BONDY

THINKING · PAPER V1.0 · ABRIL 2026

Cómo piensa, no qué sabe

Contratar mal hoy es más fácil que nunca, y cuesta más caro que nunca.

El método de evaluación cognitiva de talento técnico de Bondy para la era de la IA. Publicamos el paper completo, con sus límites y su bibliografía. No es un folleto.

Mara Schmitman · Founder, Bondy Group

Buenos Aires · Abril 2026

Prólogo

Empecé a hacer recruiting técnico en 2008, cuando el mercado argentino recién descubría que los desarrolladores eran un recurso escaso y que había que empezar a tratarlos distinto. Mi formación no era la de una recruiter: era psicóloga. Entré al mundo del hiring tech desde un lado que parecía tangencial y terminó siendo central: no me importaba tanto qué sabían hacer los candidatos como cómo pensaban cuando se enfrentaban a un problema.

Durante los primeros años, esa mirada me parecía una excentricidad mía. Los clientes querían saber cuántos años de experiencia tenía alguien con Java, cuántos proyectos había liderado, qué tamaño de equipo había manejado. Yo les insistía con preguntas que les parecían raras: cómo habían tomado tal decisión, por qué habían descartado tal alternativa, qué habían aprendido cuando algo salió mal. La respuesta más frecuente era una especie de paciencia condescendiente: "sí, Mara, pero igual necesitamos el senior con 5 años de Java".

Eso funcionó durante mucho tiempo. No porque estuviera bien, sino porque funcionaba lo suficientemente mal como para que nadie lo cuestionara. El hiring tech mundial se construyó sobre un modelo implícito: evaluar lo que la persona sabía hacer, porque hacer era lo escaso. Los lenguajes de programación, los frameworks, las certificaciones, los años de experiencia: todas esas cosas eran indicadores razonables para una pregunta más difícil, ¿esta persona va a resolver problemas reales en nuestra empresa?

Ese modelo ya no funciona. Y no funciona por una razón específica que todo el mundo intuye pero pocos nombran con claridad: la inteligencia artificial volvió barato, rápido y accesible lo que antes era el indicador principal de la competencia técnica. Escribir código ya no es el cuello de botella. Leer documentación, generar boilerplate, resolver problemas de sintaxis, incluso diseñar funciones completas: todo eso lo hace una herramienta que cualquier candidato puede usar. Lo que antes distinguía a un buen ingeniero de uno mediocre (la velocidad, la fluidez técnica, la memoria del lenguaje) se volvió una commodity.

Lo que queda como diferencial es exactamente lo que yo venía mirando hace 18 años sin poder nombrarlo del todo: cómo piensa la persona cuando tiene que decidir qué construir y qué no, qué descartar y por qué, qué confiar a la máquina y qué sostener con criterio propio. Lo que antes era una excentricidad de psicóloga hoy es la única pregunta que importa.

Escribo este documento porque ese cambio me obliga a hacer explícito lo que durante años operó como intuición. Bondy creció sobre la base de una forma de evaluar candidatos que yo podía transmitir al equipo por ósmosis: entrevistas juntas, discusiones de casos, años de calibración informal. Eso ya no alcanza. Primero porque el equipo creció y ya no puedo entrenar a cada recruiter por contacto directo. Segundo porque los clientes están empezando a hacer preguntas

que el mercado no sabe responder con autoridad: ¿cómo evalúan ustedes a un desarrollador en la era de ChatGPT? ¿Qué miran que otros no miran? ¿Por qué deberíamos confiar en su criterio cuando todo el mundo dice que tiene un método?

Tercero, y más importante: porque el problema se volvió urgente. Los clientes están contratando mal a una velocidad que antes no era posible. Un mal hire hoy se detecta más tarde, porque el output inicial del candidato se ve bien: la IA lo ayuda a producir. Cuando aparecen los problemas, seis meses después, el daño ya está hecho. deuda técnica, decisiones arquitectónicas malas, equipos desalineados, procesos que nadie puede sostener. Y el mercado sigue evaluando como en 2018.

Este documento es el intento de articular, con el mayor rigor posible, el método con el que Bondy viene operando y que quiero dejar escrito para tres públicos distintos. El primero es el equipo de Bondy, porque necesitan una herramienta común que los saque de la dependencia del criterio personal. El segundo son los clientes, porque merecen entender qué hacemos cuando decimos que evaluamos distinto. El tercero es la comunidad de recruiting y HR tech, porque creo que lo que aprendimos tiene valor más allá de Bondy y prefiero hacerlo público que guardarlo como ventaja competitiva.

Un aviso honesto antes de empezar: este método no es una receta. Es un marco de evaluación cognitiva construido sobre 18 años de práctica, influenciado por el trabajo de Will Larson sobre arquetipos de rol en ingeniería, anclado en la tradición de la psicología organizacional sobre modos de pensamiento, y probado en cientos de procesos de selección en empresas tech de LATAM y el mundo. Tiene límites. Hay cosas que no resuelve. Hay situaciones donde otro método funciona mejor. Intento ser explícita sobre todo eso a lo largo del documento, porque creo que los métodos serios son los que admiten sus bordes.

Si estás leyendo esto como cliente potencial o actual de Bondy: lo que sigue es la base de cómo evaluamos a los candidatos que te presentamos. Si estás leyendo esto como colega de la industria: espero que te sirva y me interesa tu crítica. Si estás leyendo esto como miembro del equipo de Bondy: esta es la fuente. Úsenla, cuestionenla, ajústenla con los casos reales que van acumulando.

El mercado del hiring tech se está quebrando en tiempo real. No es momento de métodos tibios.

Mara Schmitman

Buenos Aires, abril 2026

1. El problema

Hay dos formas de entender lo que está pasando en el hiring tech hoy. La primera es leer los números. La segunda es escuchar a los CTOs.

Los números son contundentes. En 2025, según el Stack Overflow Developer Survey, el 84% de los desarrolladores profesionales reportó usar o planear usar herramientas de inteligencia artificial en su trabajo diario. Casi la mitad del código nuevo que se escribe en empresas con alta adopción tecnológica es generado por asistentes de IA. GitHub reportó 43 millones de pull requests mensuales en 2025, un 23% más que el año anterior. La velocidad de producción de código se aceleró de forma drástica en menos de tres años.

Pero hay otro número, más incómodo, que rara vez se cita en la misma conversación: la productividad agregada de los equipos de ingeniería no creció en la misma proporción. El reporte DORA de 2025, publicado por Google Cloud, lo dice con una frase que merece volverse famosa: la inteligencia artificial actúa como espejo y multiplicador. En organizaciones cohesivas, con buenas prácticas y procesos claros, amplifica las fortalezas. En organizaciones fragmentadas, amplifica las debilidades. La IA no arregla equipos rotos. Los acelera hacia el lugar al que ya iban.

Eso explica por qué los CTOs están teniendo conversaciones que hace tres años no tenían. Cuentan que sus equipos producen más código que nunca y entregan menos valor que nunca. Cuentan que las decisiones técnicas tempranas se vuelven imposibles de revertir más rápido. Cuentan que los juniors que contratan parecen competentes en la entrevista y se desarman a los seis meses cuando tienen que sostener algo en producción. Cuentan que los seniors que contratan son indistinguibles entre sí en el papel y radicalmente distintos en el impacto real.

La paradoja del proxy roto (o del indicador roto)

Durante décadas, el hiring técnico operó sobre un consenso implícito: si una persona puede demostrar que sabe hacer algo, probablemente sepa pensarlo. Saber escribir código en Python era un indicador razonable de que la persona entendía programación. Saber resolver un algoritmo en una pizarra era una señal razonable de capacidad para resolver problemas. Tener cinco años de experiencia en una empresa grande era un atajo razonable para asumir que esa persona había visto suficientes situaciones complejas como para tener criterio.

Estos indicadores nunca fueron perfectos, pero funcionaban lo suficiente. Y lo que les daba sentido era una premisa que casi nadie cuestionaba: hacer es lo escaso. Si el mercado tenía pocos developers, y producir código tomaba mucho tiempo, entonces la velocidad de producción era una buena medida de la calidad del recurso humano.

La inteligencia artificial rompió esa premisa. Hoy, producir código es barato. Lo que es escaso es decidir qué código producir, qué descartar, cuándo confiar en el output de la máquina y cuándo dudarlo, cómo integrar lo que la IA genera en sistemas que tienen que sostenerse en el tiempo. Los indicadores que construimos sobre la escasez de la producción dejaron de medir lo que importa.

Esto crea una situación que podríamos llamar la paradoja del indicador roto: las herramientas de evaluación más extendidas en la industria (entrevistas técnicas tradicionales, tests de algoritmos, revisión de portfolios, métricas de productividad individual) siguen midiendo cosas que ya no diferencian a los buenos de los mediocres. Y mientras tanto, los criterios que sí diferencian (capacidad de decisión, criterio bajo ambigüedad, capacidad de leer y mejorar código ajeno, traducción entre técnico y negocio) son evaluados de manera implícita, intuitiva, no entrenable, y dependen del olfato del entrevistador.

La consecuencia económica

Lo que vuelve esto urgente no es solo el cambio técnico. Es la consecuencia económica. Una mala contratación tech siempre fue cara, pero hoy es más cara por tres razones convergentes.

La primera es que los malos hires se detectan más tarde. Antes, una persona que no podía resolver problemas se hacía evidente en las primeras semanas: no podía escribir código, no podía mantener el ritmo del equipo, no podía cumplir con sus tareas. Hoy, la IA actúa como prótesis: alguien que no puede pensar puede igual producir output decente durante meses, hasta que aparece una situación que requiere criterio real y todo se desarma.

La segunda es que las decisiones técnicas tempranas son más difíciles de revertir. La velocidad con la que se construye software hoy hace que los equipos tomen decisiones arquitectónicas significativas en semanas, no en meses. Si esas decisiones son malas, la deuda técnica que generan es enorme y se distribuye por todo el sistema antes de que alguien pueda detenerlo.

La tercera es que el costo de oportunidad subió. Los buenos ingenieros son más productivos que nunca cuando trabajan con IA. Sus capacidades se multiplican. Eso significa que el delta entre un buen hire y uno mediocre se amplió. El primero no es 1.5× mejor: es 5× o 10×. Pagar el salario de un mediocre cuando podrías estar pagando el de un buen ingeniero es perder esa diferencia multiplicada.

La traducción comercial de esto es brutal: contratar mal hoy es más fácil que nunca, y cuesta más caro que nunca. Los métodos tradicionales de hiring están optimizados para un mundo que ya no existe, y el costo de seguir usándolos ya no es un costo abstracto. Es un costo medible en deuda técnica, productividad perdida, y tiempo de líderes técnicos que se va en arreglar lo que se contrató mal.

Por qué el mercado no reaccionó todavía

Si todo esto es cierto, ¿por qué la mayoría de las empresas y firmas de recruiting siguen evaluando como en 2018? Hay tres razones que conviene nombrar.

La primera es inercia institucional. Las empresas grandes tienen procesos de hiring que tomaron años en construirse, con compliance legal, sistemas de ATS, comités de revisión, y métricas establecidas. Cambiar esos procesos requiere energía política y consenso interno que pocas organizaciones están dispuestas a movilizar mientras los procesos actuales todavía produzcan resultados aceptables, aunque esos resultados sean cada vez peores en términos relativos.

La segunda es ausencia de alternativas estructuradas. Hay mucho discurso sobre la necesidad de evaluar judgment, criterio, decision-making. Pero no hay frameworks ampliamente adoptados que operacionalicen esos conceptos en preguntas concretas, rúbricas evaluables, y protocolos repetibles. Los equipos de hiring saben que tienen que cambiar algo, pero no saben qué exactamente, y nadie les está dando una herramienta lista para usar.

La tercera es el problema del medidor. Las cosas que sí importan hoy (cómo piensa una persona, cómo decide bajo ambigüedad, cómo lee y mejora código ajeno) son más difíciles de medir que las que importaban antes. Es más fácil evaluar si alguien sabe escribir un algoritmo de búsqueda binaria que evaluar si tiene criterio para descartar tres propuestas arquitectónicas. Y los mercados, cuando enfrentan un cambio donde lo importante es difícil de medir, tienden a seguir midiendo lo fácil aunque haya dejado de ser relevante.

Bondy escribe este documento desde la convicción de que esa tercera razón es la única que se puede atacar con trabajo intelectual serio. La inercia institucional la van a romper los clientes que se cansen de contratar mal. La ausencia de alternativas se rompe escribiendo alternativas. Eso es lo que sigue.

2. Por qué los frameworks existentes no alcanzan

Antes de proponer un método nuevo, hay que ser justos con los métodos que ya existen. No los descartamos porque sean malos. Los describimos para mostrar qué resuelven bien y dónde se quedan cortos para el problema específico que enfrenta el hiring tech hoy.

El modelo de skills y certificaciones

La forma más extendida de evaluar talento técnico sigue siendo el listado de skills: lenguajes de programación, frameworks, herramientas, certificaciones. Una empresa publica una vacante con un listado de requisitos técnicos, los recruiters filtran por coincidencia, los entrevistadores verifican que la persona efectivamente sepa lo que dice saber. Este modelo tiene la ventaja de ser fácilmente operacionalizable (cualquier ATS puede filtrar por keywords) y de producir métricas comparables entre candidatos.

Su límite es estructural: mide acumulación de conocimiento técnico, no capacidad de aplicarlo bajo presión real. Una persona puede tener todas las certificaciones de AWS y no saber decidir cuándo usar una arquitectura serverless versus una basada en contenedores. Otra puede no tener ninguna certificación y haber tomado esa decisión correctamente decenas de veces en producción. El modelo de skills no distingue entre estos dos casos.

Las entrevistas técnicas tradicionales (estilo FAANG)

La generación de entrevistas técnicas que popularizaron Google, Facebook, Amazon, Netflix y otras grandes empresas tech está construida sobre dos pilares: algoritmos en pizarra y diseño de sistemas. Los algoritmos evalúan capacidad de resolver problemas estructurados bajo presión. El diseño de sistemas evalúa capacidad de pensar arquitectónicamente.

Este modelo tiene méritos reales. Es repetible, calibrable entre evaluadores, y pone a todos los candidatos en la misma situación. Pero tiene tres problemas que se volvieron críticos en la era de IA. Primero, los problemas de algoritmos en pizarra son exactamente el tipo de tarea que las herramientas de IA resuelven mejor. Entrenar candidatos a resolverlos sin ayuda evalúa una habilidad cada vez menos relevante. Segundo, el diseño de sistemas tradicional asume escalas hipotéticas (millones de usuarios, latencias subóptimas) que no reflejan los problemas reales de la mayoría de las empresas. Tercero, la presión de la entrevista en pizarra premia a quienes están preparados para entrevistar, no necesariamente a quienes son buenos ingenieros.

Los arquetipos de Will Larson

El trabajo más serio que existe hoy sobre cómo describir el trabajo de ingenieros senior es el de Will Larson en su libro Staff Engineer. Larson identificó cuatro arquetipos para describir cómo operan los ingenieros Staff+: el Tech Lead, que guía la ejecución de un equipo; el Architect, responsable del diseño técnico de un área crítica; el Solver, que se mete en problemas complejos sin camino claro; y el Right Hand, que extiende la atención de un ejecutivo en organizaciones grandes.

La taxonomía de Larson es valiosa por dos razones. Primero, está construida desde la observación empírica. Larson entrevistó a decenas de ingenieros Staff+ en distintas empresas y agrupó patrones reales, no categorías teóricas. Segundo, le da al mundo del recruiting un vocabulario común para hablar de algo que antes era inefable: cómo se ven realmente las distintas formas de ser senior en ingeniería.

Pero la taxonomía de Larson tiene un límite específico para nuestro propósito: está diseñada para describir cómo opera alguien en un rol, no para evaluarlo en un proceso de selección. Larson nos dice qué hace un Architect cuando ya es Architect. No nos dice cómo detectar en una entrevista de 45 minutos si una persona tiene la capacidad cognitiva para serlo. Su marco es descriptivo, no evaluativo. Y esa distinción importa: una recruiter no puede usar Larson directamente en una entrevista. Lo que necesita es un marco que tome la observación de Larson como insumo y la convierta en algo medible.

Los "nuevos arquetipos AI"

En los últimos dos años aparecieron varios intentos de proponer arquetipos nuevos específicamente para la era de IA. Algunos hablan del Vibe Coding Validator, el Prompt Expert, el Domain Expert. Otros proponen taxonomías de Connector, Innovator, Systems Thinker, Translator. Estos intentos son reacciones legítimas al cambio que está atravesando la industria, y algunos identifican dimensiones reales del problema.

Pero la mayoría de estos marcos comparten dos debilidades. La primera es que son superficiales: describen estilos visibles de trabajar con IA, no modos cognitivos profundos. La segunda es que no están construidos para usarse en evaluación: son taxonomías para clasificar candidatos después del hecho, no herramientas para entrevistarlos. Sirven para conversación de LinkedIn, no para procesos de hiring serios.

El hueco que justifica este método

Lo que falta en el mercado es un marco que combine cuatro cosas que hoy no están integradas en ninguna herramienta disponible:

- Que sea evaluativo, no solo descriptivo: que se pueda usar en una entrevista para diagnosticar a un candidato, no solo para etiquetarlo después.

- Que esté anclado en cómo piensa la persona, no en qué sabe hacer: porque lo segundo dejó de ser un buen indicador.

- Que integre la dimensión de IA de manera transversal, no como un modo separado: porque el uso de IA atraviesa todos los modos cognitivos.

- Que produzca diagnósticos comparables entre evaluadores: con rúbricas y protocolos, no solo intuición.

Eso es lo que el método Bondy intenta construir. No reemplaza a Larson: lo usa como insumo conceptual sobre cómo se ven los roles en ingeniería senior. No reemplaza al modelo de skills: lo complementa, porque seguís necesitando saber si una persona conoce las herramientas relevantes. Lo que hace es agregar una capa que hoy no existe en ningún lado: una capa de evaluación cognitiva estructurada, usable por recruiters reales en entrevistas reales, calibrable entre evaluadores, y diseñada para el momento histórico específico en el que estamos.

3. El giro conceptual: de qué hace a cómo piensa

El método Bondy se sostiene sobre un giro conceptual que es simple de enunciar y difícil de operacionalizar: en lugar de evaluar qué sabe hacer una persona, evaluamos cómo piensa cuando enfrenta problemas técnicos. Este capítulo explica por qué ese giro importa, qué significa exactamente, y cuáles son sus implicancias prácticas.

Qué significa "cómo piensa"

Cuando decimos cómo piensa, no nos referimos a personalidad, ni a estilos de aprendizaje, ni a perfiles de Myers-Briggs. Nos referimos a algo más específico: el modo dominante con el que una persona se relaciona con problemas técnicos abiertos. Hay quienes, cuando enfrentan un problema, lo primero que hacen es descomponerlo en alternativas y evaluar trade-offs. Hay quienes lo primero que hacen es buscar qué piezas existentes podrían combinar. Hay quienes lo primero que hacen es leer el contexto antes de actuar. Hay quienes lo primero que hacen es identificar qué puede salir mal.

Estas no son personalidades. Son modos cognitivos: formas estructuradas de aproximarse a la realidad técnica. Una misma persona puede activar distintos modos según la situación, pero todos tenemos un modo dominante: el primero al que recurrimos por defecto, el que aplicamos con menos esfuerzo, el que usamos cuando estamos cansados. Y ese modo dominante tiene consecuencias enormes en el tipo de trabajo que vamos a hacer bien y en el tipo de equipo del que somos un buen complemento o un mal encaje.

Por qué los modos cognitivos son evaluables

Una objeción razonable a este encuadre es que los modos cognitivos suenan a algo difícil de medir, casi a intuición disfrazada de método. Esa objeción merece una respuesta concreta.

Los modos cognitivos no son visibles directamente, pero son inferibles a partir de cómo una persona habla de su trabajo. Cuando le pedís a alguien que te cuente una decisión técnica importante que tomó, no estás escuchando solo qué decidió: estás escuchando la estructura del razonamiento que usó para decidir. Si la persona te cuenta tres alternativas que descartó y por qué, está mostrando que su modo dominante incluye estructuración explícita de decisiones. Si la persona te cuenta solo la solución elegida y la justifica con "me parecía mejor", está mostrando que su modo dominante no incluye esa estructuración.

Esto es exactamente lo que hace décadas la psicología organizacional con conceptos como locus de control, orientación a logro, o estilos de toma de decisiones. No se mide observando, se mide a través del discurso. Y el discurso, cuando se escucha con un marco claro de qué buscar, es sorprendentemente revelador.

La diferencia con evaluar personalidad o soft skills

Es importante marcar qué diferencia este enfoque de los tests de personalidad y las evaluaciones de soft skills que ya existen en el mercado de hiring. Los tests de personalidad miden rasgos generales (extroversión, apertura a la experiencia, conscientiousness) que tienen poca relación específica con cómo una persona aborda problemas técnicos. Las evaluaciones de soft skills miden capacidades comunicacionales y relacionales (trabajo en equipo, comunicación, liderazgo) que son importantes pero genéricas.

Los modos cognitivos del método Bondy son específicamente técnicos: describen cómo piensa una persona cuando se enfrenta a un problema de ingeniería de software, no cómo se comporta en general. Una persona puede ser extrovertida y tener un modo cognitivo de Editor Técnico (preferir mejorar lo existente sobre construir lo nuevo). Otra puede ser introvertida y tener un modo cognitivo de Traductor (excelente para convertir problemas de negocio en spec técnica). La personalidad y los modos cognitivos son dimensiones distintas y relativamente independientes.

Implicaciones prácticas del giro

El giro de "qué sabe" a "cómo piensa" tiene cuatro implicaciones prácticas para el hiring tech.

Primero: las preguntas de entrevista cambian. En lugar de "contame qué tecnologías conocés", preguntás "contame una decisión técnica importante que tomaste el año pasado". La primera mide acumulación de conocimiento. La segunda mide modo cognitivo dominante. Las dos son válidas, pero la segunda diferencia mucho mejor a los buenos de los mediocres en la era actual.

Segundo: la calibración entre evaluadores se vuelve posible. Si lo que estás evaluando es modo cognitivo y tenés un marco claro de qué señales corresponden a qué modo, entonces dos evaluadores pueden llegar al mismo diagnóstico independientemente. Eso es lo que diferencia un método de una intuición personal.

Tercero: los criterios de hiring se vuelven situacionales. En lugar de buscar "el mejor candidato" en abstracto, buscás el modo cognitivo que necesita el equipo en su momento actual. Un equipo que ya tiene tres Arquitectos de Decisiones no necesita un cuarto. Necesita un Editor Técnico que limpie la deuda que esos arquitectos generaron. Esa decisión es invisible si solo evaluás skills.

Cuarto: la evolución profesional del candidato se vuelve un dato relevante. Una persona puede estar empezando a desarrollar un modo cognitivo nuevo, o estar consolidando uno que ya tiene. Eso importa para hires donde el cliente no busca solo cubrir una posición, sino sumar a alguien que va a crecer con el equipo.

Estas cuatro implicaciones convierten al hiring tech de un problema de filtrado en un problema de diagnóstico. Filtrar es decidir si alguien cumple requisitos. Diagnosticar es entender cómo opera esa persona y dónde encaja mejor. El método Bondy es, en esencia, un protocolo de diagnóstico cognitivo aplicado al hiring.

4. Los modos Bondy

El método Bondy organiza la evaluación cognitiva en torno a seis modos. Cada modo describe un modo dominante de relacionarse con problemas técnicos. No son tipos de personalidad ni roles laborales: son formas estructuradas de pensar que se manifiestan cuando una persona enfrenta una decisión, un problema sin solución obvia, o una situación de ambigüedad técnica.

Cuatro de los seis modos son core: se evalúan en todos los candidatos, en todos los procesos. Los otros dos son opcionales: se activan cuando el rol específico lo requiere. Esta distinción no es jerárquica. Los opcionales no son "menos importantes". Es operativa: refleja la realidad de que en una entrevista de 45 minutos no se puede evaluar todo con la misma profundidad, y hay que priorizar lo que más diferencia.

Antes de presentar cada modo, una aclaración importante. Nadie es un solo modo. Todos los candidatos van a mostrar señales de varios. Lo que el método busca identificar es el modo dominante (el modo más fuerte) y el rango (cuántos otros modos puede activar cuando la situación lo pide). Esa distinción la desarrollamos en el capítulo 5.

Modo 1: Arquitecto de Decisiones (core)

El Arquitecto de Decisiones no diseña sistemas en abstracto. Diseña el proceso de decisión que precede al sistema. Cuando enfrenta un problema, su primer movimiento es identificar las alternativas posibles, evaluar trade-offs explícitos, y argumentar qué descarta y por qué. Su valor no está en elegir la solución correcta. Está en estructurar el razonamiento que hace que la elección sea defendible.

Las raíces conceptuales de este modo están en la literatura sobre toma de decisiones bajo incertidumbre, particularmente en el trabajo de Herbert Simon sobre racionalidad acotada y la distinción entre decisiones programadas y no programadas. En el contexto del hiring tech actual, este modo es crítico porque la mayoría de las decisiones técnicas relevantes son no programadas: no tienen una respuesta correcta única, y lo que diferencia a un buen ingeniero es la calidad del proceso de decisión, no la elección final.

Una persona con modo dominante Arquitecto de Decisiones suele hablar de sus proyectos en términos de "elegimos X aceptando que perdíamos Y". Distingue intuitivamente entre decisiones reversibles e irreversibles, y trata estas últimas con más cuidado. Considera quién va a mantener lo que se construye. Y, crucialmente, puede articular qué descartó incluso cuando no se le pregunta directamente.

Modo 2: Conector (core)

El Conector resuelve problemas combinando piezas que ya existen. Su valor no está en construir desde cero. Está en saber qué construir y qué tomar prestado. Conoce el ecosistema técnico amplio (APIs, servicios, librerías, plataformas) y su criterio para elegir herramientas se basa en fit con el problema, no en familiaridad o popularidad.

Este modo es probablemente el que más se transformó con la llegada de las herramientas de IA. Antes, conectar piezas existentes era visto como "menos creativo" que construir desde cero. Hoy, en un mundo donde casi todo lo que se necesita ya existe en alguna forma, el Conector se convirtió en uno de los modos cognitivos más valiosos del mercado. Las empresas que crecen rápido lo hacen porque tienen Conectores que arman soluciones en semanas con piezas existentes, no porque tienen ingenieros heroicos construyendo todo a mano.

Una persona con modo dominante Conector suele describir sus soluciones en términos de "conecté X con Y usando Z". Habla con familiaridad de contratos de API, modos de falla, dependencias externas. Tiene opiniones informadas sobre el ecosistema, no solo sobre las herramientas que usó. Y, crucialmente, puede explicar por qué no construyó algo desde cero.

Modo 3: Editor Técnico (core)

El Editor Técnico mejora lo que ya existe. Su valor está en leer código ajeno, identificar deuda técnica relevante, y ejecutar mejoras que reducen complejidad sin cambiar comportamiento. En la era de IA, donde se genera enormes volúmenes de código sin la disciplina que tendría un ingeniero entrenado escribiéndolo a mano, este modo se volvió crítico.

La intuición común es que un buen ingeniero es alguien que escribe código de calidad. Pero en un mundo donde la IA produce gran parte del código, el cuello de botella se mueve. Lo escaso ya no es escribir código: es leerlo críticamente, identificar qué partes son problemáticas, y mejorarlas sin romper lo que funciona. Un Editor Técnico es una persona que tiene la disciplina y el criterio para hacer ese trabajo, que es probablemente el menos glamoroso y más crítico del momento.

Este modo tiene raíces conceptuales en la literatura sobre refactoring (particularmente el trabajo de Martin Fowler) y en la práctica de code review como mecanismo de calidad. Pero el método Bondy lo trata como un modo cognitivo, no como una habilidad técnica: lo que evalúa no es si alguien sabe refactorear, sino si su forma natural de relacionarse con código existente es de mejora deliberada o de evitación.

Una persona con modo dominante Editor Técnico suele hablar con entusiasmo de refactors que hizo. Distingue entre tipos de deuda técnica (urgente, tolerable, estructural). Tiene opiniones sobre legibilidad, naming, organización de archivos. Y, crucialmente, no propone reescribir desde cero como respuesta default a código heredado.

Modo 4: Traductor (core)

El Traductor convierte entre mundos. Toma un problema de negocio y lo baja a una especificación técnica accionable. Toma una limitación técnica y la explica en términos de impacto de negocio sin perder rigor. Su valor no está en saber más de un lado o del otro. Está en operar como puente entre dos lenguajes que rara vez se encuentran.

Las raíces conceptuales de este modo están en la literatura sobre boundary spanners en psicología organizacional, particularmente el trabajo de Michael Tushman en los años setenta sobre los individuos que conectan grupos con códigos comunicacionales distintos. Tushman mostró que en organizaciones complejas, los boundary spanners son desproporcionadamente importantes para el desempeño del conjunto, aunque son difíciles de identificar con métodos tradicionales de evaluación.

En el hiring tech actual, el Traductor se volvió crítico por una razón específica: los equipos de ingeniería están cada vez más expuestos a conversaciones con producto, negocio, y clientes finales. Las empresas tech ya no pueden permitirse ingenieros que solo hablen con otros ingenieros. Y al mismo tiempo, la IA volvió más fácil que nunca generar código sin entender por qué. Eso hace que la capacidad de articular el porqué de negocio detrás de una decisión técnica sea uno de los diferenciadores más fuertes entre un buen ingeniero y uno reemplazable.

Una persona con modo dominante Traductor suele explicar cosas técnicas a no técnicos sin condescender. Identifica cuándo un problema "técnico" es en realidad un problema de comunicación. Adapta su nivel de detalle según el interlocutor. Y, crucialmente, no menosprecia a los stakeholders no técnicos.

Modo opcional 1: Operador

El Operador ejecuta bien lo que le piden. Tiene disciplina, atención al detalle, capacidad de seguir instrucciones complejas sin perder contexto. Es el modo dominante más común en juniors competentes y, en niveles senior, se asume como base sobre la cual descansan los demás modos.

Por eso es opcional: solo lo evaluamos explícitamente cuando el rol es junior, cuando el candidato va a trabajar con supervisión cercana, o cuando el cliente específicamente necesita capacidad de ejecución. En roles senior+, si alguien no puede ejecutar, eso aparece naturalmente en los otros modos. Un Arquitecto de Decisiones que no puede traducir sus decisiones en ejecución no es realmente un Arquitecto. Operador como modo independiente solo aporta cuando es la dimensión central del rol.

Modo opcional 2: Guardián

El Guardián se obsesiona con lo que puede salir mal. Piensa en seguridad, edge cases, observabilidad, resiliencia. No frena al equipo: asegura que lo que el equipo construye no explote

en producción. Su modo cognitivo es defensivo en el mejor sentido del término: no rechaza el cambio, sino que lo prepara para sobrevivir al contacto con la realidad.

Este modo es opcional porque su criticidad depende del contexto. En una startup en fase de exploración, donde mover rápido es más importante que prevenir, un Guardián puro puede ser un freno. En una fintech, una empresa de salud, o cualquier sistema con impacto crítico en producción, un Guardián es indispensable. Activamos este modo cuando el rol lo requiere y lo dejamos afuera cuando no.

Las raíces conceptuales del Guardián están en la literatura sobre safety engineering y high reliability organizations. En el hiring tech actual, este modo se volvió crítico por una razón específica: la velocidad con la que se construye software hoy hace que los errores en producción sean más frecuentes y más costosos. Y la IA, al producir código sin contexto humano profundo, introduce categorías nuevas de fallas que solo un Guardián entrenado puede anticipar.

5. Modo dominante y rango

El método Bondy está construido sobre una afirmación que merece defenderse explícitamente: nadie es un solo modo. Toda persona muestra rasgos de varios modos cognitivos, y lo que diferencia a un candidato fuerte de uno débil no es solo cuál es su modo dominante, sino cuántos otros modos puede activar cuando la situación lo requiere.

Esta distinción entre modo dominante y rango es central porque resuelve dos problemas que tendría un método más simple.

Por qué no alcanza con identificar el modo dominante

Un método que solo identificara el modo dominante de cada candidato produciría diagnósticos demasiado rígidos. Diría cosas como "este candidato es Arquitecto de Decisiones" y dejaría al cliente con la impresión de que esa persona no puede hacer nada más. Pero la realidad es que un buen Arquitecto de Decisiones también necesita ser Conector y Traductor, no como modos dominantes sino como capacidades activables cuando el contexto lo pide.

La métrica del rango captura exactamente eso: cuántos modos adicionales puede activar la persona cuando el modo dominante no es suficiente. Un candidato con modo dominante Arquitecto de Decisiones y rango fuerte hacia Traductor y Editor Técnico es radicalmente distinto de un candidato con el mismo modo dominante pero rango pobre. El primero es un líder técnico potencial. El segundo es un especialista que necesita un equipo que compense lo que no hace.

Cómo se mide el rango

El rango se mide a través del scoring numérico (1–5) que el método aplica a cada modo core. Un candidato con tres modos en score 4 o más tiene rango amplio. Un candidato con un solo modo en score 4+ es un perfil especializado. Un candidato sin ningún modo en 4+ está en problemas, independientemente de qué modo sea su dominante.

Esta medición tiene dos virtudes prácticas. Primero, hace explícito algo que los recruiters experimentados intuyen pero rara vez articulan: un candidato "bien redondeado" no es vago: es un candidato con rango amplio en modos específicos y observables. Segundo, permite que el cliente entienda exactamente qué está comprando: no "un senior developer", sino "un Arquitecto de Decisiones con rango hacia Traductor y Editor, que va a ser fuerte en X tipo de problema y débil en Y tipo de problema".

Por qué algunos candidatos cambian de modo dominante

Una observación que el método incorpora desde la literatura de psicología organizacional es que el modo dominante de una persona puede evolucionar a lo largo de su carrera, pero no de forma rápida ni voluntaria. Los modos cognitivos se forman a través de años de experiencia repetida en situaciones similares, y se consolidan cuando esa experiencia es validada por resultados positivos.

Esto significa que no se puede "entrenar" a alguien para cambiar su modo dominante en seis meses. Lo que sí se puede es ampliar el rango: ayudar a una persona a desarrollar modos cognitivos secundarios que complementen el dominante. Esa es información valiosa para clientes que están pensando en hires de mediano plazo: un candidato con modo dominante consolidado pero rango incipiente puede ser una apuesta inteligente si el cliente está dispuesto a invertir en su desarrollo.

Las patologías de cada modo dominante

Hay un punto que es importante hacer explícito y que distingue al método Bondy de marcos más optimistas: cada modo dominante tiene una patología asociada cuando opera sin rango o sin contrapesos. El método no celebra a todos los modos por igual. Reconoce que cada uno, llevado al extremo, se vuelve disfuncional.

El Arquitecto de Decisiones puro se convierte en el ingeniero que paraliza al equipo con análisis interminables y nunca llega a ejecutar. El Conector puro se convierte en el ingeniero que arma soluciones frágiles porque no entiende profundamente ninguna de las piezas que conecta. El Editor Técnico puro se convierte en el ingeniero obsesivo que bloquea features porque siempre hay algo más que mejorar antes. El Traductor puro se convierte en el político técnico sin sustancia, alguien que comunica bien pero no produce.

Estas patologías no son razones para descartar a un candidato: son señales para entender su rango. Un candidato con modo dominante Arquitecto de Decisiones y rango amplio no va a caer en la patología, porque puede activar otros modos cuando la situación lo requiere. Un candidato con el mismo modo dominante y rango estrecho va a caer en la patología tarde o temprano. Identificar esa diferencia es exactamente lo que el método busca hacer.

6. Aplicación en hiring

Hasta acá, este documento es un marco conceptual. Este capítulo describe cómo ese marco se traduce en un proceso de hiring concreto, desde el primer contacto con el cliente hasta la presentación del candidato seleccionado. La operacionalización detallada (preguntas de entrevista, rúbricas, plantillas) está en el Manual de Implementación, que es un documento separado. Acá presentamos la lógica del proceso, no sus mecánicas.

El kick-off con el cliente

Todo proceso de hiring serio empieza por entender qué está comprando el cliente, y la mayoría de los procesos fracasan en este primer paso. El cliente típicamente describe lo que necesita en términos de seniority y skills ("busco un senior backend con experiencia en Python y AWS"), pero rara vez articula qué tipo de modo cognitivo necesita el equipo en su momento actual. Esa traducción es responsabilidad del recruiter, y el método Bondy la convierte en una conversación estructurada.

El kick-off ideal cubre tres preguntas. Primero: ¿qué tipo de problemas va a tener que resolver esta persona en sus primeros seis meses? La respuesta a esa pregunta revela qué modos son críticos. Si los problemas son arquitectónicos, el cliente necesita un Arquitecto de Decisiones. Si los problemas son de integración, necesita un Conector. Si los problemas son de deuda técnica, necesita un Editor Técnico. Segundo: ¿con qué equipo va a trabajar, y qué modos cognitivos ya están cubiertos? Eso evita contratar redundancia y permite buscar complemento. Tercero: ¿cuál es el nivel de criticidad de producción del trabajo de esta persona? Eso decide si activamos el modo Guardián.

La pre-screening

La pre-screening tradicional filtra por keywords del CV. La pre-screening del método Bondy filtra por evidencia inicial de modo cognitivo en el LinkedIn, GitHub, o portfolio del candidato. Esto no produce un diagnóstico (es demasiado temprano para eso), pero produce hipótesis tempranas que orientan la entrevista a fondo.

Por ejemplo, un candidato cuyo LinkedIn está lleno de descripciones de proyectos donde menciona "lideré la decisión de migrar de X a Y considerando trade-offs A, B, C" probablemente tiene rasgos de Arquitecto de Decisiones, y la entrevista puede empezar verificando esa hipótesis. Un candidato cuyo GitHub muestra muchos forks y contribuciones a proyectos open source probablemente tiene rasgos de Conector. Estas hipótesis no son conclusiones: son hipótesis testeables en la entrevista.

La entrevista a fondo

La entrevista del método Bondy dura 45–60 minutos y está estructurada para evaluar los cuatro modos core (más los opcionales si aplican) a través de un set reducido de preguntas multipropósito. Cada pregunta ilumina dos o tres modos simultáneamente, y el recruiter no se concentra en hacer "una pregunta por modo", sino en escuchar qué modos se activan en cada respuesta.

La conversación es importante, pero la disciplina es el qué escuchar. Un recruiter entrenado en el método sabe que cuando un candidato cuenta una decisión técnica, tiene que escuchar tres cosas: si menciona alternativas que descartó (Arquitecto), si menciona herramientas existentes que evaluó (Conector), y si conecta la decisión con impacto de negocio (Traductor). Esa escucha simultánea de múltiples dimensiones es lo que vuelve eficiente la entrevista de 45 minutos.

El scoring post-entrevista

Inmediatamente después de la entrevista, el recruiter completa un scoring numérico (1–5) para cada modo core y una evaluación cualitativa (verde/amarillo/rojo) para cada modo opcional. Este scoring no se hace durante la entrevista. Se hace en frío, con las notas crudas a mano, en los 30 minutos siguientes mientras la conversación está fresca.

Las reglas del scoring son explícitas para evitar la inflación de notas: nunca se asigna un 5 sin evidencia de que el candidato es referencia para otros en ese modo; nunca se asigna un 1 sin evidencia activa de señales rojas. Un candidato que no mostró evidencia de un modo recibe un 3 (neutral) y se nota como "no evaluable con los datos disponibles". Esta disciplina es crucial para que los scorings sean comparables entre candidatos y entre evaluadores.

La presentación al cliente

La presentación al cliente es donde el método se vuelve visible. En lugar de presentar al candidato como "senior con 8 años de experiencia", lo presentamos como "Arquitecto de Decisiones con rango hacia Traductor y Editor Técnico". Esa diferencia de lenguaje no es estética: es lo que permite al cliente tomar una decisión informada sobre si el candidato encaja en su contexto específico.

La presentación incluye al menos una cita textual del candidato. Las citas textuales son el ancla empírica del diagnóstico: muestran al cliente que el modo cognitivo que estamos describiendo no es una opinión nuestra, sino una inferencia anclada en lo que el candidato realmente dijo. Sin cita textual, el diagnóstico es opinable. Con cita textual, es defendible.

La calibración entre evaluadores

Ningún método de evaluación cognitiva sobrevive sin un protocolo de calibración entre evaluadores. Si dos recruiters entrevistan al mismo candidato y llegan a diagnósticos distintos, el

método pierde credibilidad y se desintegra en intuiciones individuales. Por eso el método Bondy incluye un protocolo de calibración mensual: una entrevista grabada que todos los recruiters puntúan independientemente, y se discuten las diferencias hasta llegar a una interpretación común o documentar explícitamente los puntos de ambigüedad.

Esta práctica es lo que mantiene viva la metodología. Sin calibración periódica, cada recruiter desarrolla su propia interpretación de los modos y, en seis meses, el método deja de ser un método compartido para volver a ser un conjunto de criterios individuales. Con calibración, el método se afina con el tiempo y se enseña a recruiters nuevos sin depender exclusivamente de mentoría uno a uno.

7. Limitaciones honestas

Este capítulo es el que distingue un método serio de un white paper de consultora. Todo método tiene límites. Hay situaciones donde no funciona bien, hay tipos de candidatos que no captura, hay problemas que no resuelve. Lo que sigue es un inventario explícito de esas limitaciones.

No reemplaza la evaluación técnica

El método Bondy evalúa cómo piensa un candidato cuando enfrenta problemas técnicos. No evalúa si conoce las tecnologías específicas que el rol requiere. Esa evaluación sigue siendo necesaria y debe hacerse por separado, idealmente por alguien del equipo del cliente que tiene el conocimiento técnico para verificar si el candidato puede usar las herramientas que va a necesitar.

La razón es simple: el método Bondy es un protocolo de diagnóstico cognitivo, no un test técnico. Confundirlo con esto último sería un error grave. Un candidato puede ser un Arquitecto de Decisiones brillante y no saber usar Kubernetes. Eso lo hace inadecuado para un rol que requiere Kubernetes, independientemente de su modo cognitivo. Las dos evaluaciones son complementarias, no sustituibles.

Funciona mejor en niveles mid y senior que en juniors

El método se basa en escuchar cómo un candidato habla de su propia experiencia profesional. En niveles junior, donde la experiencia es limitada, el material disponible para evaluar es más escaso. Esto no hace el método inútil para juniors (el modo Operador opcional fue diseñado precisamente para esos casos), pero sí significa que el diagnóstico cognitivo es menos preciso cuando el candidato tiene menos historia profesional para discutir.

En la práctica, esto significa que para juniors el método produce hipótesis tempranas más que diagnósticos firmes. Y la decisión de hire en juniors debe pesar más otros factores (capacidad de aprendizaje, motivación, fit cultural) que el diagnóstico cognitivo solo.

Depende de la calidad del recruiter

Aunque el método busca producir diagnósticos comparables entre evaluadores, no elimina la importancia del juicio individual del recruiter. Un recruiter sin experiencia en hiring tech, o sin entrenamiento en escucha activa, va a tener dificultades para aplicar el método con precisión incluso teniendo todas las rúbricas a mano.

Esto significa que el método tiene un piso de competencia: requiere recruiters con experiencia técnica suficiente para entender lo que un candidato cuenta sobre sus proyectos, y con habilidades de escucha activa entrenadas. No es una herramienta plug-and-play para equipos de hiring sin experiencia. Es una herramienta de profesionalización para equipos que ya tienen experiencia y quieren sistematizarla.

No predice fit cultural

El modo cognitivo de una persona dice cómo piensa, no cómo se va a comportar en una cultura específica. Un Arquitecto de Decisiones puede prosperar en una cultura que valora el debate técnico estructurado y fracasar en una cultura que premia la velocidad y la acción intuitiva. Esa dimensión, el fit entre la persona y la cultura del equipo, es ortogonal al diagnóstico cognitivo y debe evaluarse por separado.

Bondy maneja esa dimensión a través de su trabajo de Compatibilidad Dinámica, que es un componente complementario del proceso de hiring pero no es parte del método Bondy en sentido estricto. El método cognitivo y la evaluación cultural son dos modos distintos sobre el mismo candidato, y los dos son necesarios para una decisión de hire informada.

No predice evolución a largo plazo

El método produce un diagnóstico de cómo piensa el candidato hoy. No predice cómo va a evolucionar en los próximos 18 meses o 5 años. Hay candidatos que tienen modos cognitivos aún en desarrollo y que pueden crecer significativamente con el contexto correcto, y hay candidatos que tienen modos cognitivos consolidados que no van a cambiar mucho. Distinguir entre estos casos requiere data longitudinal que el método actual no tiene.

Esta limitación es importante para clientes que están pensando en hires de largo plazo o en construir cantera. El método les puede decir qué tienen hoy, pero no puede prometer qué van a tener en tres años. Es una limitación real que prefiero hacer explícita antes que dejar implícita.

No funciona igual en todas las culturas

El método fue desarrollado y probado principalmente en empresas tech de LATAM con clientes globales. Hay algunas señales de que ciertas dimensiones funcionan distinto en culturas con códigos comunicacionales muy distintos a los de la región. Por ejemplo, el modo Traductor puede expresarse de formas más sutiles en culturas asiáticas donde la comunicación directa es menos valorada, y el método puede subestimar candidatos con esos modos.

Esta limitación es algo que el método va a tener que abordar a medida que se aplique en más contextos culturales. Por ahora, lo señalamos como un borde conocido y una invitación a colegas de otras regiones a probarlo y reportar resultados.

8. Cierre y proyección

Este documento articula la versión 1.0 del método Bondy. Esa numeración no es decorativa: implica una expectativa explícita de que este método va a evolucionar. Las próximas versiones van a incorporar lo que aprendamos en los próximos 12–18 meses de uso operativo, y van a ajustar lo que la realidad muestre que está mal calibrado.

Hay tres líneas de trabajo concretas que el método va a desarrollar en su próxima fase. La primera es la incorporación de casos resueltos: ejemplos reales (anonimizados) de candidatos evaluados con el método, con scoring, diagnóstico, y resultado en el cliente seis meses después. Estos casos son el componente más importante para la enseñanza del método a recruiters nuevos, y solo pueden construirse con uso operativo prolongado.

La segunda es la calibración de la dimensión longitudinal: identificar qué señales tempranas predicen evolución cognitiva en los siguientes 18 meses. Esto requiere seguimiento de candidatos contratados a través del método, y un protocolo de feedback con los clientes que reporten cómo le fue al hire después del proceso. Es un trabajo lento pero indispensable si queremos pasar de "diagnóstico actual" a "predicción razonable".

La tercera es la apertura del método a la crítica de la comunidad. Este documento se va a publicar en alguna forma: probablemente como white paper en el sitio de Bondy, posiblemente como libro corto. La intención no es comercial: es invitar a otros profesionales del recruiting tech, a CTOs, a investigadores de HR, a usar el método y reportar lo que funciona y lo que no. Bondy se beneficia más de tener un método validado externamente que de tener un secreto comercial guardado.

La convicción de fondo, la que sostiene todo este documento, es que el hiring tech está atravesando un momento histórico que requiere herramientas nuevas. Los métodos que funcionaron entre 2010 y 2020 dejaron de funcionar, y los que los van a reemplazar todavía no están consolidados. Este es exactamente el momento en el que vale la pena escribir, publicar, debatir, y refinar. Las cosas se vuelven obvias después de que alguien las articula con suficiente claridad como para que el resto pueda verlas. Este documento intenta esa articulación.

No espero que el método Bondy sea adoptado tal cual por toda la industria. Espero que abra una conversación sobre cómo evaluar talento técnico cuando lo escaso ya no es saber hacer, sino saber pensar. Si alguien lee este documento, lo critica, lo mejora, y construye algo distinto que funciona mejor, eso ya es un éxito. La industria entera necesita este tipo de trabajo, y prefiero ser parte del problema de tener demasiados métodos compitiendo a ser parte del problema de no tener ninguno.

Bibliografía

Fuentes primarias

Larson, W. (2021). Staff Engineer: Leadership beyond the management track. Stripe Press. Capítulo de arquetipos disponible en: https://staffeng.com/guides/staff-archetypes/

Stack Overflow. (2025). Developer Survey 2025. https://survey.stackoverflow.co/2025/

Google Cloud. (2025). DORA State of DevOps Report 2025. Concepto de IA como “mirror and multiplier” en organizaciones cohesivas versus fragmentadas.

Contexto del cambio en hiring técnico

Bruneaux, T. (2026). Hiring for AI-native developers in 2026. DX. https://getdx.com/blog/hiring-developers/

Built In. (2025). How to Reshape the Developer Hiring Process for the AI Era. https://builtin.com/articles/developer-hiring-process-ai-era

Ewerlöf, A. (2025). Staff archetypes can be anti-patterns. https://blog.alexewerlof.com/p/staff-archetypes-are-anti-patterns

Fundamento teórico (psicología organizacional y toma de decisiones)

Simon, H. A. (1947). Administrative Behavior: A Study of Decision-Making Processes in Administrative Organization. Macmillan. Concepto de racionalidad acotada y distinción entre decisiones programadas y no programadas, base conceptual del modo Arquitecto de Decisiones.

Tushman, M. L. (1977). Special boundary roles in the innovation process. Administrative Science Quarterly, 22(4), 587–605. Concepto de boundary spanners, base conceptual del modo Traductor.

Morin, E. (1990). Introduction à la pensée complexe. ESF éditeur. Pensamiento complejo y la lógica de modo dominante con rango activable.

Fowler, M. (1999). Refactoring: Improving the Design of Existing Code. Addison-Wesley. Práctica de refactoring como disciplina técnica, base conceptual del modo Editor Técnico.

Nota sobre desarrollo propio

Los seis modos Bondy (Arquitecto de Decisiones, Conector, Editor Técnico, Traductor, Operador, Guardián) son desarrollo original de Bondy Group, construidos a partir de 18 años de práctica en recruiting técnico y formación previa en psicología organizacional. Las fuentes citadas son influencias conceptuales y antecedentes teóricos, no fuentes directas del marco. La integración del marco evaluativo, las rúbricas de scoring, las preguntas multipropósito y el protocolo de calibración son trabajo propio.

Versión 1.0. Abril 2026

Bondy Group