del Contexto y Fiabilidad

- **Reducción de Alucinaciones mediante Verificación:** GPT-5.2 Thinking muestra una reducción del 30% en errores factuales en comparación con GPT-5.1. Para capitalizar esto, los prompts deben incentivar explícitamente la verificación cruzada interna ("Verifica cada afirmación contra el documento proporcionado antes de incluirla en la respuesta final"). Aunque el modelo lo hace por defecto, la instrucción explícita refuerza este comportamiento en tareas de alto riesgo.[2]
- **Contexto Largo (400k) y Recuperación:** GPT-5.2 ha mejorado significativamente su capacidad de recuperación en contextos largos, resolviendo benchmarks de "aguja en un pajar" de múltiples agujas (multi-needle) con alta precisión. La clave es proporcionar una estructura clara en el documento de entrada (títulos, secciones) que el modelo pueda indexar semánticamente durante la inferencia.[14]

Tokens y Uso de Herramientas

- **Agentic Tooling:** GPT-5.2 está altamente optimizado para decidir cuándo usar herramientas. Sin embargo, en flujos de trabajo críticos, se recomienda definir claramente en el prompt del sistema cuándo *no* usar una herramienta y confiar en el conocimiento paramétrico, o viceversa. El modelo es proactivo en la verificación de datos externos, por lo que el prompt debe gestionar este equilibrio para optimizar latencia y

coste.[18]

2.4. Moonshot AI Kimi (Modelos k2 Thinking)

Kimi k2, un modelo MoE (Mixture-of-Experts) de 1 billón de parámetros (con 32B activos), ha irrumpido en el escenario open-source demostrando capacidades de razonamiento profundo y una estabilidad excepcional en secuencias de herramientas largas.[3]

Estructura de Entrada: Activación de Patrones de Pensamiento

Investigaciones empíricas sugieren que Kimi k2 no opera en un modo monolítico, sino que posee "patrones de pensamiento" latentes que pueden ser disparados mediante la estructura semántica del prompt [19]:

1. **Modo Investigación (Research Mode):** Se activa con prompts que imponen una estructura de Alcance -> Hipótesis -> Evidencia -> Síntesis. Este patrón reduce los errores factuales hasta en un 30% al forzar al modelo a validar premisas antes de generar conclusiones.
2. **Modo Escritura (Writing Mode):** Se activa proporcionando un esquema detallado y una guía de estilo. En este modo, el modelo prioriza el flujo narrativo y la coherencia estilística sobre la verificación factual profunda, ideal para la generación de borradores rápidos.
3. **Modo Análisis (Analysis Mode):** Se activa solicitando razonamiento paso a paso explícito ("Show your work"), identificación de suposiciones y comparación de escenarios contrafactuales.

Manejo del Contexto y Herramientas (Estabilidad en 200-300 Pasos)

La característica más destacada de Kimi k2 es su capacidad para mantener la coherencia en bucles de agente extremadamente largos. Mientras que otros modelos tienden a degradar su rendimiento o perder el objetivo original después de 30-50 llamadas a herramientas, Kimi mantiene la estabilidad hasta en 300 interacciones secuenciales.

- **Orquestación de Herramientas:** Para aprovechar esta capacidad, el prompt debe diseñarse para soportar y guiar estos bucles largos, definiendo objetivos globales claros que persistan a través de cientos de pasos intermedios.
- **Prompt de Sistema Minimalista:** Moonshot aconseja mantener el prompt de sistema simple y directo: "You are Kimi, an AI assistant created by Moonshot AI". Instrucciones de sistema excesivamente complejas pueden interferir con el ajuste de instrucciones (instruction tuning) interno del modelo, degradando su capacidad de razonamiento nativo.[20]

2.5. Zhipu AI GLM (GLM-4.6)

GLM-4.6 es un modelo denso que sobresale en la integración nativa de herramientas y

capacidades de razonamiento planificado, diseñado específicamente para funcionar como el cerebro de sistemas agénticos complejos.[21]

Estructura de Entrada: Uso de Herramientas Nativo y Planificación

GLM-4.6 no trata el uso de herramientas como un añadido, sino como una parte integral de su proceso de inferencia.

- **Definición de Herramientas Semánticas:** El modelo soporta definiciones de funciones al estilo OpenAI y el protocolo MCP. La clave para un prompting efectivo es proporcionar descripciones de herramientas ricas y semánticamente densas. GLM-4.6 utiliza su modo "thinking" para *planificar* el uso de herramientas antes de ejecutarlas, por lo que entender el *propósito* de la herramienta es tan importante como su sintaxis para el modelo.[22]
- **Modo Thinking Explícito:** A través de la API, se puede habilitar el parámetro thinking: {"type": "enabled"}. Esto activa mecanismos internos de Chain of Thought que son visibles en la salida (o procesables por el sistema), permitiendo al modelo descomponer problemas complejos en sub-tareas antes de emitir cualquier llamada a función.[23]

Multimodalidad Nativa en el Flujo de Herramientas

Una capacidad distintiva de la variante GLM-4.6V es la integración de inputs visuales directamente en las llamadas a herramientas. A diferencia de otros modelos que requieren OCR previo o descripción de texto, GLM-4.6V puede recibir una imagen (e.g., una captura de pantalla) como argumento de una función, analizarla y pasar el resultado estructurado al siguiente paso. Los prompts para este modelo deben diseñarse para cerrar el bucle "Ver -> Razonar -> Actuar -> Observar", aprovechando esta capacidad multimodal nativa.[24]

---

3. Tabla Comparativa de Sintaxis y Estrategias

La siguiente matriz sintetiza las estrategias de prompting más efectivas validadas para cada arquitectura, proporcionando una guía visual rápida para la adaptación de prompts.

Dimensión	Claude 4.5 (Anthropic)	Gemini 3.0 (Google)	GPT-5.2 (OpenAI)	Kimi k2 (Moonshot)	GLM-4.6 (Zhipu AI)
Estructura Preferida	**XML Estricto** (<tag>content</tag>) para	**System Instructions** dedicadas + Prompt	**Markdown** estructurado / JSON. Sensible a la jerarquía	**Patrones de Modo** (e.g., Scope -> Hypothesis	**Tool Definitions** ricas / Native Function

	segmentación semántica.	de usuario conciso y funcional.	de títulos.	-> Evidence).	Calls.
Contexto Largo	**Pre-llenado** de respuesta + XML para segmentar datos. Herramientas de edición de contexto.	Datos masivos primero, **consulta al final** (End-Anchoring). Uso de puentes contextuales.	Alta fidelidad (400k). Requiere estructura clara e indexable.	Ventana de 256k. Robusto en secuencias largas sin degradación.	200k tokens. Optimizado para análisis de documentos largos.
Razonamiento (CoT)	Etiquetas <thinking> explícitas o activación de "Extended Thinking".	Implícito. Se debe solicitar "paso a paso" explícitamente si se requiere output verbalizado.	**Thinking Mode** automático. Forzar **Fases** explícitas para profundidad máxima.	**Thinking Model** nativo. Activar con comandos como "Show your work".	Parámetro API thinking: enabled. Planificación previa a la acción.
Uso de Herramientas	SDK de Agentes. Checkpoints para estado. Control de verbosidad.	Multimodal nativo. Integración profunda con Google Workspace.	Alta fiabilidad. Detección automática. Gestión de "Tool Choice".	**Estabilidad Extrema** (300+ llamadas secuenciales).	**Multimodal Inputs**. Imágenes como argumentos de función directos.
Sistema de Prompt	Rol detallado en System Prompt. Uso extensivo de XML	Definir Rol y Restricciones al inicio (System Field). Evitar charla	Definición de identidad clara. Evitar "fluff" o instruccion	Simple: "You are Kimi...". Definir modo de operación (Investigad	Soporte nativo para System + Tool schemas complejos y

	para delimitar secciones.	innecesaria.	es vagas.	or/Escritor).	MCP.
Tokens Especiales	<antThinking>, artifacts, <prefill>	N/A (Manejado por API/System Instructions).	N/A (Manejado por API).	Tokens de pensamiento internos (visibles u ocultos).	<tool_call>, <think> (en outputs crudos/API).

---

4. Frameworks de Optimización Automatizada (LLMs optimizando LLMs)

La ingeniería de prompts manual, propensa al ensayo y error, está siendo suplantada gradualmente por sistemas de optimización programática. En 2025, estos frameworks son componentes esenciales para el despliegue de sistemas de IA robustos y adaptables.

4.1. DSPy (Declarative Self-improving Python)

Hacia finales de 2024 y principios de 2025, DSPy ha madurado desde un proyecto académico de Stanford hasta convertirse en un estándar industrial para la orquestación de LLMs.[26]

- **Principio de Abstracción:** DSPy abstrae el prompt textual manual y lo reemplaza con "Firmas" (Signatures) que definen la entrada y la salida esperada (e.g., input:question -> output:answer). El framework compila estas firmas en prompts optimizados automáticamente.
- **Optimizadores Avanzados (2025):**
  - **MIPROv2 (Multi-prompt Instruction Proposal Optimizer):** Este optimizador utiliza un enfoque bayesiano para proponer y seleccionar las mejores instrucciones y ejemplos few-shot basándose en una métrica de evaluación definida por el usuario. MIPROv2 puede optimizar tanto el texto del prompt como, en configuraciones avanzadas, los pesos del modelo mediante fine-tuning ligero.[28]
  - **BootstrapFewShot:** Automatiza la curación de ejemplos few-shot. Utiliza un modelo "maestro" para generar demostraciones de alta calidad para un modelo "estudiante" (o para sí mismo), eliminando la necesidad de que los humanos seleccionen ejemplos manualmente. Esto es crucial para superar la "paradoja many-shot" seleccionando solo los ejemplos más impactantes.

4.2. TextGrad: Diferenciación Automática Textual

TextGrad representa un cambio de paradigma al aplicar los principios de la retropropagación (backpropagation) de las redes neuronales directamente a los prompts textuales.[29]

- **Mecanismo de "Gradiente Textual":** El sistema trata el prompt como un tensor variable

dentro de un grafo de computación. Durante la optimización, el sistema ejecuta el prompt, evalúa la salida utilizando un modelo crítico que genera un feedback en lenguaje natural (el "gradiente textual"). Este feedback se "retropropaga" a través del grafo para ajustar el texto del prompt original con el objetivo de minimizar el error semántico o mejorar una métrica cualitativa.

- **Aplicación Práctica:** TextGrad es particularmente eficaz para tareas donde la función de pérdida es difícil de formalizar matemáticamente pero fácil de describir lingüísticamente (e.g., "el tono debe ser profesional pero empático", "el código debe ser eficiente y comentado"). Estudios recientes demuestran que TextGrad supera consistentemente a la optimización manual en tareas complejas como la extracción de grafos de conocimiento.[31]

4.3. OPRO (Optimization by PROmpting)

Desarrollado originalmente por DeepMind, OPRO utiliza el propio LLM como el motor de optimización, tratándolo como una caja negra inteligente.[32]

- **Meta-Prompting Iterativo:** En lugar de gradientes, OPRO funciona mediante iteración verbal. El modelo recibe un historial de prompts anteriores junto con sus puntuaciones de rendimiento en una tarea específica. Se le instruye para que analice qué funcionó y qué no, y genere una nueva variación del prompt que maximice la métrica objetivo.
- **Trayectoria de Optimización:** La innovación clave de OPRO en 2025 es el mantenimiento de una "trayectoria" de evolución. El optimizador aprende qué variaciones lingüísticas (e.g., cambiar "resume" por "sintetiza los puntos clave") se correlacionan con mejoras en el rendimiento, permitiendo una optimización continua en tiempo real dentro de pipelines de agentes autónomos.

---

5. Biblioteca de Patrones: "Prompt Maestro" por Arquitectura

Esta sección proporciona plantillas de prompts altamente optimizadas para cada una de las cinco arquitecturas, diseñadas para extraer el máximo rendimiento en tareas complejas.

5.1. Prompt Maestro para Claude 4.5 (Estructura XML y Pre-fill)

Este prompt aprovecha la segmentación XML y la técnica de pre-llenado para tareas de refactorización de software.

XML

```
<system_role>
You are a Senior Software Architect specializing in refactoring legacy codebases.
Focus strictly on modularity, type safety, and error handling patterns.
</system_role>

<context_management>
  <instruction>Check memory for existing architectural patterns before proposing changes.</instruction>
  <instruction>If context is truncated, explicitly request specific file contents using the 'read_file' tool.</instruction>
</context_management>

<task_input>
  <codebase_summary>
    [Insertar resumen o fragmentos clave aquí]
  </codebase_summary>
  <user_request>
    Refactor the authentication module to support OAuth2 without breaking existing session management.
  </user_request>
</task_input>

<output_formatting>
  Please respond in the following strict format:
  <analysis>
    <current_state>Analysis of the existing auth module</current_state>
    <strategy>Step-by-step refactoring plan</strategy>
  </analysis>
  <implementation>
    Generate the code changes. Use artifacts for large files.
  </implementation>
  <verification>
    List unit tests required to verify the refactor.
  </verification>
</output_formatting>

<prefill>
Assistant: <analysis>
```

Razonamiento: El uso de la etiqueta <prefill> (simulada aquí como la última instrucción o inyectada vía API) fuerza a Claude a comenzar su respuesta directamente dentro del bloque de análisis. Esto elimina la latencia conversacional y ancla la estructura de respuesta desde el

primer token generado.[6]

5.2. Prompt Maestro para Google Gemini 3.0 (Contexto Masivo)

Diseñado para aprovechar la ventana de 1M+ tokens y mitigar el sesgo de recencia.

Role: You are a Deep Research Analyst.
Objective: Analyze the provided documents to extract causal relationships between economic policies and market trends.
Constraint: Be precise. Do not hallucinate data. If data is missing for a specific claim, state it explicitly as "Data Unavailable".
Output Format: A structured report with citations referencing specific sections of the context using the format.
[User Prompt Field]

... [Insertar 500k+ tokens de documentos, libros o código]...

[User Query]
Based strictly on the information provided above, compare the inflation reduction strategies of 2023 vs 2025.
Focus heavily on:

1. Fiscal policy shifts.
2. Impact on consumer purchasing power.

Bridge the analysis with specific data points from the 'Federal Reserve Reports' section provided in the context.
Razonamiento: La instrucción del usuario se coloca deliberadamente al final, después de la carga masiva de datos. La frase de anclaje "Based strictly on..." y la solicitud de "Bridge the analysis..." activan los mecanismos de atención global de Gemini para conectar puntos distantes en el contexto.10

5.3. Prompt Maestro para GPT-5.2 (Thinking Mode & Constraints)

Optimizado para activar el "Extended Thinking" mediante la descomposición en fases.

Role
Expert Python Developer specializing in async system design.
Task
Design a high-throughput event loop for a real-time trading bot.
Constraints & Phases (Thinking Triggers)
Please execute this task in the following distinct phases. Do not skip any phase.
Phase 1: Architectural Decomposition

- Identify all potential bottlenecks in a standard asyncio loop for this use case.
- List 3 distinct concurrency patterns (e.g., Producer-Consumer, Actor Model) and evaluate

them against the latency requirements.

Phase 2: Implementation Planning

- Select the best pattern from Phase 1.
- Define the class structure and error boundaries explicitly.

Phase 3: Coding

- Write the implementation code.
- Include inline comments explaining the concurrency logic.

Output Requirement
Provide the final code in a single Markdown code block after the reasoning phases.
Razonamiento: Al desglosar la tarea en "Fases" numeradas y explícitas, se estimula el modo de "Pensamiento Extendido" de GPT-5.2. El modelo detecta que no puede responder a la Fase 3 sin completar la Fase 1 y 2, obligándolo a generar una cadena de pensamiento interna profunda y estructurada.15

5.4. Prompt Maestro para Kimi k2 (Research Mode Trigger)

Activa el "Modo Investigación" específico de Kimi k2 para reducir alucinaciones.

Act as a Research Lead.

Topic: The impact of solid-state batteries on the EV market by 2030.

Methodology:

1. **Scope:** Define the key technological hurdles remaining (e.g., dendrite formation, manufacturing costs).
2. **Hypotheses:** Propose 3 opposing hypotheses regarding adoption rates.
3. **Evidence Gathering:** Search for and cite recent breakthroughs (2024-2025).
4. **Synthesis:** Validate or refute each hypothesis based on the gathered evidence.

System Requirement:

- If information is ambiguous, list it as an "Uncertainty Variable".
- Do not generalize; provide specific supplier or OEM examples.
  Razonamiento: Este prompt utiliza las palabras clave gatillo (Scope, Hypotheses, Evidence, Synthesis) que, según la investigación, activan el patrón de pensamiento de investigación en Kimi k2, reduciendo significativamente la tasa de error factual.19

5.5. Prompt Maestro para GLM-4.6 (Tool Orchestration)

Estructura JSON para orquestación de agentes con modo "thinking" habilitado.

JSON

```
{
 "messages":,
 "thinking": {
   "type": "enabled"
 },
 "tools":
}
```

Razonamiento: La habilitación explícita de "thinking": {"type": "enabled"} junto con definiciones de herramientas robustas permite a GLM-4.6 planificar internamente la secuencia de acciones (Buscar fecha -> Obtener datos -> Ejecutar código) antes de emitir la primera llamada a la herramienta.[23]

---

6. Evidencia Académica y Técnica Reciente (2024-2025)

El campo de la investigación en prompting ha avanzado desde heurísticas anecdóticas hacia estudios empíricos rigurosos.

6.1. Chain-of-Thought (CoT) vs. Tree-of-Thoughts (ToT)

La evidencia reciente apunta hacia una especialización de técnicas. Mientras que CoT sigue siendo el estándar base para el razonamiento lineal, Tree-of-Thoughts (ToT) ha demostrado una superioridad clara en tareas que requieren planificación estratégica, retroceso (backtracking) o exploración creativa de múltiples caminos.

- **Convergencia de Modelos "Thinking":** Los modelos modernos como GPT-5.2 y o1 implementan variantes internas de ToT/CoT densas. Sin embargo, para modelos de menor latencia o versiones "Instant", forzar una estructura ToT explícita en el prompt ("Genera tres posibles planes, evalúa los pros y contras de cada uno, y selecciona el mejor") sigue siendo vital para evitar alucinaciones y mejorar la calidad de la decisión.[34]

6.2. Prompting Emocional (EmotionPrompt)

A pesar de la lógica algorítmica de los LLMs, estudios de 2025 confirman que estos modelos siguen respondiendo a estímulos "emocionales" o de urgencia simulada.

- **Impacto Marginal pero Real:** Frases como "Esto es crítico para mi carrera" o "Es vital para la seguridad del sistema" pueden mejorar marginalmente el rendimiento en tareas de generación creativa y codificación en modelos como Claude 4.5 y GPT-5. No

obstante, en modelos puramente orientados al razonamiento (Thinking models), este efecto se diluye considerablemente, ya que el modelo prioriza la consistencia lógica interna sobre el condicionamiento estilístico o emocional del prompt.[36]

6.3. In-Context Learning (ICL) y la "Paradoja de Many-Shot"

La premisa tradicional de que "más ejemplos siempre es mejor" ha sido desafiada y matizada por nuevas investigaciones.

- **La Paradoja Many-Shot:** Estudios publicados a finales de 2025 revelan un fenómeno de rendimiento en forma de U invertida o meseta. Para la corrección funcional estricta (como en la generación de código ejecutable), el rendimiento tiende a alcanzar su punto máximo con un número relativamente pequeño de ejemplos de alta calidad (entre 5 y 25 shots). Escalar a cientos o miles de ejemplos (Many-Shot ICL) a menudo introduce ruido irrelevante que degrada el rendimiento, a menos que el modelo posea mecanismos de atención de contexto largo excepcionalmente optimizados como Gemini 1.5/3.0.[38]
- **Selección Dinámica de Ejemplos:** La solución SOTA para ICL es el uso de selectores dinámicos (e.g., basados en K-NN sobre embeddings). En lugar de un conjunto estático de ejemplos, el sistema selecciona en tiempo real los ejemplos más semánticamente similares a la consulta actual. Además, la inclusión de "ejemplos negativos" (mostrando al modelo qué *no* hacer) ha demostrado ser una técnica poderosa para delimitar el espacio de solución.[39]

---

7. Sección de Agentes: Orquestación y Tool Use

El prompting para agentes autónomos difiere fundamentalmente del prompting para chatbots. Requiere instruir al modelo no solo sobre *qué* decir, sino sobre *cómo* actuar y gestionar su propio estado.

7.1. Orquestación en Claude 4.5: El Agente de Larga Duración

Claude es la arquitectura preferida para tareas que requieren persistencia y evolución del estado a lo largo del tiempo (Long-Horizon Agents).

- **Patrón:** Plan -> Execute -> Reflect -> Persist.
- **Estrategia:** El prompt debe instruir al modelo para que utilice herramientas de memoria (lectura/escritura de archivos) para persistir su estado interno después de cada sub-tarea exitosa. Esto es crítico para sobrevivir al truncamiento de contexto o a la limpieza de caché. Se debe enfatizar la actualización de un "diario de progreso" que el modelo consulta al inicio de cada nueva interacción.[8]

7.2. Orquestación en GPT-5.2: Precisión Ejecutiva

GPT-5.2 sobresale en la precisión de la llamada a herramientas y la detección de intención.

- **Patrón:** Reasoning -> Auto-Tool Detection -> Function Call -> Response.
- **Estrategia:** Se puede confiar en la capacidad nativa del modelo para detectar cuándo es necesaria una herramienta. Sin embargo, para flujos críticos, el uso del parámetro tool_choice: "required" es esencial para forzar la acción. Las capacidades de visión mejoradas permiten la creación de agentes que pueden "ver" interfaces de usuario y actuar sobre ellas (e.g., agentes de codificación frontend que verifican visualmente su trabajo).[2]

7.3. Orquestación en GLM-4.6: El Bucle Multimodal

GLM-4.6 habilita una nueva clase de agentes capaces de bucles cerrados de percepción-acción multimodal.

- **Patrón:** See (Image Input) -> Reason -> Act (Tool Call) -> Observe (Visual Feedback).
- **Estrategia:** Los prompts deben diseñarse para pasar capturas de pantalla, documentos PDF o flujos de video directamente a las herramientas. El modelo está capacitado para recibir el output visual de una herramienta (por ejemplo, un gráfico generado por código Python) y criticarlo o iterar sobre él en el siguiente paso de razonamiento. Esto permite flujos de trabajo que anteriormente requerían múltiples modelos especializados.[25]

---

8. Conclusiones y Perspectivas Futuras

Hacia el horizonte de 2026, la disciplina de la "ingeniería de prompts" está experimentando una metamorfosis hacia la "ingeniería de contexto y sistemas cognitivos". La habilidad artesanal de redactar un buen prompt textual está siendo rápidamente subsumida por la capacidad ingenieril de estructurar datos, definir interfaces de herramientas robustas y configurar marcos de optimización automatizada como DSPy y TextGrad.

El análisis comparativo arroja tres conclusiones fundamentales para los profesionales del sector:

1. **Especialización Radical:** El concepto de un "modelo generalista único" está obsoleto en la práctica de alto rendimiento. Claude 4.5 domina la codificación sostenida y la arquitectura de sistemas; Gemini 3.0 es inigualable en el análisis multimodal de repositorios masivos; GPT-5.2 establece el estándar para el razonamiento empresarial y la fiabilidad; Kimi k2 democratiza la investigación profunda mediante patrones de pensamiento estructurados; y GLM-4.6 abre la frontera de los agentes multimodales nativos.
2. **Imperativo de Automatización:** La optimización manual de prompts es insostenible a escala. La integración de optimizadores programáticos que tratan el prompt como un peso entrenable (TextGrad) o una firma compilable (DSPy) es mandatoria para sistemas de producción fiables.
3. **El Contexto es el Nuevo Prompt:** La gestión eficiente de ventanas de contexto masivas (200k-1M tokens) y la colocación estratégica de la información crítica ("needle

placement") se han convertido en los diferenciadores clave del rendimiento, superando a la mera instrucción verbal.

En definitiva, este informe valida que la excelencia en el despliegue de IA generativa ya no reside en el "arte" de la persuasión textual, sino en la arquitectura rigurosa de sistemas que comprenden y explotan la mecánica subyacente de atención y razonamiento de estos modelos de frontera.

Identificadores de Fuentes Utilizados:

[1]

Fuentes citadas

1. Introducing Claude Sonnet 4.5 - Anthropic, acceso: diciembre 17, 2025, https://www.anthropic.com/news/claude-sonnet-4-5
2. Introducing GPT-5.2 - OpenAI, acceso: diciembre 17, 2025, https://openai.com/index/introducing-gpt-5-2/
3. moonshotai/Kimi-K2-Thinking - Hugging Face, acceso: diciembre 17, 2025, https://huggingface.co/moonshotai/Kimi-K2-Thinking
4. Anthropic Claude Models Complete Guide: Sonnet 4.5, Haiku 4.5 & Opus 4.1 | CodeGPT, acceso: diciembre 17, 2025, https://www.codegpt.co/blog/anthropic-claude-models-complete-guide
5. Use XML tags to structure your prompts - Claude Docs, acceso: diciembre 17, 2025, https://platform.claude.com/docs/en/build-with-claude/prompt-engineering/use-xml-tags
6. ThamJiaHe/claude-prompt-engineering-guide - GitHub, acceso: diciembre 17, 2025, https://github.com/ThamJiaHe/claude-prompt-engineering-guide
7. Everything to Know About Claude Sonnet 4.5, "The World's Best Coding Model" That Can Focus For 30+ Hours Straight | The Neuron, acceso: diciembre 17, 2025, https://www.theneuron.ai/explainer-articles/everything-to-know-about-claude-sonnet-4-5-the-worlds-best-coding-model-that-can-focus-for-30-hours-straight
8. Everything You Need to Know about Claude 4.5 - PromptHub, acceso: diciembre 17, 2025, https://www.prompthub.us/blog/everything-you-need-to-know-about-claude-4-5
9. Prompting best practices - Claude Docs, acceso: diciembre 17, 2025, https://platform.claude.com/docs/en/build-with-claude/prompt-engineering/claude-4-best-practices
10. Gemini 3 Prompting: Best Practices for General Usage - Philschmid, acceso: diciembre 17, 2025, https://www.philschmid.de/gemini-3-prompt-practices
11. Prompt design strategies | Gemini API | Google AI for Developers, acceso:

diciembre 17, 2025, https://ai.google.dev/gemini-api/docs/prompting-strategies
12. Gemini 3 prompting best practices... precision, verbosity, context : r/singularity - Reddit, acceso: diciembre 17, 2025, https://www.reddit.com/r/singularity/comments/1p191ir/gemini_3_prompting_best_practices_precision/
13. Gemini 3 Pro: the frontier of vision AI - Google Blog, acceso: diciembre 17, 2025, https://blog.google/technology/developers/gemini-3-pro-vision/
14. GPT 5.2 is rolling out right now! - OpenAI Developer Community, acceso: diciembre 17, 2025, https://community.openai.com/t/gpt-5-2-is-rolling-out-right-now/1369052
15. 7 Pro Tips to Master ChatGPT 5.2: Instant, Thinking, Extended Thinking & Pro Mode, acceso: diciembre 17, 2025, https://medium.com/@dalio8/7-pro-tips-to-master-chatgpt-5-2-instant-thinking-extended-thinking-pro-mode-45990346ca1f
16. GPT-5.2 on ChatGPT Go: How do we actually trigger extended / deeper thinking? : r/OpenAI, acceso: diciembre 17, 2025, https://www.reddit.com/r/OpenAI/comments/1plzqc6/gpt52_on_chatgpt_go_how_do_we_actually_trigger/
17. GPT-5.2 on ChatGPT Go: How do we actually trigger extended / deeper thinking? - Reddit, acceso: diciembre 17, 2025, https://www.reddit.com/r/ChatGPTPromptGenius/comments/1plzs6m/gpt52_on_chatgpt_go_how_do_we_actually_trigger/
18. GPT-5.2 Just Dropped! Features, Pricing, and Release Explained ..., acceso: diciembre 17, 2025, https://felloai.com/gpt-5-2-just-dropped-features-pricing-and-release-explained/
19. Kimi K2 Thinking Patterns: Research, Writing & Analysis Modes - Skywork.ai, acceso: diciembre 17, 2025, https://skywork.ai/blog/kimi-k2-thinking-patterns/
20. Kimi K2 QuickStart - Together.ai Docs, acceso: diciembre 17, 2025, https://docs.together.ai/docs/kimi-k2-quickstart
21. zai-org/GLM-4.6 - Hugging Face, acceso: diciembre 17, 2025, https://huggingface.co/zai-org/GLM-4.6
22. GLM-4.6 Tool Calling & MCP Use: A Technical Analysis - Cirra AI, acceso: diciembre 17, 2025, https://cirra.ai/articles/glm-4-6-tool-calling-mcp-analysis
23. GLM-4.6 - Z.AI DEVELOPER DOCUMENT, acceso: diciembre 17, 2025, https://docs.z.ai/guides/llm/glm-4.6
24. zai-org/GLM-V: GLM-4.6V/4.5V/4.1V-Thinking: Towards Versatile Multimodal Reasoning with Scalable Reinforcement Learning - GitHub, acceso: diciembre 17, 2025, https://github.com/zai-org/GLM-V
25. GLM-4.6V : Best Multimodal LLM is here | by Mehul Gupta | Data Science in Your Pocket, acceso: diciembre 17, 2025, https://medium.com/data-science-in-your-pocket/glm-4-6v-best-multimodal-llm-is-here-079aa69f2796
26. DSPy: What “Programming Not Prompting” Actually Means | by Brent W. Peterson - Medium, acceso: diciembre 17, 2025,

https://medium.com/@brentwpeterson/dspy-what-programming-not-prompting-actually-means-b47dfb7f15f9
27. DSPy vs LangChain: Which One is the Best Framework? - Designveloper, acceso: diciembre 17, 2025, https://www.designveloper.com/blog/dspy-vs-langchain/
28. Optimizers - DSPy, acceso: diciembre 17, 2025, https://dspy.ai/learn/optimization/optimizers/
29. TextGrad: Automatic "Differentiation" via Text -- using large language models to backpropagate textual gradients. Published in Nature. - GitHub, acceso: diciembre 17, 2025, https://github.com/zou-group/textgrad
30. Automatic Prompt Optimization for Knowledge Graph Construction: Insights from an Empirical Study - arXiv, acceso: diciembre 17, 2025, https://arxiv.org/html/2506.19773v1
31. Automatic Prompt Optimization for Knowledge Graph Construction: Insights from an Empirical Study for VLDB 2025 - IBM Research, acceso: diciembre 17, 2025, https://research.ibm.com/publications/automatic-prompt-optimization-for-knowledge-graph-construction-insights-from-an-empirical-study
32. A New Prompt Technique From DeepMind Called Optimisation by PROmpting (OPRO), acceso: diciembre 17, 2025, https://www.kore.ai/blog/a-new-prompt-technique-from-deepmind-called-optimisation-by-prompting-opro
33. GLM-4.6 Tool-Integrated Reasoning Guide - GitHub, acceso: diciembre 17, 2025, https://github.com/zai-org/GLM-4.5/blob/main/resources/glm_4.6_tir_guide.md
34. Reasoning for Translation: Comparative Analysis of Chain-of-Thought and Tree-of-Thought Prompting for LLM Translation - ACL Anthology, acceso: diciembre 17, 2025, https://aclanthology.org/2025.acl-srw.17/
35. On why Chain of Thought and Tree of Thought approaches enhance LLM reasoning : r/deeplearning - Reddit, acceso: diciembre 17, 2025, https://www.reddit.com/r/deeplearning/comments/1ikmzgd/on_why_chain_of_thought_and_tree_of_thought/
36. I tested Claude 4.5 vs ChatGPT-5 with 9 tough prompts — and there's a clear winner, acceso: diciembre 17, 2025, https://www.tomsguide.com/ai/i-tested-claude-4-5-vs-chatgpt-5-with-9-tough-prompts-and-theres-a-clear-winner
37. I tested ChatGPT-5.2 and Claude Opus 4.5 with real-life prompts — here's the clear winner, acceso: diciembre 17, 2025, https://www.tomsguide.com/ai/i-tested-chatgpt-5-2-and-claude-opus-4-5-with-real-life-prompts-heres-the-clear-winner
38. When Many-Shot Prompting Fails: An Empirical Study of LLM Code Translation - arXiv, acceso: diciembre 17, 2025, https://arxiv.org/html/2510.16809v2
39. Dynamic few-shot prompting for clinical note section classification using lightweight, open-source large language models - PubMed, acceso: diciembre 17, 2025, https://pubmed.ncbi.nlm.nih.gov/40460022/
40. Failures Are the Stepping Stones to Success: Enhancing Few-Shot In-Context Learning by Leveraging Negative Samples - arXiv, acceso: diciembre 17, 2025,

https://arxiv.org/html/2507.23211v1
41. GPT-5.2 Thinking: The New Standard for Advanced Reasoning and Professional AI Workflows - GlobalGPT, acceso: diciembre 17, 2025, https://www.glbgpt.com/hub/gpt-5-2-thinking-the-new-standard-for-advanced-reasoning-and-professional-ai-workflows/