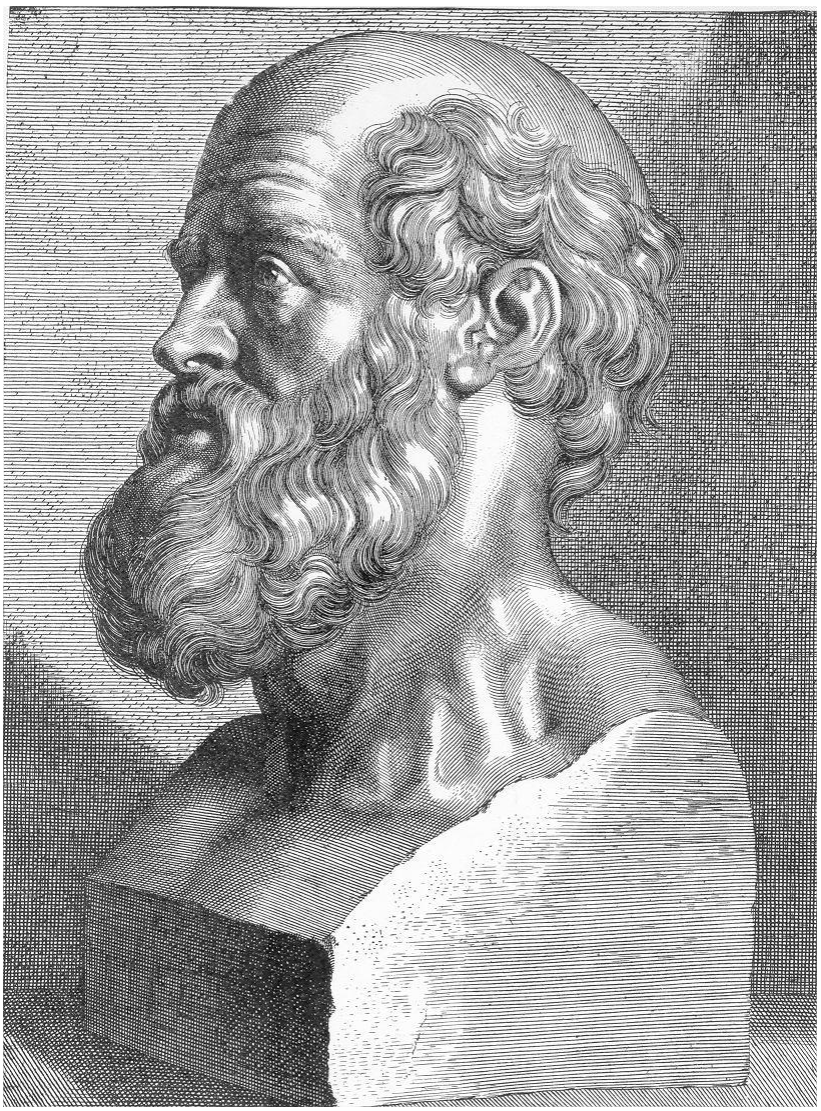
Cálculo como argumento clínico:

Hipócrates como sistema de apoyo computacional formalizado

Julio David Rojas
4/8/2026
Ensayo académico



Resumen

El presente ensayo examina Hipócrates, un motor de apoyo clínico computable estructurado en torno a un principio que rara vez se explicita en los sistemas de este tipo: toda inferencia válida sobre datos clínicos requiere, antes de ser ejecutada, que esos datos sean formalmente posibles. Hipócrates no es un sistema de inteligencia artificial generativa ni un asistente conversacional médico. Es una arquitectura determinista que, mediante capas secuenciales de validación de estructura, validación de dominio, cómputo formalizado y auditoría criptográfica, intenta hacer computables ciertas operaciones parciales de razonamiento clínico sin pretender suplantar el juicio del médico. El ensayo analiza su arquitectura conceptual y operatoria, los cinco módulos clínicos actuales —razonamiento bayesiano secuencial, interpretación gasométrica ácido-base, análisis de curva de decisión, farmacocinética monocompartimental y estratificación de sepsis—, su régimen de validación y trazabilidad, y sus límites reales. La tesis central sostiene que el valor de Hipócrates reside menos en la sofisticación de sus cálculos que en el rigor con el que delimita qué puede calcular y qué no, y en la honestidad con que lo comunica. Ese doble movimiento —cómputo preciso y prudencia semántica explícita— es lo que convierte a este prototipo en algo más que una demostración técnica.

Palabras clave: apoyo clínico computable, razonamiento bayesiano secuencial, farmacocinética clínica, análisis de curva de decisión, trazabilidad auditada, arquitectura determinista, prudencia semántica.

Hipócrates es un sistema de apoyo clínico computable concebido para ejecutar de forma ordenada, verificable y trazable ciertas operaciones parciales del razonamiento médico. Su propósito no es reemplazar al clínico ni simular una inteligencia autónoma, sino volver formalmente tratables algunos problemas que, en la práctica, suelen resolverse de manera dispersa o poco transparente: la actualización secuencial de probabilidades diagnósticas, la interpretación de alteraciones ácido-base, la evaluación de utilidad clínica de modelos predictivos, la estimación farmacocinética orientativa y la estratificación de gravedad en sepsis. En ese sentido, su interés principal no radica únicamente en las fórmulas que implementa, sino en el modo en que delimita con rigor qué puede calcular, bajo qué condiciones puede hacerlo y hasta dónde llegan sus resultados.

El sistema opera a partir de una secuencia sencilla, aunque conceptualmente exigente. Primero recibe datos clínicos estructurados; después comprueba que esos datos tengan forma coherente y que sus valores no sean físicamente absurdos; solo entonces ejecuta el cálculo correspondiente y devuelve una salida homogénea, legible y acompañada de trazabilidad. Este orden importa más de lo que parece. Un cálculo puede ser correcto en términos matemáticos y, sin embargo, resultar clínicamente inútil si se apoya en datos imposibles, mal definidos o insuficientes. Hipócrates intenta evitar precisamente ese tipo de falsa precisión: prefiere bloquear una entrada inviable antes que producir una respuesta elegante sobre bases erróneas.

Lo decisivo, por tanto, es que Hipócrates no convierte el cálculo en mandato. Sus resultados no deben leerse como órdenes clínicas, sino como apoyos formales para una deliberación que sigue perteneciendo al médico. De ahí que su valor no consista en prometer autonomía, sino en hacer explícita la diferencia entre calcular y decidir. **Como prototipo de investigación**, su aportación es metodológica antes que triunfalista: muestra que el apoyo computacional serio en medicina exige precisión técnica, límites claros y una forma de honestidad poco frecuente en este campo, la honestidad de no decir más de lo que realmente puede demostrarse.

1. Introducción

Hay una asimetría curiosa en la medicina computacional contemporánea: los sistemas más ruidosos suelen ser los que menos saben lo que hacen. Se presentan con jerga de inteligencia artificial, prometen reducir el error diagnóstico, afirman integrar miles de variables, y producen recomendaciones que parecen más seguras cuanto menos se examinan sus fundamentos. En ese paisaje, la pregunta que debería hacerse antes que cualquier otra no es qué calcula el sistema, sino qué sabe el sistema sobre los límites de su propio cálculo.

Hipócrates parte de esa pregunta. No como retórica, sino como decisión de diseño. Cada módulo del sistema lleva inscrita en su código fuente una advertencia explícita: «Motor de apoyo computacional. No usar en decisiones clínicas autónomas.» Esa frase podría leerse como cobertura legal. En el contexto de este sistema, es algo más:

es la expresión de una arquitectura que intenta ser honesta sobre lo que puede y no puede hacer.

El problema que Hipócrates intenta resolver no es nuevo. Desde hace décadas existe el reconocimiento de que ciertas operaciones del razonamiento médico —actualizar probabilidades diagnósticas con resultados de pruebas, interpretar equilibrios ácido-base, estimar concentraciones plasmáticas de fármacos, estratificar la gravedad de una sepsis— son, en principio, computables. Sus fórmulas están en la literatura, sus parámetros son medibles, sus resultados son reproducibles. El obstáculo no es la complejidad matemática, sino algo más difícil: hacer que esas operaciones sean ejecutables de manera formal, consistente y auditable, sin mezclar lo que el cálculo produce con lo que el clínico debe decidir.

La mayoría de los sistemas existentes colapsan esas dos cosas. Calculan y deciden al mismo tiempo, sin marcar con claridad dónde termina el cómputo y dónde empieza la interpretación. El resultado es una opacidad funcional: el sistema produce un output que parece una recomendación, pero no se puede verificar si ese output se basa en un cálculo correcto, si los inputs eran físicamente posibles, o si la acción sugerida es una clasificación computacional o una instrucción clínica.

Hipócrates intenta resolver ese colapso mediante una arquitectura de capas. Cada capa hace una cosa distinta y sólo esa cosa: el esquema de entrada verifica que el payload tiene estructura correcta; el gate de validación verifica que los valores son físicamente posibles; el módulo clínico ejecuta el cálculo correspondiente; el registro de auditoría documenta qué se calculó, cuándo y con qué inputs. La salida es siempre del mismo tipo —una estructura homogénea con campos fijos— y la acción que emite es siempre una clasificación computacional, nunca una orden autónoma.

Este ensayo analiza esa arquitectura, sus módulos actuales, su régimen de validación y trazabilidad, y sus límites reales. El argumento no es que Hipócrates sea un sistema clínico listo para producción. Es que su manera de delimitar el problema —qué puede calcularse formalmente, qué no, y cómo documentar la diferencia— constituye un enfoque serio y metodológicamente relevante para pensar cómo debe diseñarse el apoyo computacional a la decisión médica.

2. Planteamiento del problema y delimitación

El problema de fondo en los sistemas de apoyo clínico no es técnico. Es semántico. ¿Qué significa exactamente que un sistema «recomiende» un tratamiento? ¿Qué diferencia hay entre un cálculo que sitúa la probabilidad posterior de un diagnóstico en 0.87 y una recomendación de tratar? ¿Es lo mismo un output que dice «review_dosing» que una prescripción? La mayoría de los sistemas que circulan bajo el nombre de apoyo a la decisión clínica no responden esas preguntas. Las evaden con una advertencia legal en letra pequeña y continúan comportándose como si sus outputs fueran instrucciones.

Hipócrates intenta responder esas preguntas desde el diseño. Para eso, distingue con precisión tres cosas que habitualmente se confunden: el cálculo clínico, el apoyo a la decisión y la prescripción o autonomía clínica. El cálculo clínico es lo que Hipócrates hace: dado un conjunto de inputs validados, ejecuta un algoritmo formalizado y produce un resultado numérico o categórico. El apoyo a la decisión es lo que ese resultado puede ofrecer al clínico: información estructurada que reduce la carga cognitiva de una operación parcial de razonamiento. La prescripción o autonomía clínica es lo que Hipócrates explícitamente rechaza hacer: tomar decisiones sobre el paciente real, incorporar el contexto clínico completo, responder por las consecuencias de un acto médico.

Esa distinción tiene consecuencias concretas. Cuando el módulo bayesiano emite la acción «start_treatment», no está ordenando iniciar un tratamiento. Está señalando que, dado el modelo de actualización de probabilidades aplicado con los parámetros que el clínico introdujo, la probabilidad posterior ha cruzado el umbral que el propio clínico definió como criterio de tratamiento. La decisión de qué umbral usar, si los parámetros son apropiados para ese paciente, y si la acción computacional se traduce en una intervención real, recae enteramente sobre el médico. El sistema calcula. El médico decide.

Esto también delimita lo que Hipócrates no pretende resolver. No accede a historiales clínicos electrónicos. No evalúa el contexto epidemiológico del paciente en tiempo real. No incorpora la anamnesis, la exploración física ni el juicio experto sobre la presentación atípica. No aprende de los casos que procesa. No se adapta a ningún paciente individual más allá de los parámetros que se le introducen manualmente en cada llamada. Su modelo de datos es un payload estático por sesión.

Calificar esto como limitación es exacto pero parcialmente engañoso. Toda herramienta tiene un dominio. Un tensiómetro no diagnostica la causa de la hipertensión; simplemente mide la presión. Eso no lo convierte en un instrumento incompleto: lo convierte en un instrumento preciso dentro de su dominio. Hipócrates intenta ser preciso dentro de su dominio —el cómputo formal de operaciones parciales— y honesto sobre sus fronteras. Que eso baste o no para una aplicación clínica real es una pregunta distinta, que este ensayo abordará en la sección de límites.

La categoría conceptual más adecuada para Hipócrates es la de prototipo de investigación. En el contexto del sistema, esa categoría no es modestia retórica: es una descripción técnica precisa. El proyecto implementa 187 tests automatizados que verifican el comportamiento correcto de cada módulo, declara explícitamente qué versión de cada componente está activa (v1, v2), documenta en el código las limitaciones de cada modelo, y distingue entre lo que está implementado y lo que está previsto pero aún no existe. Esa transparencia sobre el estado real del sistema es, en sí misma, una decisión metodológica relevante.

3. Arquitectura conceptual y operatoria de Hipócrates

Cualquier sistema computacional puede describirse en términos de lo que recibe, lo que hace y lo que devuelve. La particularidad de Hipócrates es que esas tres fases no son simplemente funciones encadenadas: son capas con contratos formales y responsabilidades separadas. Entender la arquitectura del sistema es entender por qué esa separación importa.

3.1 El esquema de entrada

Toda solicitud al sistema comienza con la validación del payload de entrada. El esquema —implementado en el componente Clinical_IO_Schema— exige que cualquier llamada contenga exactamente cinco campos: un identificador de paciente, el nombre del módulo a ejecutar, los inputs clínicos como diccionario, las restricciones opcionales y la versión del esquema. Si falta alguno de esos campos, o si el módulo solicitado no está entre los cinco reconocidos por el sistema, la solicitud se rechaza antes de que se ejecute ningún cálculo.

Eso parece obvio hasta que se considera por qué importa. Un sistema que no valida la estructura de sus entradas puede ejecutar cálculos sobre datos malformados y producir resultados que parecen válidos pero derivan de inputs incorrectos. La validación de esquema garantiza que el sistema sólo opera sobre datos que tienen la forma esperada. No garantiza que esos datos sean clínicamente correctos —eso es responsabilidad de la capa siguiente— pero sí garantiza que tienen estructura coherente.

La versión del esquema también cumple una función: el sistema sólo acepta payloads cuya versión comience con el prefijo «SMNC-5+». Eso permite detectar llamadas que provienen de versiones antiguas del cliente o de integraciones que no han sido actualizadas, antes de que esas llamadas produzcan resultados potencialmente incorrectos con una versión de contrato diferente.

3.2 El gate de dominio

La segunda capa es, en muchos sentidos, la más relevante desde el punto de vista clínico. El Units_Validity_Gate no verifica si los inputs tienen estructura correcta —eso ya lo hizo la capa anterior— sino si los valores numéricos son físicamente posibles dentro del dominio de cada módulo.

Las reglas son concretas. Para el módulo ácido-base, el pH debe estar entre 6.5 y 8.0; la PaCO2 entre 5 y 120 mmHg; el bicarbonato entre 1 y 60 mEq/L. Para el módulo bayesiano, la probabilidad pretest debe estar estrictamente entre cero y uno: una probabilidad de exactamente cero o de exactamente uno viola los fundamentos matemáticos de la actualización bayesiana, porque convierte los odds en cero o en infinito. Para el módulo farmacocinético, el clearance, el volumen de distribución y las dosis deben ser estrictamente positivos. Para el módulo de sepsis, la frecuencia respiratoria y la presión arterial sistólica no pueden ser cero ni negativos.

Si cualquiera de esas reglas se viola —si un solo campo está fuera de dominio— el gate bloquea la solicitud completa con la acción «blocked», lista todas las violaciones detectadas, registra ese bloqueo en el log de auditoría y devuelve el resultado sin ejecutar ningún módulo clínico. El campo «units_ok» en el output adopta el valor falso, señalando inequívocamente que ningún cálculo se produjo.

La razón de diseño es directa: un sistema que calcula sobre datos físicamente imposibles no produce resultados con error; produce resultados sin sentido. Un pH de 9.2 no es un valor erróneo que el sistema deba manejar con gracia: es una señal de que algo está mal antes de la consulta. Permitir que el cálculo continúe sobre ese dato sería producir una salida que parece precisa pero está basada en un supuesto fisiológicamente absurdo.

3.3 El orquestador

Una vez validados el esquema y el dominio, el orquestador despacha la solicitud al módulo correspondiente. El pipeline es lineal y explícito: schema → gate → módulo → auditoría → output. No hay ramificaciones ocultas, no hay lógica distribuida en múltiples lugares. El orquestador es el único punto de entrada al sistema; ningún componente externo llama directamente a los módulos clínicos.

Esa centralización tiene una ventaja que no es obvia a primera vista: garantiza que toda solicitud pasa por las mismas capas de validación, sin excepción. No existe un camino alternativo para llamar al módulo bayesiano saltándose el gate de dominio. La única forma de omitir el gate es mediante un parámetro explícito de omisión que, según la documentación del código, está reservado exclusivamente para los tests internos y que el propio código etiqueta con la advertencia «Nunca en producción». Esa restricción arquitectónica no depende de la disciplina del programador: está inscrita en la estructura del sistema.

3.4 La salida homogénea

Todos los módulos de Hipócrates devuelven exactamente el mismo tipo de objeto: un ClinicalOutput con ocho campos fijos. El campo «result» contiene los datos específicos del cálculo; los campos «action», «p», «U», «NB», «units_ok», «explain» y «ci» son estructuralmente idénticos en todos los módulos, independientemente de qué hayan calculado.

La homogeneidad no es un detalle de implementación: es una declaración conceptual. Significa que cualquier interfaz que consuma outputs de Hipócrates puede hacerlo sin necesidad de conocer qué módulo específico generó ese output, porque la estructura siempre será la misma. El campo «action» siempre contendrá uno de los verbos canónicos definidos en el sistema. El campo «units_ok» siempre indicará si el gate de

dominio fue superado. El campo «request_id» siempre permitirá localizar el registro de auditoría correspondiente.

Esa coherencia estructural también tiene una función epistémica: impide que el usuario interprete incorrectamente una salida como si fuera de un módulo distinto. Si todos los módulos devuelven el mismo formato, el consumidor del output debe mirar el campo «action» y entender qué tipo de clasificación computacional acaba de recibir, sin confundir una recomendación de revisión farmacológica con una probabilidad diagnóstica.

4. Los módulos clínicos actuales y su lógica propia

Hipócrates cuenta en su versión actual con cinco módulos clínicos. Cada uno implementa un dominio distinto del razonamiento médico formalizable, con sus propias fórmulas, sus propias invariantes de dominio y sus propias acciones canónicas. No son intercambiables: cada módulo tiene una pregunta específica que puede responder y una respuesta específica que no puede dar.

4.1 Razonamiento bayesiano secuencial con parada temprana

El primer módulo implementa el procedimiento secuencial de prueba de hipótesis de Wald —conocido por sus siglas en inglés SPRT— aplicado al razonamiento diagnóstico. La pregunta que responde este módulo es: dada una probabilidad pretest del diagnóstico de interés, y dado un conjunto ordenado de resultados de pruebas diagnósticas, ¿cuál es la probabilidad posterior tras incorporar esas pruebas secuencialmente? ¿Supera esa probabilidad el umbral que justificaría iniciar tratamiento? ¿Cae por debajo del umbral que justificaría descartar el diagnóstico?

El mecanismo es el siguiente. Antes de aplicar ninguna prueba, el clínico estima la probabilidad pretest del diagnóstico: cuánto, en una escala de cero a uno, cree que ese diagnóstico es probable en ese paciente antes de revisar resultados. Esa probabilidad se convierte en odds mediante la transformación estándar. Luego, por cada prueba disponible, se multiplican los odds por la razón de verosimilitud de esa prueba —un número que expresa cuánto más probable es el resultado observado si el diagnóstico es verdadero que si es falso. Los odds resultantes se reconvierten en probabilidad. Después de cada paso, el sistema verifica si esa probabilidad ha

cruzado alguno de los dos umbrales definidos por el clínico: el umbral de tratamiento y el umbral de descarte.

Si la probabilidad cruza alguno de esos umbrales, el loop se detiene inmediatamente. Las pruebas restantes no se procesan. Este no es un defecto de implementación: es la característica definitoria del SPRT. Una vez que la evidencia acumulada es suficiente para tomar una decisión —en cualquier dirección— seguir incorporando pruebas no añade información que cambie esa decisión. El sistema registra qué pruebas fueron procesadas, qué pruebas quedaron sin procesar y en qué paso se cruzó el umbral. Si el loop completa todas las pruebas disponibles sin cruzar ningún umbral, la acción resultante es «obtain_test»: la evidencia es insuficiente para decidir y se necesitan más datos.

El módulo expone su propia limitación en el código: asume independencia condicional entre pruebas. Si dos pruebas diagnósticas están correlacionadas —si un resultado positivo en una hace más probable un positivo en la otra, independientemente del diagnóstico— el modelo producirá una actualización sesgada. Para esos casos, se debería usar una razón de verosimilitud condicional, que este módulo no implementa. Esa limitación está documentada explícitamente en el docstring del módulo. No está escondida.

4.2 Interpretación gasométrica ácido-base

El segundo módulo interpreta la gasometría arterial mediante una combinación de dos marcos fisicoquímicos complementarios. El primero es la ecuación de Henderson-Hasselbalch, que relaciona el pH con la presión parcial de CO2 arterial (PaCO2) y la concentración de bicarbonato (HCO3-). Esa ecuación permite calcular el pH teórico a partir de los otros dos valores, lo que el sistema usa para verificar la consistencia interna del triplete de valores introducido: si el pH medido difiere en más de 0.05 unidades del pH calculado, el sistema reporta una inconsistencia que puede indicar error de transcripción o que los valores provienen de muestras distintas.

A partir de esos datos básicos, el módulo calcula una serie de parámetros derivados. La brecha aniónica —definida como sodio menos la suma de cloruro y bicarbonato— estima la concentración de aniones ácidos no medidos en plasma. Un valor elevado puede indicar presencia de lactato, cetonas, tóxicos o productos de retención urémica.

La corrección de esa brecha por albúmina es particularmente importante: la hipoalbuminemia reduce artificialmente la brecha aniónica medida, enmascarando acidosis que de otro modo pasarían desapercibidas. El delta-delta, a su vez, examina si en presencia de una brecha aniónica elevada existe también un trastorno metabólico superpuesto —alcalosis o acidosis sin brecha— lo que señalaría un cuadro mixto.

El segundo marco es el de Stewart, que analiza el equilibrio ácido-base desde la perspectiva de la diferencia iónica fuerte: la diferencia entre cationes y aniones completamente disociados en el plasma. El sistema calcula una aproximación de esa diferencia —denominada SIDa— y una estimación de los ácidos débiles no volátiles a partir de la albúmina y el fosfato. La documentación del módulo aclara explícitamente que esta es una aproximación cualitativa: la solución completa del modelo de Stewart requeriría resolver numéricamente un sistema de ecuaciones no lineales que este módulo no implementa. Esa limitación está declarada tanto en el código como en el output, campo a campo.

El diagnóstico resultante clasifica el trastorno primario —acidosis o alcalosis, metabólica o respiratoria— y evalúa el estado de compensación mediante las fórmulas estándar: Winter para la compensación respiratoria en acidosis metabólica, y las fórmulas de compensación renal aguda y crónica para la acidosis respiratoria. La acción que emite este módulo es siempre «obtain_test»: el resultado es información para correlacionar con el cuadro clínico, no una decisión terapéutica autónoma.

4.3 Análisis de curva de decisión

El tercer módulo implementa el Análisis de Curva de Decisión, un método estadístico para evaluar la utilidad clínica de modelos predictivos o diagnósticos. La pregunta que responde es distinta a la de los módulos anteriores: no qué diagnóstico es más probable, sino si tiene sentido usar un determinado modelo predictivo para tomar decisiones, dado un umbral de riesgo que el clínico considera relevante.

El concepto central es el beneficio neto: la diferencia entre los verdaderos positivos que el modelo identifica correctamente y los falsos positivos que genera, ponderados por el coste relativo de tratar a alguien que no necesita tratamiento versus no tratar a

alguien que sí lo necesita. Ese balance depende del umbral de decisión —la probabilidad mínima del evento que justifica intervenir— y ese umbral, a su vez, codifica el juicio clínico sobre las consecuencias de cada tipo de error.

El módulo calcula esa función sobre una grilla de umbrales dentro del rango clínicamente relevante definido por el usuario, y la compara con dos estrategias de referencia: tratar a todos los pacientes sin discriminación, y no tratar a ninguno. Si el modelo genera más beneficio neto que ambas estrategias en el umbral de referencia, la acción es «use_model». Si sólo lo supera en parte del rango evaluado, la acción es «restrict_to_threshold_range», con indicación de cuál es ese subconjunto útil. Si el modelo no supera ninguna estrategia alternativa, la acción es «do_not_use_model».

Lo que este módulo evalúa no es la exactitud del modelo en abstracto, sino su utilidad para tomar decisiones bajo condiciones específicas. Un modelo con alta sensibilidad y baja especificidad puede ser útil cuando el umbral de decisión es bajo —cuando el coste de perder un caso verdadero es alto— y completamente inútil cuando el umbral es alto. Esa dependencia del contexto es precisamente lo que la curva de decisión visualiza y cuantifica.

4.4 Farmacocinética monocompartimental y monitoreo terapéutico

El cuarto módulo es el más extenso del sistema. En su versión actual —designada v2— implementa diez modos de cálculo farmacocinético, todos basados en el modelo de un compartimento: una simplificación que asume que el fármaco se distribuye instantáneamente y de manera homogénea en un volumen único, y que su eliminación sigue una cinética de primer orden proporcional a la concentración plasmática.

Los siete modos heredados de la versión anterior cubren los casos más frecuentes del análisis farmacocinético básico: la concentración plasmática tras un bolo intravenoso; la concentración durante una infusión continua y en estado estacionario; la concentración máxima y mínima en estado estacionario con dosis repetidas; la absorción oral mediante la ecuación de Bateman; el cálculo inverso de dosis de carga y mantenimiento para alcanzar una concentración objetivo; la simulación paso a paso de la cinética no lineal de la fenitoína mediante el modelo de Michaelis-Menten; y el ajuste proporcional de dosis en función de la depuración renal del paciente.

Los tres modos nuevos de la versión 2 añaden capacidades específicas para el monitoreo terapéutico individualizado. El primero estima la depuración de creatinina mediante la fórmula de Cockcroft-Gault a partir de la edad, el sexo, el peso y la creatinina sérica del paciente: un cálculo de uso rutinario en la práctica clínica para ajustar dosis en pacientes con función renal comprometida. El segundo combina ese cálculo con el modo de dosis objetivo, permitiendo ajustar automáticamente el clearance estimado del fármaco en función de la función renal del paciente. El tercero implementa una estimación bayesiana MAP —del inglés Maximum A Posteriori— que, dados uno o varios niveles séricos observados del fármaco y priors sobre los parámetros farmacocinéticos del paciente, estima los parámetros individualizados más probables mediante optimización por sección dorada.

El sistema aclara con precisión lo que ese último modo es y lo que no es. Es una estimación bayesiana básica de un modelo de un compartimento, con priors log-normales y optimización determinista. No es equivalente a un software de monitoreo terapéutico validado clínicamente, que usaría modelos poblacionales específicos por fármaco, métodos de integración MCMC completos, y validación prospectiva en poblaciones reales. La acción que todos los modos de este módulo emiten es «review_dosing»: el resultado es una estimación orientativa que requiere confirmación con niveles séricos y criterio clínico.

4.5 Estratificación de sepsis

El quinto módulo implementa los criterios de Sepsis-3 —publicados por Singer y colaboradores en 2016 en JAMA— adaptados para su aplicación computacional parcial. La sepsis se define en ese marco como una disfunción orgánica potencialmente mortal causada por una respuesta desregulada del huésped a una infección; el choque séptico se define como sepsis con requerimiento de vasopresores para mantener una presión arterial media superior a 65 mmHg y lactato sérico mayor de 2 mmol/L en ausencia de hipovolemia.

El módulo calcula tres instrumentos de estratificación. El qSOFA —una herramienta de cribado rápido— puntúa tres componentes: frecuencia respiratoria elevada (mayor o igual a 22 respiraciones por minuto), presión arterial sistólica baja (menor o igual a 100 mmHg) y alteración del estado mental. Una puntuación de dos o más indica cribado positivo. El SOFA —Sequential Organ Failure Assessment— evalúa la

disfunción orgánica en seis sistemas; en este módulo se calculan cuatro de esos sistemas a partir de los datos disponibles —renal por creatinina, hepático por bilirrubina, de coagulación por plaquetas, y respiratorio por el cociente PaO2/FiO2— y se indica explícitamente qué componentes no se evalúan: la puntuación cardiovascular formal y la puntuación neurológica por escala de Glasgow, que requieren datos que el módulo no recoge en esta versión. El lactato sérico, finalmente, se clasifica en cuatro niveles de alerta según su concentración.

La clasificación resultante ubica al paciente en una de tres categorías de severidad: baja sospecha, sepsis probable o choque séptico probable. La acción depende del nivel: «observe» para baja sospecha, «obtain_test» para sepsis probable con datos insuficientes, y «start_treatment» para choque séptico probable. La documentación del módulo aclara que «start_treatment» en este contexto significa que el output computacional apoya iniciar manejo clínico inmediato, y que eso no constituye una orden autónoma de tratamiento. El módulo también declara explícitamente qué no hace: no accede a historial clínico, no prescribe antibióticos ni dosis específicas, no evalúa respuesta a fluidos ni foco infeccioso, no predice mortalidad.

5. Validación, trazabilidad y prudencia del sistema

Hipócrates tiene un problema de comunicación que cualquier sistema de apoyo computacional enfrenta: cómo garantizar que quien consume sus outputs entiende exactamente qué producen esos outputs y qué no producen. Esa garantía no puede resolverse solo con un texto de advertencia en la documentación. Requiere decisiones de diseño que la hagan estructuralmente difícil de ignorar.

La primera de esas decisiones es el gate de dominio, ya descrito. Bloquear activamente las solicitudes con inputs físicamente imposibles no es sólo una verificación de calidad: es una afirmación sobre la naturaleza del sistema. Un sistema que acepta un pH de 9.2 y produce un resultado sobre ese dato está implícitamente afirmando que puede manejar inputs arbitrarios. Uno que rechaza ese dato con una lista explícita de las violaciones detectadas está afirmando lo contrario: que su dominio tiene límites definidos y que esos límites no son negociables.

La segunda decisión es la auditoría dual. Cada solicitud procesada por el sistema genera un registro en un archivo de log en formato JSONL —un estándar que permite

leer los registros línea a línea sin necesidad de cargar el archivo completo en memoria. Ese registro contiene el identificador de la sesión, el identificador del paciente, el módulo ejecutado, los inputs exactos y la versión del esquema. También contiene dos hashes SHA-256 calculados sobre el contenido de la solicitud.

Esos dos hashes tienen funciones distintas y complementarias. El primero —sha256_input— se calcula sobre los datos clínicos puros: paciente, módulo, inputs, versión. No incluye el momento de ejecución ni el identificador único de la solicitud. Eso lo hace determinista: el mismo payload de entrada producirá siempre el mismo hash, independientemente de cuándo se ejecute. Si dos solicitudes tienen el mismo hash de input, los datos clínicos son idénticos; si son distintos, hay alguna diferencia en los datos. El segundo —sha256_event— incluye el momento de ejecución y el identificador único. Por eso es irrepetible: dos ejecuciones del mismo payload siempre producirán hashes de evento distintos, aunque los hashes de input sean idénticos.

Esa arquitectura de trazabilidad permite responder preguntas concretas sobre cualquier output que el sistema haya producido: ¿Cuándo se calculó exactamente? ¿Con qué datos? ¿Ha sido modificado el registro? ¿Se puede reproducir el cálculo? Esas preguntas no son abstractas: son exactamente las que se formularían en una revisión clínica, en una auditoría de seguridad o en la validación de un sistema de apoyo médico. El hecho de que el sistema las anticipe y las haga respondibles es una decisión de diseño con implicaciones epistémicas reales.

La tercera decisión de diseño relevante es el vocabulario de acciones canónicas. El sistema define exactamente nueve verbos posibles para la acción que cualquier módulo puede emitir: «start_treatment», «discard_diagnosis», «obtain_test», «observe», «use_model», «do_not_use_model», «restrict_to_threshold_range», «review_dosing», «error» y «blocked». Cada verbo tiene una semántica precisa definida en la documentación del sistema. Ninguno de esos verbos es una instrucción clínica autónoma. Todos son clasificaciones computacionales que señalan al clínico qué tipo de resultado ha producido el cálculo y qué tipo de acción podría ser pertinente, pero sin ejecutarla ni decidirla.

La diferencia entre «start_treatment» y una orden de tratamiento no es trivial. «Start_treatment» significa: con la probabilidad pretest que introdujiste, con los resultados de prueba que introdujiste y con el umbral de tratamiento que definiste, la

probabilidad posterior ha cruzado ese umbral. Eso es un resultado computacional que el clínico puede considerar al tomar su decisión. No dice nada sobre si esa probabilidad pretest era apropiada, si los resultados de prueba se introducieron correctamente, si los parámetros del modelo corresponden a ese paciente, o si existe alguna contraindicación para el tratamiento. Todos esos elementos pertenecen al juicio clínico, que el sistema no pretende sustituir.

6. Objeciones, límites y matices

Un sistema que delimita cuidadosamente su propio dominio genera una objeción inmediata: ¿para qué sirve, entonces? Si Hipócrates no puede incorporar el contexto clínico completo, no individualiza a los pacientes más allá de los parámetros que se le introducen manualmente, no aprende de sus propios resultados y no puede usarse en decisiones clínicas autónomas, ¿qué ventaja ofrece sobre un médico que aplica mentalmente las mismas fórmulas?

La objeción es legítima, pero parte de un supuesto cuestionable: que la alternativa a usar Hipócrates es un médico que aplica las mismas fórmulas con igual rigor. En la práctica, eso rara vez es así. La actualización bayesiana secuencial con razones de verosimilitud, el cálculo del delta-delta y su corrección por albúmina, la estimación bayesiana MAP de parámetros farmacocinéticos individuales o la evaluación del beneficio neto de un modelo predictivo bajo distintos umbrales son operaciones que raramente se ejecutan a mano con la precisión que el sistema ofrece. El valor de Hipócrates no es reemplazar al médico que ya hace esos cálculos correctamente: es hacerlos accesibles, reproducibles y auditables para quienes, sin la herramienta, no los harían o los harían con menos precisión.

Sin embargo, existe una segunda objeción más difícil. Un sistema que produce outputs estructurados, con vocabulario de acciones aparentemente decisivo, puede generar una falsa sensación de rigor. Un médico que recibe el output «start_treatment» de un sistema llamado Hipócrates puede interpretar ese resultado como una validación más fuerte de lo que realmente es. La homogeneidad y la consistencia del output, que son virtudes desde el punto de vista del diseño, pueden funcionar como vectores de autoridad epistémica inmerecida cuando el usuario no comprende exactamente qué ha calculado el sistema.

Esa objeción no tiene solución arquitectónica completa. Por más advertencias que el sistema incluya en su documentación y en sus outputs, si el médico no entiende qué significa «start_treatment» en el contexto de un cálculo bayesiano con los parámetros que él mismo introdujo, ningún campo del output lo protegerá de una interpretación incorrecta. Hipócrates intenta mitigar ese riesgo mediante el campo «explain», que describe en lenguaje natural el razonamiento que produjo el resultado, y mediante la interfaz visual —construida en Streamlit— que muestra la traza paso a paso de la actualización bayesiana, la curva de beneficio neto o los parámetros calculados. Pero esa mitigación depende de que el médico lea esas explicaciones y las entienda. Si no lo hace, el riesgo persiste.

Una tercera objeción apunta a la simplicidad de los modelos. El módulo farmacocinético trabaja exclusivamente con modelos de un compartimento. Muchos fármacos clínicamente importantes —antibióticos con distribución tisular compleja, fármacos con alta unión proteica, compuestos con cinética bifásica— no se comportan bien bajo esa simplificación. Un modelo de un compartimento puede subestimar o sobreestimar significativamente las concentraciones plasmáticas en la fase de distribución, o en condiciones de equilibrio dinámico con tejidos periféricos. El sistema lo dice: «Modelo de 1 compartimento únicamente (ni 2C ni multi-C)». Pero esa declaración en la documentación no impide que alguien use el módulo para un fármaco al que ese modelo no aplica.

Del mismo modo, el módulo de sepsis calcula un SOFA parcial: cuatro de los seis componentes del score original. Los componentes ausentes —la puntuación cardiovascular formal y la neurológica por escala de Glasgow— no son secundarios: en muchos pacientes críticos, la disfunción del sistema nervioso central o el estado hemodinámico son los indicadores más determinantes de la severidad. Un SOFA parcial que omite esos componentes puede clasificar como «sepsis probable» a un paciente que cumple criterios de choque séptico completo, o viceversa. El sistema señala esa limitación explícitamente en la salida, pero la clasificación de severidad que produce sigue siendo parcial y potencialmente incompleta.

Frente a esas objeciones, el argumento que Hipócrates puede sostener tiene dos partes. La primera es que los sistemas que no declaran sus limitaciones son peores, no mejores. Un sistema que calcula un SOFA completo sin decir que su modelo

farmacocinético es de un compartimento, o que produce recomendaciones de tratamiento sin especificar qué parámetros usó, no resuelve los problemas que Hipócrates deja sin resolver: simplemente los oculta. La honestidad sobre las limitaciones, aunque no las elimine, permite al usuario calibrar su confianza en el sistema de manera más informada.

La segunda parte del argumento es que el estado actual del proyecto es explícitamente el de un prototipo de investigación, no un sistema de producción. Los módulos llevan designaciones de versión que indican madurez limitada: v1 para sepsis, v2 para farmacocinética. Los campos de utilidad esperada y de intervalos de confianza están reservados en el contrato de salida pero no implementados, con una nota en la documentación que explica que su ausencia no implica certeza sino que esa versión no cuantifica la incertidumbre. Los tests automatizados cubren el comportamiento correcto, no la validación clínica en poblaciones reales. Hipócrates ya supera el nivel de demostración improvisada —tiene arquitectura, contratos formales, trazabilidad y cobertura de tests— pero no pretende ser un sistema validado clínicamente. Esa distancia entre el prototipo y el producto es exactamente la que el sistema declara, sin minimizarla.

7. Conclusión

La pregunta que Hipócrates responde no es «¿puede un sistema computacional apoyar decisiones clínicas?» Esa pregunta lleva décadas respondida afirmativamente, con sistemas de distintas sofisticaciones y contextos. La pregunta que este sistema intenta responder es más específica y más difícil: ¿cómo se diseña ese apoyo de manera que el cómputo y el juicio clínico permanezcan claramente separados, que los errores sean detectables y auditables, y que el sistema sea honesto sobre lo que puede y no puede saber?

La respuesta de Hipócrates pasa por cuatro decisiones arquitectónicas que se refuerzan mutuamente. Primero, la validación secuencial antes de calcular: ningún módulo clínico se ejecuta sobre datos estructuralmente malformados o físicamente imposibles. Segundo, la salida homogénea: todos los módulos devuelven el mismo tipo de objeto, con el mismo vocabulario de acciones, sin variaciones de formato que puedan inducir a confusión. Tercero, la auditoría dual: cada ejecución queda

documentada con hashes que permiten verificar integridad y reproducibilidad. Cuarto, la prudencia semántica: las acciones canónicas son clasificaciones computacionales, no instrucciones autónomas, y el sistema lo explicita en la documentación de cada módulo y en el campo de explicación de cada output.

Lo que Hipócrates aporta no es un repertorio de cálculos que los médicos no pudieran hacer sin él. Aporta un marco en el que esos cálculos se hacen de manera reproducible, auditable y formalmente delimitada. La diferencia entre calcular mentalmente una actualización bayesiana y calcularla con un sistema que valida los inputs, documenta el proceso y distingue entre el resultado del cómputo y la decisión clínica no es sólo de precisión: es de trazabilidad y de honestidad epistémica.

Sería erróneo concluir de este análisis que Hipócrates está listo para su uso en entornos clínicos reales. Los módulos tienen limitaciones documentadas pero reales: modelos simplificados, cobertura parcial de scores clínicos establecidos, ausencia de cuantificación formal de la incertidumbre, dependencia de inputs manuales sin conexión a sistemas hospitalarios. Su validación es técnica —187 tests automatizados que verifican el comportamiento correcto de los algoritmos— pero no clínica: no hay estudios prospectivos que evalúen el impacto de su uso en outcomes de pacientes reales.

Lo que sí puede afirmarse es que la arquitectura del sistema expresa una concepción metodológicamente rigurosa de lo que debería ser el apoyo computacional a la decisión médica. En un campo donde abundan sistemas que mezclan cálculo, interpretación y prescripción sin marcar con claridad dónde termina cada cosa, Hipócrates constituye un argumento diferente: que la contribución más valiosa de un sistema computacional al razonamiento clínico puede ser, precisamente, saber cuándo detenerse.

---

8. Referencias

Las referencias de este ensayo se organizan según los materiales primarios del sistema analizado y las fuentes clínicas y estadísticas que sus módulos implementan explícitamente.

Fuentes primarias del sistema Hipócrates

[1] Hipócrates. Código fuente del proyecto. src/hipocrates/core/io_schema.py — Clinical_IO_Schema: validación del payload de entrada clínica. Versión analizada: SMNC-5+.

[2] Hipócrates. Código fuente del proyecto. src/hipocrates/core/units_gate.py — Units_Validity_Gate: gate de validación de dominio. Versión analizada: SMNC-5+.

[3] Hipócrates. Código fuente del proyecto. src/hipocrates/core/audit.py — Audit_Log_Provenance: auditoría criptográfica dual SHA-256 en JSONL. Versión analizada: SMNC-5+.

[4] Hipócrates. Código fuente del proyecto. src/hipocrates/core/orchestrator.py — Hipocrates_Orchestrator: pipeline schema → gate → módulo → auditoría → output. Versión analizada: SMNC-5+.

[5] Hipócrates. Código fuente del proyecto. src/hipocrates/utils/types.py — ClinicalInput, ClinicalOutput, Action: contrato de tipos del sistema. Versión analizada: SMNC-5+.

[6] Hipócrates. Código fuente del proyecto. src/hipocrates/modules/bayes_sprt.py — Bayes_SPRT_Engine: actualización bayesiana secuencial con parada temprana. Versión analizada: SMNC-5+.

[7] Hipócrates. Código fuente del proyecto. src/hipocrates/modules/abg_hh_stewart.py — ABG_HH_Stewart_Engine: interpretación gasométrica ácido-base. Versión analizada: SMNC-5+.

[8] Hipócrates. Código fuente del proyecto. src/hipocrates/modules/dca.py — DCA_Utility_Module: Decision Curve Analysis. Versión analizada: SMNC-5+.

[9] Hipócrates. Código fuente del proyecto. src/hipocrates/modules/pk_tdm.py — PK_TDM_Core v2.0: farmacocinética monocompartimental y monitoreo terapéutico. Versión analizada: v2.

[10] Hipócrates. Código fuente del proyecto. src/hipocrates/modules/sepsis_protocol.py — Sepsis_Protocol_Engine v1: estratificación de sepsis basada en criterios Sepsis-3. Versión analizada: v1.

[11] Hipócrates. README.md: documentación técnica del proyecto, glosario canónico, estructura de respuesta, criterios de uso y limitaciones.

Pintor, J. (2026). Hipócrates — Motor de Apoyo Clínico Computable (SMNC-5+) (Version 0.5.0) [Software]. Zenodo. https://doi.org/10.5281/zenodo.19475479

Código fuente: https://github.com/julespintor-tech/hipocrates

Proyecto OSF: https://osf.io/v46ue/

Criterios clínicos implementados

[12] Singer, M., Deutschman, C. S., Seymour, C. W., Shankar-Hari, M., Annane, D., Bauer, M., ... & Angus, D. C. (2016). The Third International Consensus Definitions for Sepsis and Septic Shock (Sepsis-3). JAMA, 315(8), 801–810. — Criterios de referencia para el módulo Sepsis_Protocol_Engine v1.

[13] Vincent, J.-L., de Mendonça, A., Cantraine, F., Moreno, R., Takala, J., Suter, P. M., ... & Thijs, L. (1996). Use of the SOFA score to assess the incidence of organ dysfunction/failure in intensive care units. Critical Care Medicine, 24(7), 1707–1710. — Tabla de puntuación SOFA utilizada en los componentes renal, hepático, de coagulación y respiratorio.

[14] Wald, A. (1945). Sequential tests of statistical hypotheses. The Annals of Mathematical Statistics, 16(2), 117–186. — Fundamento del procedimiento SPRT implementado en Bayes_SPRT_Engine.

[15] Vickers, A. J., & Elkin, E. B. (2006). Decision curve analysis: A novel method for evaluating prediction models. Medical Decision Making, 26(6), 565–574. — Marco conceptual del módulo DCA_Utility_Module.

[16] Cockcroft, D. W., & Gault, M. H. (1976). Prediction of creatinine clearance from serum creatinine. Nephron, 16(1), 31–41. — Fórmula estándar implementada en el modo cockcroft_gault del módulo PK_TDM_Core v2.

Hipócrates, por J.G de Lint (1867-1936)

---

9. Enlaces y recursos

El código fuente completo del sistema Hipócrates, incluyendo todos los módulos descritos en este ensayo, los tests automatizados, los ejemplos y la documentación, se encuentran disponibles en los siguientes repositorios:

GitHub: https://github.com/julespintor-tech/hipocrates

Zenodo: https://doi.org/10.5281/zenodo.19475479

OSF: https://osf.io/v46ue/

Nota metodológica

Este ensayo se elaboró a partir del análisis directo del código fuente del proyecto Hipócrates, en particular los archivos io_schema.py, units_gate.py, audit.py, orchestrator.py, types.py, bayes_sprt.py, abg_hh_stewart.py, dca.py, pk_tdm.py y sepsis_protocol.py, junto con el README.md del proyecto y los archivos de ejemplo en el directorio examples/. La bibliografía clínica citada corresponde únicamente a las referencias que el propio sistema declara explícitamente en su documentación y código como criterios de referencia para sus módulos. No se citan obras externas que no estén directamente referenciadas en los materiales del proyecto. No se inventó ni atribuyó ningún módulo, capacidad o limitación que no sea verificable en el código analizado.