

Working paper — borrador académico

No revisado por pares ni publicado en revista. Borrador v5 (2026). El contenido puede cambiar. © José Luis D'Andrade · IdentIA / Self-Improving Society. **Todos los derechos reservados.** Cita breve con atribución permitida. **Queda prohibida** la reproducción, traducción, adaptación o reutilización, total o parcial (incluido uso comercial), sin **autorización previa por escrito** del titular.

La escalera de selección: una restricción de orden para la auto-mejora recursiva acotada

Autor: José Luis D'Andrade **Afiliación:** LegalDesign.DO **Proyecto:** IdentIA — identia.do **Contacto:** www.linkedin.com/in/josedandrade **Borrador:** v5.0 (posición) — paper de posición. Parte [paper-v4-es.md](#) (retirado el 2026-08-27; en el historial de git). **Paper de sistemas compañero:** [paper-v5-systems-es.md](#) — *Instituciones persistentes de agentes como unidad de despliegue: invariantes de identidad, gobernanza-como-código y una compuerta de fiabilidad medida*. Citado abajo como [PA]. **Fecha:** Agosto 2026 · **Revisado:** 2026-09-28 (sigue la retirada de [PA] §7.3) · **Inglés:** [paper-v5-position.md](#)

Versión de solo texto principal, generada el 2026-09-28 a partir de [paper-v5-position-es.md](#): sin agradecimientos, declaración de IA ni referencias. Los [n] y la mención al Apéndice C remiten al fichero completo. No se versiona.

Resumen

Un sistema capaz de editar su propio andamiaje necesita una manera de decidir qué ediciones sobreviven. La respuesta dominante en la literatura actual es dejar que el sistema juzgue su propia salida —con un modelo, una rúbrica o un benchmark— y quedarse con lo que puntúa más alto. Sostenemos que ese es el default equivocado, y que el **orden** de lo que un sistema debe ser capaz de *medir* es en sí mismo la propiedad de seguridad.

Presentamos una **escalera de selección** de seis peldaños (F0–F5) en la que la auto-mejora recursiva es el *último* peldaño y es estructuralmente inalcanzable hasta que se cumplan tres propiedades previas: que el fitness esté **ajustado por dificultad** y sea medidamente **fiable** (F2), que actuar sobre él sea **estable y reversible** (F3), y que los **nichos raros pero valiosos estén protegidos** de ser desplazados por un ganador temprano (F4). Cada peldaño avanza por un **self-trigger** —una condición que el sistema mide sobre sus propios datos, no una fecha en una hoja de ruta—. La dependencia se escribe como una conjunción que el sistema evalúa sobre sí mismo, y $\text{Done}(F2)$ significa la medición en vivo, no la existencia del código.

Este es un paper de posición y la posición no ha sido probada. F0–F2 están implementados y uno de ellos ha corrido en vivo; esa mitad se reporta en el paper de sistemas compañero [PA], que además **retira** la medición en vivo que versiones anteriores de los dos papers ofrecían como la compuerta que bloqueaba todo lo que está por encima ([PA] §7.3): la bandera de convergencia en que se apoyaba no podía valer `True` bajo el criterio

entonces en uso. F3 y F4 computan en sombra y nunca han actuado, **por decisión y no por una compuerta medida**. F5 nunca ha corrido. Las propiedades que este paper llama precondiciones son por tanto **requisitos que imponemos, no garantías que hayamos verificado**, y §6 enuncia qué falsaría la posición en vez de dejarla infalsable.

Enunciamos además, por separado y como afirmación normativa y no como resultado, la tesis de **metacentralidad** que moldeó el esquema de gobernanza alrededor de la escalera: que toda arquitectura de poder tiende a capturarse o a disolverse, y que la respuesta es un centro deliberadamente acotado por auditoría, rotación, transparencia y autolimitación. Le damos sección propia, límites propios, y ninguna credibilidad prestada de las mediciones de [PA].

Palabras clave: auto-mejora recursiva; calidad-diversidad; teoría de respuesta al ítem; presión de selección; gobernanza de IA; sociedades de agentes.

1. Introducción

El runtime de agentes ya provee un **operador de variación**: los agentes modernos escriben y parchean sus propias skills y consolidan su propia memoria. Lo que el campo no ha resuelto es el **operador de selección** —el procedimiento que decide cuáles de esos cambios auto-escritos son realmente mejores y deben sobrevivir—.

El patrón predominante es dejar que el objeto cambiado se certifique a sí mismo: un modelo puntúa su propia salida, un número de benchmark sube, la edición se conserva. Eso funciona hasta que la señal es gameable, y toda señal construida desde la misma fuente que aquello que se optimiza es gameable. El modo de fallo no es dramático: es deriva que se lee como mejora.

Nuestra posición es que el arreglo es el **orden**, y que el orden no es una hoja de ruta sino una restricción. Antes de que a un sistema se le permita editarse a sí mismo, tiene que poder demostrar —sobre sus propios datos, por su propia medición— que la señal que optimizaría está ajustada por dificultad, es fiable, es estable para actuar sobre ella, y no está colapsando su propia diversidad. Cada una de esas cosas es un peldaño. Ninguna es novedosa como estimador; cada una se toma sin cambios de una literatura establecida. Lo que se afirma es sobre qué están cableadas a **condicionar**.

Lo enunciamos como posición porque lo es. Los peldaños bajos de la escalera están contruidos y uno ha corrido; los peldaños que cargan el argumento, no. La sección 6 no es por tanto una formalidad: es donde decimos qué resultado nos diría que esto está mal.

2. Trabajo relacionado

2.1 Agentes auto-mejorantes y auto-diseñadores

Voyager [4] construyó una **biblioteca de skills** ejecutable y en crecimiento continuo para un único agente de Minecraft mediante un currículo automático y auto-verificación. **ADAS / Meta Agent Search** [5] hace que un meta-agente *programme nuevos agentes en código* sobre un archivo en crecimiento. La **Darwin Gödel Machine**

[6] relaja el requisito de prueba de la Gödel Machine original [26] a validación *empírica*, reescribiendo iterativamente su propio código y manteniendo un archivo de variantes descubiertas. **AlphaEvolve** [7] (el sucesor de **FunSearch** [8]) evoluciona programas mediante cambios directos a su código, contra evaluadores automatizados. **PromptBreeder** [16], **EvoPrompt** [35] y PBT [15] evolucionan prompts y configuraciones.

SIS adopta estos como el operador de *variación* y la *búsqueda evolutiva* de su peldaño superior (F5). Su contribución distintiva es la **disciplina de ordenamiento**: una *escalera* en la que la fitness empírica se hace confiable *antes* de que se le permita actuar, mediante ajuste de dificultad (Teoría de Respuesta al Ítem, F2) y preservación de diversidad (MAP-Elites, F4), con un conjunto held-out anti-Goodhart que *se computa pero todavía no condiciona* (§7). La Darwin Gödel Machine demuestra fitness empírica + archivo pero no impone un escalonamiento ajustado por dificultad, autoimpulsado y con compuerta de gobernanza de *cuándo* se permite la edición recursiva. SIS hace de ese escalonamiento el argumento central de seguridad (§4), y — a diferencia de un evolucionador puro guiado por benchmark — mide la *fiabilidad de su propia señal de fitness* (los self-triggers de convergencia IRT y estabilidad de θ) antes de dejar que la selección actúe.

Lo más cercano al planteamiento de SIS es **Synergy** [31], que define al *Agentic Citizen* de una "Open Agentic Web" mediante tres requisitos — colaboración nativa de la web, identidad y personhood del agente, y evolución de por vida — e implementa el tercero como *experience learning*: las trayectorias de interacción se codifican con recompensas multidimensionales derivadas del diálogo y se re-inyectan en tiempo de inferencia por recuperación consciente-del-valor, con grandes ganancias intra-benchmark (+31–36% relativo en SWE-bench Verified; +148% en OpenRCA) y transferencia inmediata a instancias frescas. La convergencia de preguntas con IdentIA es notable — la memoria tipada, el perfil y la agenda de Synergy son paralelos a nuestra descomposición SOUL/memoria/linaje — pero la disciplina de selección difiere fundamentalmente: su recompensa derivada del diálogo alimenta la recuperación directamente, sin ajuste de dificultad, sin ancla held-out anti-Goodhart y sin compuerta de fiabilidad entre medición y acción; la gobernanza se nombra como problema abierto en lugar de proveerse como mecanismo. SIS ocupa la posición complementaria: la *institución* como la unidad que hace gobernables a los ciudadanos agénticos — la selección actúa solo cuando es medidamente fiable, y la transmisión de capacidad está supeditada a la validación.

2.2 Selección, rating y calidad-diversidad

Los estimadores que SIS asciende son clásicos. **Bradley–Terry** [12] modela el "mejor-que" pareado; **TrueSkill** [13] da habilidad bayesiana con incertidumbre y empates; la **Teoría de Respuesta al Ítem** [20] separa la habilidad latente de la dificultad del ítem; el **muestreo de Thompson** [22] convierte posteriors en una política de enrutamiento; **NSGA-II** [23] maneja compensaciones multiobjetivo; **MAP-Elites** [14] y **POET** [21] preservan y hacen crecer la diversidad conductual; **PBT** [15] evoluciona configuraciones en línea. **Chatbot Arena** [24] popularizó Bradley–Terry para la evaluación de LLM a escala. La contribución de SIS no son estos estimadores sino la *asignación* de cada uno a un objeto de fitness distinto (skill, perfil, hecho, repertorio, enrutamiento) y la *progresión autoimpulsada* entre ellos (§3), más la decisión de ingeniería de implementarlos con **ninguna dependencia de rating de terceros** (solo la biblioteca estándar de Python), lo que mantiene la capa de selección auditable y libre de superficie de cadena de suministro.

3. La escalera

Los principios de IdentIA A (Impulso Dinámico), B (Continuidad Absoluta) y C (Pragmatismo Emergente), el puente A → C, y los niveles de gobernanza L1–L8 que figuran en los peldaños se definen en [PA] §3 y §6.1. Este paper usa las etiquetas sin repetirlos.

3.1 Variación vs. selección

El runtime del agente ya provee un **operador de variación**: escribe y parchea sus propias skills y consolida memoria en una revisión de auto-mejora. Lo que falta es un **operador de selección medida**: una forma de decidir *cuáles* de esos cambios son realmente mejores y deberían sobrevivir, propagarse (el puente A → C), y anclarse (principio B). Esquemáticamente:

$$\text{RSI} = \underbrace{\text{Variation}}_{\text{exists (runtime)}} \circ \underbrace{\text{Selection}}_{\text{this ladder}} \text{ in a closed loop, editing the scaffold, gated by fitness.}$$

La variación sin selección es **deriva**; la selección sin variación no tiene nada que elegir. El encuadre es deliberadamente darwiniano [28]: la mejora nunca la *declara* el objeto que cambió; la *selecciona* la comparación medida contra alternativas en un entorno. La escalera construye la selección primero, y solo su peldaño superior cierra el lazo recursivo.

3.2 Cinco objetos de fitness

No hay un único "número de fitness". Cinco objetos portan señal de forma diferente:

Objeto	Señal cruda	Tabla fuente	Estimador objetivo
Skill	usage_count , success_count	skill_publications	Posterior Beta → Bradley–Terry/TrueSkill
Perfil/agente	éxito/fracaso de tarea, heartbeats	Kanban task_runs , runtime_heartbeats	IRT (ajustado por dificultad) + reputación de grafo
Hecho	sustitución, citas	facts / society_facts	tasa de supervivencia/sustitución + predicción-por-pares
Repertorio	diversidad de skills/perfiles activos	fitness_repertoire	Calidad-Diversidad (MAP-Elites)
Enrutamiento	quién recibe la próxima tarea	fleet_manager , despacho Kanban	Muestreo de Thompson sobre posteriors

3.3 Los peldaños y sus self-triggers

Cada peldaño es un hito que declara un algoritmo, los datos que necesita, un **self-trigger** (una condición que el sistema mide sobre *sí mismo* para avanzar), su principio dominante de IdentIA, y su nivel de gobernanza. Los umbrales de self-trigger son configurables por entorno (SOCIETY_FITNESS_*). Los badges reflejan el código a junio de 2026:  implementado y probado;  implementado, en curso o shadow;  solo diseño.



3.4 Los peldaños medidos, en un párrafo

F0 pone una posterior Beta–Bernoulli sobre la tasa de éxito de cada objeto y ordena por la **cota inferior de Wilson**, de modo que un objeto con un éxito en un intento no supere a uno con noventa en cien —la incertidumbre se carga, no se descarta—. F1 convierte los resultados en habilidad **relativa** con Bradley–Terry [12] y TrueSkill [13], alimentados por un juez pareado que evalúa cada comparación en los dos órdenes para suprimir el sesgo de posición. F2 añade **Teoría de Respuesta al Ítem** [20]: la habilidad θ se estima conjuntamente con la dificultad del ítem, así que un profile no puede subir de rango eligiendo trabajo fácil. Los tres están implementados y con pruebas unitarias, y F2 ha corrido en vivo; la matemática, los caminos de código, los tests y el resultado fechado están en **[PA] §5.4–§5.6 y §7.2**. Este paper no los repite, porque el argumento de abajo no depende de sus internos —solo de la propiedad que F2 está construido para certificar, y del hecho de que en la única ocasión en que corrió, reportó esa propiedad como **ausente**—.

3.5 F3–F5 — actuar, preservar diversidad, y RSI con compuerta

F3 (lazo cerrado — implementado en shadow). El muestreo de Thompson [22] sobre los posteriors Beta/TrueSkill/IRT selecciona al próximo asignado (explorar/explotar sin ajuste manual); **NSGA-II** [23] mantiene el enrutamiento en el frente de Pareto de {calidad, costo, latencia} mediante un orden rápido no-dominado en lugar de maximizar solo la calidad; una métrica de cobertura de Calidad-Diversidad vigila el monocultivo. El motor (evaluator/routing.py: thompson_select, non_dominated_sort, pareto_front, qd_coverage) tiene pruebas unitarias (tests/test_routing_f3.py, 12/12), y un **consumidor está integrado en fleet-manager** (routing_client, kanban_task_create) pero corre en **shadow / deshabilitado por defecto** (SOCIETY_FITNESS_ROUTING_ENABLED=false): *recomienda* pero no asigna. La activación requiere un F2 en vivo y convergido más gobernanza L3b/L5 y una fuente real de señal de costo; el modo **active** es **fail-open** —degrada a shadow si no se cumple la estabilidad de θ o la convergencia (trigger_f3_ready). La latencia ya se captura de task_runs; una fuente real de costo de gasto LLM es el insumo faltante. Diseño y estado: docs/F3-F4-FLEET-WIRING.md.

F4 (diversidad — implementado en shadow). **MAP-Elites** [14] sobre un descriptor de comportamiento (dominio dominante $\times$ banda de dificultad media; nivel de costo diferido) mantiene la élite de cada celda (un trinquete por celda), impidiendo el *monocultivo a lo largo de los descriptores definidos*: los nichos raros-pero-valiosos están protegidos de ser superados por el ganador temprano. El archivo (migración 014_fitness_repertoire.sql, _recompute_repertoire, protected_elites, population_plateau) tiene pruebas unitarias (tests/test_fitness_f4.py, 10/10, incl. cobertura monótona bajo presión de selección). Un consumidor de protección-de-nicho está integrado en las vías de retirar/destruir perfil (una élite requiere ? force=true; el archivado *no* tiene compuerta, según IdentIA-B) pero está deshabilitado por defecto y es fail-open, a la espera del cableado de F3-activo.

F5 (RSI empírica con compuerta — solo diseño, frontera). Solo aquí el sistema edita su propio andamiaje (SOUL, skills) bajo la fitness fiable construida abajo. **PBT** [15] explota-y-explora sobre configuraciones —cuyo paso de "explotar" es el trinquete de no-regresión (principio B)— con **PromptBreeder** [16] o **EvoPrompt** [35] como operador de variación sobre SOUL/skills, y generación estilo **POET** [21] de tareas más difíciles cuando F4 detecta una meseta. La variación está **siempre** con compuerta de la fitness de F2. La identidad RSI, además,

debe ser un **perfil persistente dedicado** (un `operator / self_improver` en un "host Dev/Operator" separado con el repositorio montado detrás de compuertas de `worktree/branch/test/build/smoke/governance`), no el arquitecto conversacional ni la semilla por defecto del runtime — porque por el principio B una identidad auto-mejorante necesita su propio SOUL, memoria, sesiones, linaje y fitness observable.

4. Por qué el orden es la afirmación

Los tres peldaños de §3.5 no son una hoja de ruta de funcionalidades en el orden en que nos tocó construirlas. Son una dependencia, y esa dependencia es la posición que este paper defiende. Dicho con cuidado: **la afirmación aquí es sobre condiciones necesarias, no sobre seguridad demostrada.**

F5 **no puede** ser seguro antes, porque solo F2 hace la fitness ajustada por dificultad (**diseñada para resistir** el gaming por selección de tareas fáciles) y medible-fiable (convergencia + estabilidad de θ); solo F3 hace el actuar estable y reversible; y solo F4 **pretende** proteger los nichos raros de ser superados. Ninguna de esas propiedades está establecida empíricamente —F3 y F4 nunca han actuado en vivo y F5 nunca ha corrido—, así que son prerequisites que imponemos, no garantías que hayamos verificado. Encender la auto-edición recursiva sin F2–F4 sería "RSI sin selección fiable = deriva peligrosa". La dependencia se aplica:

$$\text{enable}(F5) \iff \text{Done}(F2) \wedge \text{Done}(F3) \wedge \text{Done}(F4).$$

Crucialmente, $\text{Done}(F2)$ **no** es "el código IRT existe"; es el self-trigger en vivo — θ estable y `converged=True` . Versiones anteriores de este paper añadían que la corrida en vivo de junio de 2026 reportó `converged=False` , y concluían que la activación de F3, y por ende F5, estaba bloqueada por la propia medición del sistema y no meramente por política. **[PA] §7.3 lo retira:** la bandera no podía valer `True` bajo el criterio entonces en uso, y la mitad de estabilidad de θ del disparador se satisfacía a sí misma con datos sin cambiar. F3 y F5 están apagados, y hoy lo están **por decisión**. Queda retirada la afirmación de que se haya *observado* a la conjunción hacer el bloqueo; la afirmación de que la conjunción es lo correcto que exigir es el argumento de este paper, y no queda afectada.

La compuerta y esta consecuencia están comprobadas por máquina: [PA], Apéndice E, enuncia la regla sobre banderas *medidas* en Lean 4 y prueba que sin convergencia F5 no se puede habilitar (`enableF5_iff` , `f5_blocked_of_not_converged`). La prueba es sobre las *banderas*, así que la retirada no la toca — lo que se retira es la afirmación de que la corrida del 2026-06-18 fuera un estado con la bandera sin poner. Lo comprobado es la regla, no la afirmación de que la regla haga seguro al sistema.

Esta postura tiene un antecedente clásico. El segundo teorema de incompletitud de Gödel [27] muestra que un sistema formal suficientemente expresivo no puede establecer su propia consistencia desde dentro. No afirmamos que el teorema aplique literalmente a SIS; adoptamos su moraleja de ingeniería: las garantías que autorizan la auto-modificación — convergencia, estabilidad de θ , el ancla held-out rotatoria y, en última instancia, las compuertas humanas de [PA] §6 — están colocadas deliberadamente *fuera* del lazo que certifican. Un sistema auto-mejorante que califica su propia mejora con la misma señal que está optimizando no tiene suelo donde pararse; los anclajes externos de la escalera son ese suelo.

5. Metacentralidad: una posición normativa sobre dónde debe estar el poder

El esquema de gobernanza expresa una tesis de diseño: toda arquitectura de poder tiende a **capturarse** o **disolverse**. La respuesta no es elegir un polo sino diseñar **metacentralidad consciente** — un centro con auditoría explícita, rotación, transparencia y autolimitación. Operacionalmente, cinco movimientos: reconocer la tendencia a capturar; incorporar verificación interna continua; permitir la redistribución dinámica del poder; mantener la cohesión sin absolutizar el centro; integrar la memoria histórica para que los ciclos no se repitan inconscientemente. Estos se mapean sobre la pila de gobernanza ([PA] §6.1), y **el mapeo es desigual: dos de los cuatro movimientos corren hoy y dos no**. La verificación interna continua es el evaluador y la capa de fitness **F0–F2** (L4–L6), que están vivos; F3–F4 computan en sombra. La redistribución dinámica *sería* el enrutamiento impulsado por fitness bajo presupuestos (L3b/L5) — pero ese enrutamiento es F3, desactivado por defecto, y la compuerta L5 se aplica opt-in ([PA] §6.1), así que este movimiento está **enunciado, no operando**. La cohesión-sin-absolutismo **es** el invariante de ninguna-verdad-unilateral más la cuarentena (L4), que sí se aplican. La memoria histórica **es** archivar-no-borrar y la sustitución de hechos (B), que sí se aplican. A medida que el poder cognitivo supera al poder político humano, la defensa es la **metagobernanza** (L7–L8) — límites verificables y auditoría permanente que el centro puede aplicarse a *sí mismo* antes de volverse absoluto — en lugar de más coerción. Esta es una posición normativa, ofrecida para escrutinio, no un resultado empírico.

Y no añade ninguna restricción propia. Todo mecanismo nombrado arriba lo aplica —o no— alguno de los niveles de [PA] §6.1 o de los peldaños de §3. La metacentralidad los *interpreta*; no les añade nada, y nada de esta sección dejaría de funcionar si la sección se borrara. Lo que no especifica es cómo el centro se audita a *sí mismo* más allá de aprobación humana, PR y tests: no hay mecanismo de rotación, ni límite de mandato, ni procedimiento por el cual L7–L8 pudieran volverse contra quien los tiene. Ese es el punto más delgado de la tesis. Lo nombramos aquí porque el vocabulario de esta sección es más elevado que el de la tabla de gobernanza que interpreta, y un vocabulario elevado es justamente lo que hace que una idea no aplicada se lea como aplicada.

6. Qué falsaría esta posición

Una posición que no puede estar equivocada no es una posición. Cuatro resultados contarían en contra de esta, en orden creciente de lo que nos costarían:

1. **Auto-edición bajo una señal poco fiable que no degrada.** Si se mostrara que un sistema edita su propio andamiaje mientras su señal de fitness es medidamente poco fiable — θ no convergida, inestable entre re-estimaciones— y su desempeño en tareas posteriores no degrada a lo largo de una corrida sostenida, la afirmación central se debilita de inmediato. El orden sería entonces un coste sin el riesgo correspondiente.
2. **Un monocultivo que no ocurre.** El estatus de F4 como precondition descansa en la premisa de que la selección sin protección colapsa sobre un ganador temprano y pierde nichos raros pero valiosos. Si una sociedad de larga duración bajo enrutamiento tipo F3 *sin* MAP-Elites preservara igualmente su repertorio, F4 sería una optimización y no una compuerta, y a la escalera le sobraría un peldaño.

3. **Una compuerta que nunca abre.** Si `converged=True` resultara inalcanzable a volúmenes de datos institucionales —no porque el diseño esté mal sino porque el tráfico real nunca produce la densidad uno-a-uno que el IRT necesita— la escalera sería correcta e inútil. Es el fallo que hoy consideramos más probable, y es la razón de que exista el gym ([PA] §7.1). Una advertencia sobre cómo lo leeríamos: hasta el 2026-09-26 nuestro propio criterio no podía reportar convergencia para ningún dato, así que una bandera inalcanzable es una afirmación sobre los datos solo después de mostrar que el instrumento se mueve ([PA] §7.3). No refutaría el orden; refutaría la escalera como instrumento práctico.
4. **Que el primer peldaño en vivo no compre nada.** El experimento decisivo más barato es el primer punto del trabajo futuro de [PA] §10: alcanzar `converged=True` en una instalación de cliente y no en el entorno de desarrollo donde se reparó el criterio de convergencia, activar F3, y comparar contra enrutamiento round-robin sobre la misma suite. Si el enrutamiento guiado por fitness bajo una señal convergida no muestra ventaja medible sobre round-robin, la propuesta de valor de la escalera falla en el primer peldaño donde pudo haberse demostrado.

Anotamos cuáles podemos correr y cuáles no. (4) está al alcance y es el siguiente trabajo. (1) y (2) requieren un estudio longitudinal de una sociedad a la que se le haya permitido actuar, cosa que por construcción no puede empezar hasta que (4) tenga éxito. (3) solo será responsable desde tráfico de producción a lo largo de meses.

7. Estado: nada de este paper ha actuado

Los peldaños sobre los que este paper argumenta no han corrido. Dicho exacto, y trazable al ledger de claims en [PA] Apéndice C:

- **F3** (muestreo de Thompson + enrutamiento NSGA-II) está implementado y con pruebas unitarias pero corre **en sombra, desactivado por defecto**. Recomienda; no asigna. Ningún camino lo consulta como precondition. Ledger E-14.
- **F4** (archivo MAP-Elites, protección de nichos, detección de meseta) está implementado y con pruebas unitarias e igualmente **desactivado por defecto y fail-open**, a la espera del cableado de F3-activo. Ledger E-14.
- **F5** es **solo diseño y con compuerta dura**. Nunca ha corrido.
- La razón de que ninguno esté activo es una **decisión, no una medición**. Versiones anteriores decían lo contrario, citando la corrida en vivo de F2 del 2026-06-18 como un self-trigger que retuvo la activación; esa lectura queda retirada ([PA] §7.3, ledger E-27), porque la bandera de convergencia no podía valer `True` bajo el criterio entonces en uso. Tras el arreglo, un ajuste en un entorno de desarrollo reporta `converged=True` (ledger E-28), y F3 sigue apagado porque no lo hemos encendido. Desde `c811267` la bandera se guarda y `fitness_route` en modo *activo* sí vuelve a shadow cuando el último ajuste no convergió, fallando cerrado (ledger E-29) — construido y probado, nunca ejercitado en vivo.
- El ancla anti-Goodhart de reserva, que el diseño de la escalera trata como salvaguarda, **se computa y se sirve pero no bloquea nada** —un indicador, no una compuerta ([PA] §5.8, ledger E-15)—.

De los 21 componentes que registra [PA] §7.4, doce no restringen nada hoy. El bucle cerrado variación → selección → mutación necesita elementos de dos de esos grupos —F3 actuando en vez de en sombra, y F5 existiendo siquiera—, así que no puede correr hoy por construcción, no por configuración.

8. Conclusión

Hemos argumentado que el orden de propiedades medibles es la restricción de seguridad en una sociedad de agentes auto-mejorante, y que la auto-edición recursiva pertenece al final de ese orden y no al principio de una hoja de ruta. La escalera hace la dependencia explícita y evaluable por el propio sistema. Antes añadíamos que en la única ocasión en que la compuerta corrió se negó, y llamábamos a eso la única parte del argumento observada en vez de afirmada; **[PA] §7.3 lo retira**, así que hoy ninguna parte de este argumento se ha observado. La versión honesta es más corta: el orden está escrito, comprobado por máquina como regla, y sin probar en operación.

El resto es una posición. F3 y F4 nunca han actuado, F5 nunca ha corrido, y las propiedades que llamamos precondiciones son requisitos que imponemos. Hemos mantenido este argumento fuera del paper de sistemas por esa razón, hemos dicho en §6 qué nos diría que está equivocado, y hemos nombrado en §5 la parte más delgada de la tesis de gobernanza que lo acompaña —que todavía no especifica cómo se audita el centro a sí mismo—. El resultado útil siguiente no es una afirmación más grande; es la matriz del gym rediseñada hasta que θ sea identificable, y luego la primera comparación honesta entre enrutamiento guiado por fitness y round-robin.