مراجعة حالة الفن 2026

هندسة الذكاء الوكيلي، نماذج الاستدلال المتقدمة، وبنى **AGI** الموجّهة للإنتاج

STATE-OF-THE-ART REVIEW: AGENTIC AI, ADVANCED REASONING MODELS & AGI-ORIENTED SYSTEMS

النطاق — دمج علمي لتقارير Google Drive المؤرخة 11 أغسطس 2026، ثم إكمال الفجوات عبر مصادر أولية محدثة حتى 16 أغسطس 2026. اللغة عربية RTL مع المصطلحات والـ keywords الإنجليزية الأصلية.

صيغة: Arabic / English technical keywords • Font: Amiri • Evidence-first synthesis

02 الملخص التنفيذي ومنهجية المراجعة

Scientific synthesis of the August-11 Drive corpus + primary-source verification as of 16 Aug 2026

تجمع هذه المراجعة جميع تقارير 11 أغسطس الموجودة في Google Drive في نموذج معرفي واحد، ثم تعيد فحص الادعاءات التي يمكن أن تتغير سريعاً مقابل مصادر أولية: صفحات النماذج الرسمية، وثائق الأطر والبروتوكولات، تقارير المختبرات، أوراق arXiv/ACL، ووثائق الجهات التنظيمية. الهدف ليس إعادة جمع أخبار متفرقة، بل تحويلها إلى خريطة هندسية قابلة للبناء: ما هي طبقات النظام الوكيلي الحديث؟ أين يحدث الاستدلال؟ كيف تُدار الذاكرة والحالة؟ كيف تُعزل الأدوات؟ وكيف تُقاس موثوقية المسار الكامل بدلاً من دقة الإجابة النهائية فقط؟

الخلاصة التقنية هي أن حالة الفن في 2026 لا تُختزل في «أفضل نموذج». الأداء التشغيلي هو حاصل ضرب نموذج الاستدلال في الـ harness، وسياسة ميزانية الاستدلال، وإدارة السياق، والتنسيق متعدد الوكلاء، وإتاحة الأدوات، والتحقق، والعزل الأمني، والمراقبة. GPT-5.6 Sol يقدم max وultra مع Programmatic Tool Calling وتنسيق متوازٍ؛ Gemini 3.1 Pro يبرهن قوة multimodal reasoning وcontext 1M؛ Claude Opus 4.8 يركز على agentic judgment؛ Kimi K3 يضيف نموذجاً مفتوح الأوزان 2.8T وcontext 1M؛ وDeepSeek-V4 مؤكد الإصدار في أبريل 2026. [R01][R07][R11][R15][R16]

- المعيار الأول: مصدر أولي أو ورقة علمية لكل ادعاء تقني حاسم.
- المعيار الثاني: فصل benchmark capability عن system capability وعن production reliability.
- المعيار الثالث: عدم اعتماد نسب السوق أو ROI من تقارير تسويقية ثانوية كأساس علمي.
- المعيار الرابع: توثيق الادعاءات الداخلية غير المتحققة بدلاً من إعادة تدويرها.

> النتيجة — الحد الأعلى الواقعي اليوم هو أنظمة Agentic متقدمة ذات استقلالية طويلة ومضبوطة؛ لا يوجد برهان علمي عام يثبت تحقق AGI الشامل غير المقيد.

Keywords: *Agentic AI • AGI • Reasoning Models • Harness • Tool Use • Evaluation • Reliability*

03 دمج **corpus** أغسطس: ما أُبقي وما صُحّح

Evidence audit of the 22 matched Aug-11 reports

التقارير الداخلية تتفق بقوة على ستة محاور: الانتقال من chatbots إلى outcome-oriented agents؛ هيمنة orchestration/state/memory/tooling كطبقات مستقلة؛ أهمية MCP وA2A؛ صعود agentic BI؛ تحول الاختبارات نحو long-horizon tasks؛ وضرورة guardrails وHITL. هذه المحاور أُبقيت لأنها تتطابق مع الوثائق الأولية الحديثة. في المقابل، ظهرت ادعاءات تحتاج تصحيحاً: بعض التقارير تعاملت مع تواريخ تنظيمية أو أسماء نماذج أو نسب سوقية كحقائق مستقرة، بينما لا تسمح المراجعة العلمية بذلك.

أهم تصحيح تنظيمي: اعتبار 2 أغسطس 2026 «دخول كل متطلبات high-risk حيز التنفيذ» غير دقيق. وفق FAQ الرسمي للاتحاد الأوروبي، يبدأ إنفاذ بعض سلطات الشفافية وGPAI في ذلك التاريخ، بينما تطبيق قواعد Annex III high-risk انتقل إلى 2 ديسمبر 2027، والأنظمة عالية الخطورة المدمجة في المنتجات المنظمة إلى 2 أغسطس 2028. [R38] كذلك لم تُعتمد في الاستنتاجات أرقام 41% للحصة المفتوحة، 171% ROI، أو نسب NLP-BI ما لم تكن مدعومة بدراسة أولية قابلة للتدقيق.

ومن ناحية النماذج، تم تثبيت Claude Opus 4.8 من مصدر Anthropic المباشر. أسماء مثل «Opus 5» ونتيجة 95.5% على ARC-AGI-3 في بعض النسخ لم تُعامل كحقيقة مثبتة. OpenAI وKimi ينشران جداول مقارنة تتضمن Claude Fable 5، لكن غياب صفحة Anthropic مباشرة ضمن نطاق هذه المراجعة يجعلنا نتجنب استخدامه كمرساة هندسية مستقلة. هذه السياسة تمنع تحول تقرير تجميعي إلى حلقة تضخيم للادعاءات.

- Accepted: MCP 2026-07-28 stateless core, MRTR, routing, authorization hardening. [R26]
- Accepted: Microsoft Agent Framework هو successor يجمع أفكار AutoGen وSemantic Kernel مع workflows وحالة طويلة. [R17]
- Accepted: long-running agent harnesses تحتاج checkpointing/context management وليس مجرد context window أكبر. [R08][R09]
- Excluded from scientific core: secondary market forecasts and unverifiable model-score claims.

> قاعدة الدمج — كل حقيقة حديثة في هذا التقرير تحمل مرجعاً أولياً أو تُوسم صراحة كاستنتاج هندسي.

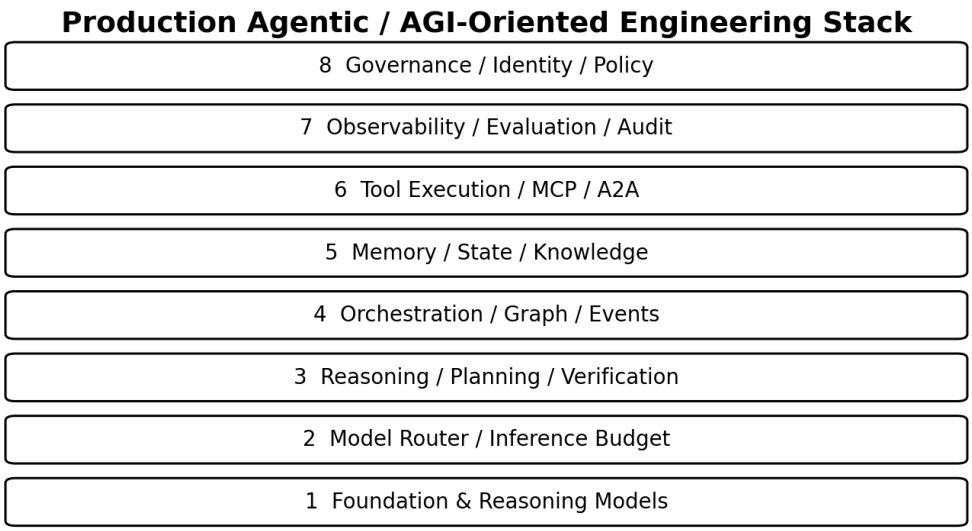
Keywords: Evidence Hierarchy • Source Audit • Reproducibility • Primary Sources

04 الخريطة الهندسية لحالة الفن

Eight-layer production stack

أفضل طريقة لقراءة مشهد 2026 هي كمنظومة من ثماني طبقات متراكبة. الطبقة الأولى هي model substrate: نماذج reasoning/multimodal/open-weight. فوقها model router الذي يقرر النموذج وreasoning effort وparallelism. ثم طبقة reasoning التي تقوم بالتخطيط والتحقق والاستدعاء البرنامجي. بعدها orchestration التي تمثل العمل كـ graph أو event network. الذاكرة والحالة تحفظ checkpoint وsemantic/episodic/working memory. طبقة الأدوات والبروتوكولات تربط النظام بالعالم. وأخيراً observability/evaluation ثم identity/governance والسياسات.

هذا التفكيك يفسر لماذا قد يكون model leaderboard مضللاً في الإنتاج. نفس النموذج قد يختلف أداؤه جذرياً مع harness مختلف أو سياسة ضغط سياق مختلفة أو أداة terminal مختلفة. كما أن «الذكاء» التشغيلي يحتاج مسار side effects قابلاً للمراجعة: من طلب المستخدم إلى خطة، إلى authorization، إلى sandbox، إلى execution ثم verification. لذلك يجب بناء AGI-oriented systems كـ control plane + execution plane + evidence plane وليس كـ prompt عملاق فقط. [R02][R09][R17][R18][R35]



الشكل 1 — طبقات النظام الوكيلي الإنتاجي. المصطلحات الإنجليزية مقصودة لتوحيدها مع وثائق الأطر.

> **مبدأ معماري** — Model capability ≠ Agent capability ≠ System reliability. يجب قياس المستويات الثلاثة منفصلة.

Keywords: *Control Plane • Execution Plane • Evidence Plane • State Machine • Model Router*

05 المشهد الحالي لنماذج الاستدلال

Frontier models should be selected by workload, harness and budget

في أغسطس 2026، لا توجد هيمنة واحدة على كل أنماط العمل. GPT-5.6 Sol يركز على efficiency + max/ultra + programmatic tool calling؛ Gemini 3.1 Pro يجمع reasoning عالياً، multimodality، و1M context؛ Claude Opus 4.8 يركز على الاستمرارية والحكم في المهام الوكيلية؛ Kimi K3 يقدم frontier open weights مع MoE ضخم وسياق مليون؛ DeepSeek-V4 مؤكد كإصدار رئيسي مفتوح في 24 أبريل. [R01][R07][R11][R15][R16]

المقارنة العلمية يجب أن تحافظ على harness parity. مثلاً Gemini 3.1 Pro يسجل رسمياً 77.1% على ARC-AGI-2، و94.3% على GPQA Diamond، و68.5% على Terminal-Bench 2.0 باستخدام Terminus-2، و80.6% على SWE-Bench Verified single-attempt. [R11] أما GPT-5.6 فيبلغ 92.2% BrowseComp و62.6% OSWorld 2.0 في صفحة الإطلاق، ويضيف ultra الذي ينسق أربعة وكلاء افتراضياً. [R01] لا يجوز خلط هذه الأرقام في ranking واحد دون توحيد وقت التنفيذ والأدوات والجهد والنسخة.

النموذج	قوة هندسية أساسية	ملاحظة التقييم
GPT-5.6 Sol	max/ultra, tools, long-horizon knowledge work	قارن بال harness وال reasoning effort [R01]
Gemini 3.1 Pro	multimodal reasoning, 1M context	أرقام رسمية واسعة ومفصلة [R11]
Claude Opus 4.8	agentic judgment, computer/browser use	مصدر Anthropic مباشر [R07]
Kimi K3	2.8T MoE, open weights, 1M context	MXFP4/MXFP8, max reasoning [R15]
DeepSeek-V4	open-model frontier release	الإصدار مؤكد؛ التفاصيل من model card [R16]

Keywords: *Max Reasoning • Ultra • Multimodal • Open Weights • Harness Parity*

06 GPT-5.6 Sol: من reasoning إلى agent runtime

Programmatic Tool Calling, max/ultra and multi-agent execution

يمثل GPT-5.6 Sol انتقالاً واضحاً من نموذج يجيب إلى نموذج يشارك في تنفيذ برنامج عمل. وفق OpenAI، يوفر `max` وقتاً أكبر للاستكشاف والتحقق، بينما `ultra` ينسق أربعة وكلاء بالتوازي افتراضياً، ويمكن بناء تجربة مشابهة في API عبر multi-agent beta. الأكثر أهمية هندسياً هو Programmatic Tool Calling: يستطيع النموذج كتابة وتشغيل برامج خفيفة لترشيح المخرجات الوسيطة، تنسيق الأدوات، الاحتفاظ بالمهم فقط، وتقليل round trips. [R01]

هذه الآلية تعالج مشكلة شائعة في agent systems: إذا أُعيد كل ناتج أداة إلى LLM كاملاً، يتضخم السياق وتزداد الكلفة وتتدهور الإشارة. بدلاً من ذلك تصبح نتيجة الأداة مادة تُعالج داخل execution runtime قبل أن تصل للنموذج. وتضيف عائلة 5.6 cache breakpoints وprompt caching أكثر قابلية للتنبؤ، وهو اتجاه مهم لأن تكلفة الوكيل الطويل غالباً يحكمها إعادة قراءة الحالة أكثر من توليد الجواب النهائي. [R01]

في التصميم الإنتاجي، لا ينبغي تشغيل max أو ultra دائماً. الأفضل router ثلاثي: مهام extraction/routing إلى نموذج رخيص، مهام reasoning محددة إلى Sol max، والمهام شديدة التعقيد ذات parallelizable subtasks فقط إلى ultra. يجب أن يربط scheduler بين قيمة التحسن المتوقعة وبين token budget وlatency SLO.

- Programmatic Tool Calling: تقليل context pollution عبر pre-processing برمجي.
- max: زيادة test-time reasoning للمهام التي تستفيد من البحث والتحقق.
- ultra: parallel subagents؛ يحتاج budget guard وmerge/verifier.
- Prompt cache: عنصر أساسي في اقتصاد long-running sessions.

> **مبدأ التشغيل** — Reasoning effort متغير قرار runtime، وليس خاصية ثابتة للنموذج.

Keywords: GPT-5.6 Sol • Programmatic Tool Calling • max • ultra • Multi-Agent Beta • Prompt Caching

07 Gemini 3.1 Pro: reasoning متعدد الوسائط والسياق الطويل

1M context, multimodal inputs and strong agent/tool benchmarks

تصف Google DeepMind Gemini 3.1 Pro بأنه نموذج reasoning متعدد الوسائط قادر على فهم النص والصورة والصوت والفيديو ومستودعات كود كاملة، مع نافذة سياق تصل إلى 1M token ومخرجات حتى 64K. [R11] هذه السعة لا تعني أن وضع مليون token في كل استدعاء ممارسة جيدة؛ قيمتها الحقيقية هي تمكين ingestion كبير عند الحاجة، بينما تظل هندسة الذاكرة والضغط مطلوبة لأن جودة الاسترجاع داخل السياق ليست ثابتة.

بطاقة النموذج تفيد بأن Gemini 3.1 Pro Thinking (High) يسجل 94.3% على GPQA Diamond، 77.1% على ARC-AGI-2، 68.5% على Terminal-Bench 2.0، 80.6% على SWE-Bench Verified، 33.5% على APEX-Agents و69.2% على MCP Atlas. [R11] تنوع هذه الاختبارات مفيد لأنه يقيس knowledge reasoning وabstract adaptation وterminal/tool execution وlong-horizon professional tasks، وليس مجرد امتحان معلومات.

هندسياً، Gemini مناسب عندما تكون multimodality أصلية في المسار: تحليل فيديو/صور مع مستندات، أو codebase مع artifacts بصرية، أو agent يعمل على Google Cloud. Google ADK وA2A يضيفان طبقة runtime مستقلة عن النموذج، لذلك يفضل عدم ربط كل business logic بواجهة Gemini نفسها. [R13][R14]

> تحذير — 1M context ليس بديلاً عن checkpoints، summaries، retrieval، أو semantic memory؛ إنه سقف سعة وليس ضمان جودة.

Keywords: *Gemini 3.1 Pro • 1M Context • Multimodal Reasoning • MCP Atlas • APEX-Agents*

08 Claude Opus 4.8 وهندسة الوكيل الطويل

Agentic judgment, harness persistence and containment

Claude Opus 4.8 هو النموذج عالي المستوى الذي أمكن تثبيته مباشرة من Anthropic في هذه المراجعة. صفحة الإطلاق تصفه بتحسن في coding وagentic skills وreasoning، وتذكر اختبارات خارجية حول computer/browser agents، مع تأكيد على تقليل خطوات tool calling وتحسين الاستمرارية عبر المسار. [R07] القيمة الهندسية هنا ليست رقماً واحداً بل نمط «judgment under tools»: أن يعرف الوكيل متى يتوقف، متى يعيد الفحص، ومتى يطلب معلومة.

أبحاث Anthropic عن long-running scientific computing توضح أن العمل متعدد الأيام يحتاج أكثر من context: test oracles، ذاكرة خارجية، تقسيم العمل، إعادة التشغيل، وتحققاً بعد كل مرحلة. [R08] وفي إرشادات harness الطويل، يتم الفصل بين initializer يهيئ environment/state وبين coding/working agent ينفذ دفعات صغيرة قابلة للتحقق؛ كما أن compaction وحده لا يحل فقدان التقدم. [R09]

أمنياً، Anthropic يميز بين sandbox وnetwork egress وexternal-content risk؛ محتوى MCP أو الويب قد يحمل prompt injection، والذاكرة المستمرة قد تتحول إلى قناة poisoning. لذلك يجب ألا تُمنح أداة حساسة للوكيل بمجرد أنها داخل sandbox؛ تحتاج policy خارجية و[R10] scoped credentials.

Keywords: *Claude Opus 4.8 • Long-Running Agent • Harness • Checkpoint • Sandbox • Egress Control*

09 DeepSeek-V4, Kimi K3: الجبهة المفتوحة

Open weights, MoE efficiency and preserved reasoning history

Kimi K3 من Moonshot AI يقدم اتجاهاً مختلفاً: نموذج open-weight native multimodal agentic بحجم 2.8T parameter، مع 104B activated parameters، 896 experts يُفعّل منها 16 لكل token، ومزيج KDA + Gated MLA، وسياق 1,048,576 token. [R15] يذكر المشروع Stable LatentMoE وتحسناً تقريبياً 2.5× في scaling efficiency مقارنة Kimi K2، مع quantization-aware training: MXFP4 weights / MXFP8 activations. هذه خصائص مهمة لبناء private/open deployments عالية السعة.

Kimi K3 يبقي thinking مفعلاً ويتيح `reasoning_effort` من low إلى max، كما يطلب preserved thinking history في الحوارات متعددة الأدوات كي يحتفظ بالاستدلال السابق. [R15] هذه الخاصية توضح إشكالية memory semantics: history ليست مجرد messages النهائية؛ أحياناً يعتمد continuation على internal reasoning fields أو tool traces، ما يفرض طبقة adapter دقيقة عند تبديل المزود.

DeepSeek-V4 مؤكد من مركز الشفافية الرسمي كإصدار بتاريخ 24 أبريل 2026 مع model card و[R16] technical report في تقرير state-of-art لا ينبغي اختزاع تفاصيل معمارية غير موجودة في المصدر المتاح؛ لذلك نستخدم V4 كخيار open-model frontier مؤكد، ونترك اختيار quantization/serving بعد قراءة model card الفعلي للنسخة المراد نشرها.

- Open weights تخفض vendor lock-in لكنها ترفع مسؤولية serving, security, evals.
- MoE يجعل total parameters مقياساً غير كافٍ؛ activated compute أهم للتكلفة.
- Preserved reasoning/tool history يحتاج compatibility layer في multi-provider router.

Keywords: *Kimi K3 • DeepSeek-V4 • MoE • KDA • Gated MLA • MXFP4 • Open Weights*

test-time compute، Max reasoning كطبقة تحكم 10

Adaptive inference budgets instead of one fixed model setting

أهم تحول في reasoning models هو أن الذكاء المتاح عند inference أصبح قابلاً للضبط. `xhigh`, `high`, `max`, أو ما يماثلها ليست مجرد presets واجهة؛ هي قرارات تخص عدد مسارات الاستكشاف والتحقق، طول التفكير، وربما parallel agents. GPT-5.6 يصرح بأن max يعطي وقتاً أكبر من xhigh، وultra يضيف عدة وكلاء بالتوازي. Kimi K3 يعرّف low/high/max. [R01][R15]

ينبغي إذاً أن يملك Agent OS «حاكم Compute Governor» compute. المدخلات ليست prompt فقط، بل تقدير صعوبة، حساسية الخطأ، إمكانية التقسيم، قيمة المهمة، والمهلة. يمكن تمثيل القرار تقريباً بـ: B* = argmax_B [P(success | task,B) × V(task) - C_tokens(B) - C_latency(B) - C_risk(B)]. هذه ليست معادلة تدريب بل policy عملية لتخصيص ميزانية الاستدلال.

الأسلوب الأكثر كفاءة هو escalation: ابدأ بإعداد متوسط؛ إذا فشل verifier أو انخفضت confidence أو ظهرت تعارضات في الأدلة، ارفع effort أو شغّل second model critic. لا تحجز frontier/max لكل subtask. بهذه الطريقة يصبح «الذكاء الأقصى» مورد burst يشبه GPU burst وليس الوضع الدائم.

> **Engineering rule —** Spend reasoning compute where uncertainty and consequence intersect.

Keywords: *Test-Time Compute • Reasoning Budget • Compute Governor • Escalation • Value of Information*

11 الاستدلال الموجّه بالعملية

Deep Interaction + Thinking Checklist Reward

تضيف أبحاث 2026 طبقة مهمة بين supervised answer scoring وRL: تقييم «كيفية الوصول» إلى الحل. Deep Interaction يسمح للمستخدم بتعديل خطوة خاطئة داخل Chain-of-Thought الأصلي ثم استئناف المسار من نقطة التصحيح، بدلاً من بدء محادثة إصلاح جديدة؛ الورقة تبلغ عن تحسن يفوق 25% في نجاح التصحيح وتقليل يقارب 40% في tokens في إعداداتها. [R28] الفكرة الهندسية الأهم هي editable execution state: يجب أن تكون الخطة، الأدلة، والقرارات intermediate قابلة للعرض والتعديل كحالة، لا كنص مخفي فقط.

Thinking Checklist Reward يحول pairwise preferences إلى checklists خاصة بالعينة لتقييم القيود والمفاضلات في التفكير، ويستخدم residual قائم على EMA لعزل «thinking surplus». [R29] هذا يفتح مساراً لبناء verifiers محليين: بدلاً من LLM judge عام يسأل «هل الجواب جيد؟»، يولّد النظام contract/checklist من spec ويقيم كل خطوة حرجة عليه.

في الإنتاج، ينبغي تطبيق هذه الأفكار على observable plan لا على كشف CoT الخاص بالنموذج. خزّن Task Graph، assumptions، evidence، acceptance tests، tool results، pointers، وdecision log. هذه البنية تحقق قابلية التدخل البشري والتدقيق دون الاعتماد على استخراج التفكير الداخلي.

Keywords: *Process Supervision • Deep Interaction • Thinking Checklist Reward • Editable State • Verifier*

12 الاستدلال غير المرئي وحدود مراقبة CoT

Latent reasoning means monitoring must be defense-in-depth

ورقة «Not All LLM Reasoning Is Visible in Chain-of-Thought» تبين أن إدخال filler tokens محايدة دلالياً قد يحسن بعض مهام الاستدلال حتى 13 نقطة مئوية، وأن جزءاً من الحساب قد يحدث في representations داخلية دون ظهوره صراحة في النص الوسيط. [R30] وSeLaR يقدم selective latent reasoning يعتمد على entropy-gated soft embeddings لتخصيص التفكير الكامن فقط عند الحاجة. [R31]

النتيجة الأمنية: لا يمكن اعتبار CoT سجل تدقيق كامل. OpenAI نفسه يناقش CoT monitoring ضمن defense-in-depth لا كضمان وحيد. [R05] إذا كان النظام يملك side effects، يجب بناء الأدلة من الأحداث الخارجية القابلة للملاحظة: browser action، SQL query، filesystem diff، API call، network egress، credential use، وverification outcome.

وبالتالي فإن «شفافية الوكيل» = traceability of actions + explicit task state + provenance، وليس «قراءة أفكار النموذج». يمكن استخدام reasoning monitor كمستشعر إضافي، لكن القرار الأمني الحاسم يجب أن يظل deterministic policy أو external authorization يمكن اختباره واسترجاعه.

> قاعدة أمان — Never equate visible CoT with complete causal explanation.

Keywords: *Latent Reasoning • Chain-of-Thought Monitoring • SeLaR • Defense-in-Depth • Provenance*

13 السياق الطويل: من نافذة ضخمة إلى هندسة سياق

Context engineering, compaction, retrieval and cache locality

نافذة 1M في Gemini 3.1 Pro وKimi K3 تجعل استيعاب repository أو corpus كبير ممكنًا، لكنها لا تلغي context rot، الكلفة، أو تشوش الانتباه. [R11][R15] في agent engineering يجب فصل أربعة مخازن: working context الحالي؛ durable task state؛ long-term semantic/episodic memory؛ وcold evidence store. لا ينبغي أن يكون transcript هو قاعدة البيانات.

تقنيات السياق المتقدمة تشمل summaries، progressive disclosure، context compaction مرتبطة بـ provenance، عند نقاط آمنة، retrieval حسب task state، prompt caching، ومبدأ «الأدلة قبل السرد»: عند إعادة استئناف المهمة، استرجع commitments والاختبارات والـ diffs أولاً، ثم history الوصفي. Anthropic يبين أن compaction وحده لا يكفي للمهام الطويلة إذا لم يوجد handoff/state صريح. [R09]

ومن الناحية الاقتصادية، locality مهمة: schema، policies، system prompt، وknowledge ثابتة يجب أن تستفيد من cache؛ أما نواتج الأدوات الضخمة فيجب تقطيرها قبل الإرجاع. GPT-5.6 يضيف explicit cache breakpoints، ما يؤكد أن context management أصبح جزءاً من runtime وليس prompt trick. [R01]

Keywords: *Context Engineering • Compaction • Prompt Cache • Semantic Memory • Episodic Memory • Evidence Store*

14 هندسة الوكلاء طويلة الأمد

Checkpoint, resume, idempotency and test oracles

الوكيل الذي يعمل لساعات أو أيام يواجه failures مختلفة عن chatbot: process restarts، tool timeout، تغير environment، duplicate side effects، stale credentials، وdrift عن الهدف. لذلك كل task طويل يحتاج durable state machine. checkpoint يجب أن يحتوي goal version، context loss، retry counters، outstanding approvals، tool receipts، artifacts hashes، completed nodes، plan graph، وnext safe action.

الـ idempotency شرط أساسي: يجب أن يحمل كل action خارجي idempotency key أو guard يمنع تكرار الدفع/الإرسال/الكتابة عند resume. وتحتاج المهام البرمجية test oracle؛ Anthropic يوضح أن long-running scientific workflows تستفيد من اختبارات يمكن تشغيلها مراراً لتحديد إن كانت المرحلة نجحت بالفعل. [R08][R09]

النمط الموصى به: micro-commit بعد كل وحدة قابلة للتحقق، summary صغير للمرحلة، artifact persistence خارج context، ثم verifier مستقل. عند crash، يبدأ النظام من آخر checkpoint مثبت لا من إعادة قراءة transcript بالكامل. هذا هو الفارق بين autonomous demo وproduction agent.

- Checkpoint != chat history.
- Resume must be deterministic enough to avoid repeated side effects.
- Verification must be executable when possible: tests, schemas, constraints, hashes.
- Long-running agents need explicit stop/abort and lease ownership.

Keywords: *Checkpoint • Resume • Idempotency • Test Oracle • Lease • Durable Execution*

15 تقييم البرمجة والوكلاء البحثيين

Beyond SWE-Bench: exploration, long-horizon and contamination-resistant evals

تقييم agentic coding لم يعد محصوراً في patch واحد. SWE-Explore يقيس codebase exploration عبر 848 قضية و10 لغات و203 repositories، مع تركيز على كفاءة العثور على السياق الصحيح. [R32] DeepSWE يقدم 113 مهمة هندسية طويلة أصلية مع verifiers مكتوبة يدوياً لتقليل contamination وقياس completion الحقيقي. [R33] هذه الاختبارات تكشف أن النجاح يعتمد على navigation + tool use + persistence بقدر ما يعتمد على code generation.

في production eval، يجب الاحتفاظ بأربعة محاور منفصلة: recovery، cost/latency، task success، وsafety. مثال لبطاقة اختبار: success@budget، عدد tool calls، نسبة side-effect violations، human interventions، retries، wall-clock، context tokens، invalid calls، وartifact quality. يمكن بناء holdout من مهام المؤسسة نفسها مع deterministic graders متى أمكن وLLM judge فقط للصفات غير القابلة للترميز.

الـ benchmark الخارجي مهم للمقارنة، لكنه لا يحل domain eval. Framework أو harness قد يغير النتيجة؛ لذلك أي قرار شراء أو routing يجب أن يمر بـ bake-off داخلي على نفس الأدوات ونفس policy ونفس budget.

> **مقياس أقوى** — Success per unit cost under a fixed safety policy أهم من score منفرد.

Keywords: SWE-Explore • DeepSWE • Long-Horizon Evals • Success@Budget • Regression Suite

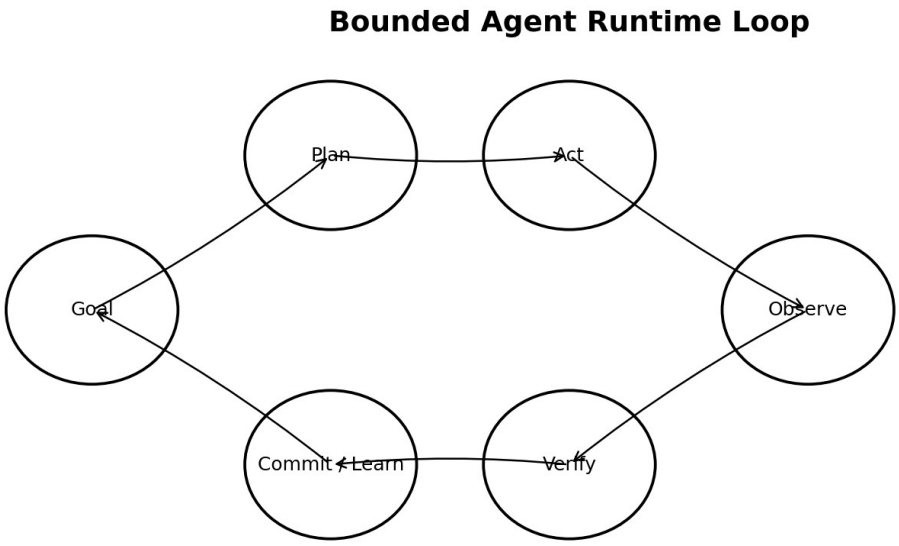
16 حلقة التشغيل الأساسية للوكيل

Bounded Goal → Plan → Act → Observe → Verify → Commit

يمكن اختزال runtime القابل للضبط إلى دورة محدودة: Goal يحدد الهدف والقيود، Plan يحوله إلى graph من subtasks، Act يختار tool/model، Observe يلتقط النتيجة الخام، Verify يختبرها ضد acceptance criteria، Commit يثبت الحالة أو يعيد التخطيط. يجب أن تكون الحلقة bounded بواسطة policy violations، cost budget، time budget، max steps، وstagnation detector.

هذا النمط يضم ReAct لكنه يضيف state contract وverification. الـ ReAct الحر مناسب لمهام صغيرة، أما المهام المؤسسة فتحتاج control flow واضحاً: ما الذي يمكن للنموذج اختياره؟ وما الذي يجب أن يبقى deterministic؟ Microsoft Agent Framework نفسه يوصي بالتمييز: agent للمهمة المفتوحة، workflow عندما تكون الخطوات معرفة مسبقاً، بل يقول: إذا أمكن كتابة function للمهمة فافعل ذلك بدلاً من agent. [R17]

أفضل بنية hybrid: deterministic shell حول stochastic core. النموذج يقترح plan/arguments أو يقرر بين بدائل، النظام يتحقق من schema/policy/capability؛ tool runner ينفذ؛ verifier يحكم؛ orchestrator فقط يقرر الانتقال.



الشكل 2 — حلقة تشغيل محدودة مع نقطة تحقق قبل تثبيت الأثر.

Keywords: ReAct • Bounded Loop • Deterministic Shell • Stochastic Core • Acceptance Criteria

17 أنماط **orchestration** الإنتاجية

Planner-Executor, Supervisor-Worker, Maker-Checker, Graph and Swarm

Planner-Executor يفصل decomposition عن التنفيذ، ويصلح عندما تكون الخطة قابلة لإعادة التقييم. Supervisor-Worker يجعل وكيل supervisor يوزع subtasks على متخصصين؛ ميزته isolation، وخطره أن supervisor يصبح bottleneck ونقطة خطأ. Maker-Checker يضع generator وverifier مستقلين، وهو قوي عندما يوجد rubric واضح أو artifact يمكن اختباره.

Graph/State Machine هو النمط الأكثر قابلية للتدقيق في الأعمال الحساسة، لأن transitions معلنة ويمكن إضافة interrupts وfallbacks. Sequential pipeline ممتاز عندما تكون المراحل ثابتة. Swarm/network مناسب عندما يكون توزيع العمل ديناميكياً، لكنه يحتاج identity، routing protocol لكل agent، shared state semantics، وحدود fan-out حتى لا يتحول parallelism إلى انفجار تكلفة.

قاعدة الاختيار: لا تستخدم multi-agent لمجرد أنه متاح. كل وكيل إضافي يزيد coordination overhead وsurface area للأخطاء. استخدمه عندما يحقق independent verification، parallelism، specialization، أو permission separation قابلة للقياس.

الخطر الرئيسي	متى يُستخدم	النمط
خطة أولية خاطئة أو قديمة	أهداف طويلة قابلة للتقسيم	Planner-Executor
اختناق وتجميع غير صحيح	تخصصات متعددة	Supervisor-Worker
تواطؤ أخطاء النموذجين	Artifacts عالية المخاطر	Maker-Checker
تعقيد تصميم graph	Workflow حساس أو قابل للتدقيق	Graph/State Machine
fan-out، تكلفة، حالة مشتركة	بحث/تنفيذ متوازٍ	Swarm/Network

Keywords: *Planner-Executor • Supervisor-Worker • Maker-Checker • Graph • Swarm*

18 الأحداث والشبكات والتوازي

Event-driven multi-agent systems require backpressure and ownership

عندما تتحول الوكلاء إلى خدمات مستقلة، يصبح event bus أكثر واقعية من تبادل الرسائل داخل prompt واحد. كل task/event يحتاج correlation ID، causation ID، schema version، deadline، identity، capability scope، وretry policy. يجب أن يملك orchestrator أو channel lease يحدد من يملك حق التقدم في الحالة.

AG2 v1.0 يتجه إلى Network مبني على Hub + typed channels مع write-ahead log وaudit trail، ويعامل channel كوحدة تنسيق بدلاً من group chat فقط. [R25] Microsoft AutoGen Core تاريخياً يقدم event-driven runtime، بينما Agent Framework الحالي يجمع workflows وsessions وmiddleware. [R17] هذه الأفكار تقرّب agent systems من distributed systems التقليدية: exactly-once illusion، dead-letter queues، backpressure، idempotency، وcircuit breakers.

التوازي لا يعني دائماً أسرع. GPT-5.6 ultra يوضح أن parallel agents يمكن أن يغيروا score-latency frontier، لكن merge cost وduplicate exploration يزيدان مع العدد. [R01] يجب ضبط fan-out ديناميكياً بناء على قابلية تقسيم المهمة، وليس رقماً ثابتاً.

Keywords: *Event-Driven • Typed Channels • Backpressure • Fan-Out • Write-Ahead Log • Circuit Breaker*

LangGraph + Microsoft Agent Framework 19

Two strong control-oriented runtimes

LangGraph يركز على long-running stateful agents عبر interrupts/HITL، persistence، durable execution، graph runtime، وstores للذاكرة عبر threads. [R18] ميزته الأساسية أن workflow state يصبح first-class ويمكن حفظه واستئنافه. هذا مناسب للعمليات التي تحتاج approvals، branches، retries، أو human intervention في مواضع محددة.

Microsoft Agent Framework هو الآن successor رسمي يجمع بساطة abstractions من AutoGen مع enterprise features من Semantic Kernel: session-based state، type safety، middleware، telemetry، ودعم graph-based workflows. الوثائق تذكر Harness Agent للمهام الطويلة مع planning/todo/context compaction/file access/memory، وWorkflows صريحة، إضافة إلى MCP clients وcontext providers. [R17]

الاختيار ليس «أي إطار أذكى»، بل integration fit. LangGraph جيد عندما تريد Python/JS graph control ونظام LangChain/LangSmith. Microsoft جيد عندما تكون .NET/Azure/Foundry، أو تحتاج middleware وenterprise type/state integration. في الحالتين يجب إبقاء domain state في schemas مستقلة حتى لا يصبح migration مستحيلاً.

المحور	LangGraph	Microsoft Agent Framework
Control flow	Graph/state-centric	Agents + graph/functional workflows
Durability	Checkpoint/persistence/interrupts	Sessions + state + long-running harness
Governance	HITL + ecosystem observability	Middleware + telemetry + enterprise integrations
Best fit	Custom state machines	Microsoft/Azure/.NET/Python estates

Keywords: *LangGraph • Microsoft Agent Framework • Durable Execution • Middleware • HITL*

OpenAI Agents SDK + Google ADK 20

Provider-native harnesses with explicit interoperability

OpenAI أعاد تطوير Agents SDK ليكون harness طويل الأمد: memory، native sandbox، قابلة للضبط، shell، AGENTS.md، skills، MCP، وapply_patch، مع فصل harness عن compute حتى يمكن snapshot/rehydration والتوسع الأمني. [R02] هذه فلسفة «thin model, rich harness»: النموذج يقرر، لكن environment والحالة والسياسات ليست داخل prompt.

Google ADK يقدم open-source toolkit لبناء agents وworkflows، مع أدوات evals، integrations، production، ودعم A2A/MCP ضمن منظومة Google. [R13][R14] عامل مهم هو cross-language A2A: يمكن أن تتعاون وكلاء مبنية بلغات أو runtimes مختلفة دون مشاركة internal memory أو code.

المعيار الأفضل هو جعل provider SDK طبقة execution adapter، لا قلب business architecture. ضع domain workflow وstate contracts في طبقة محايدة، ثم adapter لـ OpenAI/Google/Anthropic/open models. بهذه الطريقة يمكن routing بين النماذج والاستفادة من native tools دون lock-in كامل.

> **Pattern —** Provider-native SDK for capability; framework-neutral state and policy for portability.

Keywords: *OpenAI Agents SDK • Google ADK • Native Sandbox • Skills • A2A • Provider Adapter*

CrewAI + Mastra + PydanticAI 21

Role teams, TypeScript workflows and type-safe agents

CrewAI يقسم المنظومة إلى Agents/Crews/Flows؛ Flows مناسبة للأتمتة stateful والمنظمة، بينما Crews تعطي فرقاً role-based مع sequential/hierarchical patterns و[R19]. guardrails/memory/knowledge/observability هو مناسب للسرعة في تصميم collaboration، لكن يجب ربطه بstate schemas واختبارات خارجية إذا استُخدم في عمليات حاسمة.

Mastra إطار TypeScript-first يجمع model routing، observability، evals، memory، graph workflows، agents و[R20]. MCP ويقدم suspend/resume وObservational Memory التي تضغط history، ما يجعله مناسباً لتطبيقات full-stack/edge/serverless التي تريد runtime واحداً قريباً من stack JavaScript.

PydanticAI/Pydantic Graph يركز على typed structured agent definitions والقرارات/forks داخل graphs مع Pydantic validation و[R21]. Evals هذه الفلسفة مهمة: كل final artifact، state transition، tool arguments يجب أن يمر عبر type safety. schema لا تمنع hallucination، لكنها تمنع فئة كاملة من integration failures.

- CrewAI: collaboration-first.
- Mastra: TypeScript full-stack + memory/evals/MCP.
- PydanticAI: typed contracts + validation + graph/evals.

Keywords: *CrewAI • Mastra • PydanticAI • Type Safety • Structured Output • Flows*

LlamaIndex + DSPy + smolagents + AG2 22

Specialized frameworks that fill distinct engineering niches

LlamaIndex يظل قوياً عندما يكون قلب النظام knowledge/RAG؛ pattern «agents as tools» يسمح لوكيل رئيسي بتفويض البحث أو التقرير إلى subagents مع نماذج مختلفة. [R22] DSPy يغير مستوى البرمجة: signatures/modules/optimizers بدلاً من hand-written prompts، ويستطيع تحسين instructions/few-shot أو weights وفق metric؛ في 2026 توجد Optimizers مثل MIPROv2 و[R23] .GEPA

smolagents من Hugging Face يوفر CodeAgent الذي يكتب أفعال الأدوات ككود وToolCallingAgent التقليدي، مع sandbox backends وMCP ودعم متعدد المزودين. [R24] فائدته خفة abstraction وتجريب agents قابلة للقراءة. AG2 v1.0 يقدم AgentOS مع Network/Hub/typed channels وknowledge/context compaction وHITL وtelemetry/evaluation، بعد فصل AG2 Classic القديم. [R25]

هذه الأطر لا يجب وضعها في leaderboard واحد؛ هي تحل مشاكل مختلفة: minimal code-agent، optimization-centric، retrieval-centric، وnetwork-centric. التصنيف الوظيفي أفضل من عبارة «أفضل framework».

متى أختاره	المجال الأقوى	الإطار
المصدرية والاسترجاع قلب النظام	RAG / knowledge agents	LlamaIndex
لديك metric وeval set وتريد compile/optimize	Prompt/program optimization	DSPy
تجارب خفيفة وعمل مباشر tool/code	Minimal CodeAgent	smolagents
قنوات typed وحالة/معرفة طويلة	Networked multi-agent	AG2

Keywords: *LlamaIndex • DSPy • MIPROv2 • GEPA • smolagents • AG2*

23 MCP 2026-07-28: طبقة الأدوات القياسية

Stateless core, MRTR, routing, authorization hardening and extensions

إصدار Model Context Protocol بتاريخ 2026-07-28 هو أحد أهم التحديثات الهندسية في corpus. المواصفة تنتقل إلى stateless request/response لتحسين core reliability وhorizontal scaling، وتضيف Multi Round-Trip Requests للتفاعلات التي تحتاج أكثر من request، وheader-based routing، وcacheable list results، وauthorization hardening، وextensions framework رسمي، مع SDKs محدثة. [R26]

مفهومياً، MCP لا يحول الأدوات إلى «وكلاء». هو contract بين client/server لتقديم primitives: Tools model-controlled، Resources application-controlled، Prompts user-controlled. هذا الفصل مهم للسياسات. tool schema يجب أن يصف input/output والآثار؛ resource يجب ألا يحصل تلقائياً على صلاحيات كتابة؛ والـ prompt template ليس executable capability.

في enterprise design، ضع MCP Gateway أمام schema enforcement، secret brokering، audit، rate limits، allowlist، tenant routing، identity servers. stateless core يسمح load balancing أفضل، لكنه لا يعني أن business operation نفسها stateless؛ state الدائم يجب أن يعيش في store/service واضح.

- MCP ≠ memory layer.
- MCP ≠ agent-to-agent semantics؛ A2A يغطي التعاون بين agentic applications.
- Authorization يجب أن يُربط بالـ resource/audience وبهوية المستخدم والوكيل.
- لا تمرر credentials طويلة العمر داخل model context.

Keywords: *MCP 2026-07-28 • Stateless • MRTR • Tool Discovery • Authorization • MCP Gateway*

24 A2A وبنية بروتوكولات الوكلاء

Inter-agent interoperability without sharing internal state

A2A هو بروتوكول مفتوح لتواصل agentic applications، يركز على negotiation of modalities، discovery of capabilities، وإدارة long-running tasks دون كشف internal state للوكيل. [R27] لذلك هو مكمل لـ MCP: MCP يربط agent بأداة/بيانات، بينما A2A يربط agent/application ب agent/application. Google يضع أيضاً UCP/AP2/A2UI/AG-UI ضمن خريطة أوسع للبروتوكولات حسب نطاق التفاعل. [R13]

للاستخدام الإنتاجي يجب أن تحتوي Agent Card أو capability metadata على task semantics، supported modalities، auth requirements، version، وSLOs. الرسالة بين الوكلاء ليست ثقة تلقائية: كل agent principal يحتاج provenance، authorization، identity، وسجل. إذا استدعى Agent A Agent B الذي يملك صلاحيات أكبر، يجب منع privilege laundering عبر delegation.

البنية المقترحة: A2A Gateway يطبق identity and policy؛ MCP Gateway يطبق tool policy؛ event bus يدير asynchronous state؛ وorchestrator يربطهما عبر correlation IDs. هذا يجعل interoperability قابلة للتبديل دون تحويل البروتوكول نفسه إلى control plane.

> تمييز حاسم — MCP = tool/data interoperability. A2A = agent-to-agent task interoperability. Neither replaces authorization policy.

Keywords: *A2A • Agent Card • Capability Discovery • Delegation • Protocol Mesh*

25 الأدوات والمهارات و**Agentic RAG**

Skills, tools and retrieval must be scoped, typed and evidence-bearing

الوكيل الفعال لا يحتاج آلاف الأدوات مكشوفة في كل turn. يجب استخدام progressive tool disclosure: router يختار مجموعة صغيرة حسب task state؛ تكون schemas typed؛ تتضمن descriptions preconditions/side effects؛ وكل tool call ينتج receipt/provenance. Google ADK skills وOpenAI Agents SDK skills يرسخان فكرة packaging capability كمهارة يمكن تحميلها عند الحاجة. [R02][R14]

Agentic RAG يختلف عن RAG التقليدي لأنه حلقة بحث: formulate query → retrieve → inspect → identify gap → re-query → cite → verify. LlamaIndex وغيره يسهل agents-as-tools، بينما BI/data agents تستخدم semantic layer بدلاً من vector search فقط. [R03][R22] يجب أن يفصل النظام factual evidence عن generated notes، وأن يحتفظ بمؤشر إلى المصدر/الإصدار/السطر أو query result.

الأمن: كل مصدر خارجي untrusted content. صفحات الويب، GitHub issues، PDFs، حتى MCP resources قد تحمل تعليمات هجومية. sanitize ليس كافياً؛ افصل data plane عن instruction plane، واسمح فقط للأوامر التي تأتي من policy/task state الموثوق. [R04][R10]

Keywords: Skills • Tool Registry • Progressive Disclosure • Agentic RAG • Provenance • Instruction/Data Separation

26 الذاكرة والحالة: تصميم متعدد الطبقات

Working, task, episodic, semantic and artifact memory

الذاكرة الوكيلية تحتاج taxonomy. Working memory هي ما يجب أن يراه النموذج الآن. Task state هو الحقائق التنفيذية: plan, status, retries, approvals. Episodic memory يلخص تجارب سابقة مرتبطة بسياق. Semantic memory يحمل knowledge ثابتة نسبياً. Artifact store يحمل الملفات والـ diffs والنتائج الكبيرة. مزج هذه الطبقات في vector DB واحد يؤدي إلى استرجاع غير متوقع وتسرب صلاحيات.

كتابة الذاكرة يجب أن تمر عبر write policy: ما الذي يستحق الاستمرار؟ ما TTL؟ ما مستوى الثقة؟ ما مصدره؟ هل يحتوي PII/secret؟ يجب تطبيق deduplication وversioning وforget/rollback. لا تسمح للنموذج بأن يحول أي نص وارد إلى «حقيقة طويلة الأجل» دون verifier، لأن memory poisoning خطر واضح في الوكلاء طويلة العمر. [R10]

عند الاسترجاع، استخدم scoped retrieval مرتبطاً بالـ tenant/user/task والقدرات. أفضل memory هي التي تقلل tokens وتزيد consistency دون أن تخفي provenance. Mastra observational memory وMicrosoft context providers وLangGraph stores أمثلة هندسية مختلفة لنفس المشكلة. [R17] [R18] [R20]

طبقة الذاكرة	المحتوى	سياسة العمر
Working	المعلومات اللازمة للخطوة الحالية	قصيرة جداً / context
Task state	الخطة والحالة والاعتمادات	حتى إغلاق المهمة + سجل
Episodic	ملخصات خبرة موثقة	TTL/quality gated
Semantic	حقائق/سياسات/تعريفات	versioned source-controlled
Artifacts	ملفات، raw evidence، diffs	object store + hashes

Keywords: *Working Memory • Episodic Memory • Semantic Memory • Artifact Store • Memory Poisoning*

Multi-model routing واقتصاد الاستدلال 27

Route by capability, uncertainty, privacy, latency and cost

حالة الفن العملية هي mixture-of-models على مستوى النظام، لا نموذج واحد. router يقرر بين frontier cloud model، نموذج متوسط، open-weight local model، أو deterministic code. المدخلات: latency، context size، privacy boundary، tool competence، required modalities، task class، token cost، SLO، وhistorical eval scores. يجب أن يكون routing مبنياً على data من traces لا على الانطباع.

يمكن استخدام two-stage router: cheap classifier يحدد candidate set، ثم policy يحسب expected utility. إذا تعارضت نتائج verifier، models، أو escalation يحسم. للمهام الحساسة يمكن فصل planner عن executor: frontier model يخطط دون secrets، local/private model يتعامل مع البيانات، deterministic tool runner ينفذ. هذا يخلق privacy-preserving orchestration.

وفي الأطر المفتوحة، Kimi K3/DeepSeek-V4 يتيحان مسارات self-hosted، لكن 2.8T MoE ليس «نموذج laptop»؛ يحتاج serving stack مناسب. [R15][R16] لذا يجب التمييز بين open weights وlow-resource deployability.

Router objective — Maximize verified task utility, not model benchmark prestige.

Keywords: *Model Routing • Cost-Aware • Privacy-Aware • Escalation • Self-Hosted • Expected Utility*

28 المراقبة والتقييم المستمر

Trace-level observability turns agent behavior into an engineering object

كل run يجب أن ينتج trace قابل لإعادة البناء: input hash، model/version/effort، prompt policy version، retrieved evidence IDs، tool calls وarguments، tool results hashes، transitions، approvals، cost، latency، errors، final verifier. لا يكفي logging نص المحادثة. الهدف أن تستطيع الإجابة: لماذا وصل النظام إلى هذه side effect؟ وهل يمكن replay بدون تنفيذ الأثر؟

التقييم الإنتاجي ثلاثي: offline regression قبل النشر؛ shadow/canary online؛ وpost-run scoring. استخدم deterministic evaluators لـ schema/tests/constraints، وLLM-as-judge للصفات اللغوية مع calibration ومقارنة متعددة. Pydantic Evals وMastra evals وMicrosoft telemetry/LangSmith تعطي أدوات، لكن taxonomy للمقاييس يجب أن تكون ملك النظام لا الأداة. [R17][R18][R20][R21]

أهم مقاييس الوكيل: Task Success Rate، Tool Correctness، Recovery Rate، Mean Steps vs optimal، Escalation Rate، Cost per verified success، Human minutes saved، Stale-state errors، Safety Violations، وArtifact acceptance. يجب تقطيع النتائج حسب task type والنموذج والـ harness حتى تُرى regressions.

Keywords: Tracing • Evals • LLM-as-a-Judge • Canary • Replay • Cost per Verified Success

29 العزل الأمني و**Prompt Injection**

Sandbox is necessary but not sufficient

Prompt injection في agent system ليس مشكلة نص فقط؛ هو تحويل بيانات خارجية إلى أوامر تحاول تغيير plan أو سرقة secrets أو تشغيل tools. كلاهما يركزان على الدفاع متعدد الطبقات. [R04][R10] الخط الأول: instruction hierarchy وcontent boundaries. الخط الثاني: tools بحد أدنى من الصلاحيات. الخط الثالث: sandbox filesystem/process. الخط الرابع: network egress allowlist. الخط الخامس: human approval أو policy gate للأثر الحساس.

يجب فصل credentials عن model context. يحصل tool runner على short-lived scoped token بعد authorization، وليس الوكيل على secret عام. browser/download content يمر كـ untrusted evidence. أي محاولة من المحتوى لتغيير system policy تعامل كبيانات لا كتعليمات. يمكن إضافة classifier، لكنه لا يستبدل capability security.

وكل code execution يحتاج filesystem mount، resource limits محدود، secret redaction، package allowlist، network policy، وartifact scanning. مثلاً يدعم smolagents sandbox backends؛ يقدم OpenAI SDK native sandbox؛ لكن جودة sandbox تعتمد على configuration وليس اسم framework. [R02][R24]

- Least privilege by default.
- Short-lived credentials outside the prompt.
- Read and write permissions separate.
- Network egress explicit.
- Consequential actions require policy or human approval.

Keywords: Prompt Injection • Sandbox • Egress • Least Privilege • Capability Security • Secret Brokering

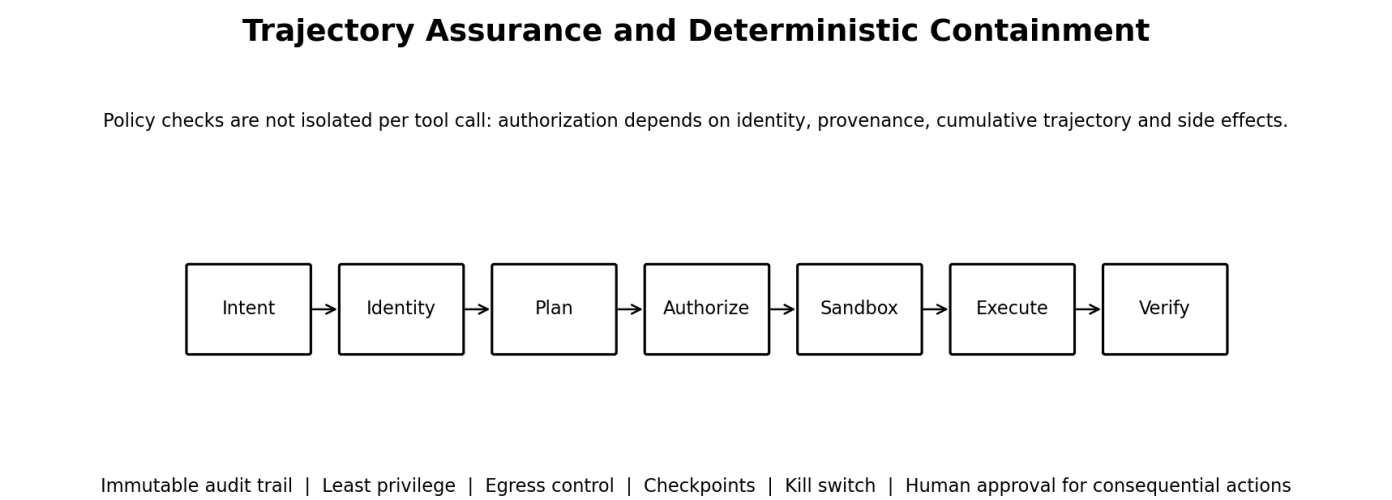
Trajectory Assurance وهوية الوكيل 30

Secure the sequence of actions, not only each action

ورقة Trajectory Assurance تشير إلى أن سلسلة من أفعال تبدو كل واحدة منها مسموحة قد تنتهك invariant على مستوى النظام؛ لذلك التحقق لكل tool call منفرداً غير كافٍ. [R35] يجب أن تنظر policy إلى cumulative trajectory: من طلب ماذا؟ بأي هوية؟ ما البيانات التي قُرئت؟ ما الصلاحيات التي اكتسبت؟ ما الأثر النهائي؟

NIST في 2026 يركز صراحة على identity/authorization/auditing/non-repudiation للوكلاء، وعلى منع prompt injection. [R36][R37] يجب تعريف agent principal مستقل عن model name: `agent_instance_id`, owner, tenant, delegated user, capabilities, token audience, session lease. إذا فوّض agent مهمة لوكيل آخر، تنتقل صلاحيات محدودة لا هوية مطلقة.

الـ kill switch يجب أن يكون خارج runtime الذي قد يفشل. كذلك pre-execution authorization للأفعال عالية العواقب يجب أن ينفذه policy service مستقل. بهذا تصبح الثقة خاصية معمارية قابلة للإثبات جزئياً، لا وعداً بأن النموذج «سيتصرف جيداً».



الشكل 3 — نقل الأمن من فحص *action* منفرد إلى ضمان مسار *execution* كامل.

Keywords: *Trajectory Assurance • Agent Identity • Authorization • Non-Repudiation • Delegation • Kill Switch*

31 الأنظمة والتسريع: CPU-GPU boundary

Agentic workloads are heterogeneous distributed systems

دراسة «Architectural Implications of Agentic AI Workflows» على بيئات Azure إنتاجية وتجارب مضبوطة تبين أن agent workflows تعبر مراراً بين CPU-side orchestration/tool processing وGPU inference، ما يجعل host runtime جزءاً من critical path. [R34] الحمل bursty ومجزأ: عدة استدعاءات قصيرة للنموذج تتخللها network wait، serialization، retrieval، code، browser، parsing.

هذا يغير optimization priorities: لا يكفي رفع tokens/s. تحتاج tool locality، CPU role pooling، prefetch، connection pooling، async I/O، batching حين لا يضر latency، وseparation بين latency-critical control وthroughput background jobs. كما يجب جدولة model calls والأدوات على resources مختلفة؛ بعض الأدوات GPU-heavy وبعضها CPU/network-heavy.

على مستوى serving، context cache، وKV cache locality مهمان للوكيل الطويل. open models مثل Kimi K3 تقترح vLLM/SGLang وغيرها، لكن النظام يجب أن يقيس end-to-end task latency لا raw inference فقط. [R15]

> **Performance rule —** Optimize the full agent trace: inference + tools + orchestration + serialization + waits.

Keywords: *CPU-GPU Boundary • Agent Serving • Async I/O • KV Cache • Prefetch • End-to-End Latency*

Agentic BI: من dashboard إلى closed-loop decision system 32

Semantic layer + analytics agents + governed actions

Agentic BI لا يعني إضافة chatbot أمام warehouse. البنية الصحيحة: semantic layer تعرف metrics والعلاقات والصلاحيات؛ analytics agent يفسر الهدف ويولد query؛ execution engine يستعلم؛ verifier يراجع النتائج والمنطق؛ narrative layer تشرح؛ action agent يقترح أو ينفذ remediation ضمن policy. Google Looker يصف صراحة الانتقال إلى BI Agents وagentic workflows فوق semantic model. [R39][R40]

الـ semantic layer حاسمة لأن text-to-SQL الخام قد يكتب query صحيحة نحوياً لكنها خاطئة أعمالياً. metric مثل «revenue» قد يختلف حسب booked vs recognized أو timezone أو exclusions. يجب أن يحصل agent على governed metric definitions وlineage بدلاً من تخمين schema.

Closed-loop لا يعني autonomy مطلقة. monitoring/anomaly detection قد يعمل تلقائياً؛ root-cause analysis يكوّن evidence graph؛ action proposal يمر عبر risk tier. إرسال تنبيه منخفض المخاطر آلي، تغيير سعر أو دفع مالي يحتاج approval.

Keywords: *Agentic BI • Semantic Layer • Root-Cause Analysis • Closed Loop • Governed Metrics • Decision Intelligence*

33 منصات BI الحالية: Looker, Power BI, Databricks, Snowflake

Different stacks converge on governed agents over enterprise data

Looker يربط Conversational Analytics وBI Agents بـ LookML semantic layer، ويضيف Agentic Workflows لمراقبة البيانات وتحليل الأسباب الجذرية. [R39][R40] Microsoft Power BI يوثق agentic capabilities ومنافذ MCP/agent skills حول semantic models، ويؤكد أن Copilot يحتاج model محضراً جيداً حتى لا تكون الإجابات مضللة. [R41]

Databricks AI/BI يطور Genie نحو agent mode متعدد الخطوات مع reasoning وتقارير/citations/visualizations، مناسب لبيئة lakehouse. [R42] Snowflake Cortex Agents يقدم managed agentic layer يجمع Cortex Analyst للبيانات المنظمة وCortex Search لغير المنظمة، مع custom tools/MCP/thread state، وRBAC ومراقبة وتقييم. [R43][R44]

التقارب واضح: data platform + semantic/governance layer + agent planner + retrieval/search + tool/action layer + audit. لذلك لا ينبغي تصميم Agentic BI كمنتج منفصل عن data governance؛ هو سطح تشغيل جديد فوقها.

المنصة	المرساة	قيمة Agentic
Looker	LookML semantic layer	BI agents + monitoring workflows
Power BI	Semantic models	Agentic developer capabilities / MCP
Databricks	Lakehouse + Genie	Multi-step analysis and artifacts
Snowflake	Cortex Analyst/Search	Managed agents + RBAC + custom tools

Keywords: Looker • Power BI • Databricks Genie • Snowflake Cortex Agents • Semantic Model

34 الطبقة الدلالية وعمليات القرار

Metrics contracts are the API between business truth and agents

في المؤسسة، semantic model يجب أن يُعامل كـ contract. كل metric لها definition, dimensions, grain, filters, freshness, owner, lineage, allowed audiences. عندما يسأل agent «لماذا انخفض الهامش؟» يجب أن يبني analysis plan على هذه العقود، لا أن يخمن joins. هذه الطبقة تجعل الإجابة قابلة للتكرار عبر human dashboard وagent.

التحليل المتقدم يمر بمراحل: detection → attribution → hypothesis → evidence → counterfactual/check → recommendation → action. ينبغي تخزين كل مرحلة كـartifact مع queries والنتائج، بحيث يمكن مراجعته. إذا تغير data snapshot، يعرف النظام أي conclusions قد تصبح stale.

المسار الأكثر قيمة اقتصادياً ليس «اسأل بياناتك»، بل continuous intelligence: مراقبة KPI، اكتشاف deviation، تشغيل diagnostic subgraph، إرسال briefing، ثم تنفيذ bounded remediation. لكن كل action needs policy tier. هذا يربط BI مباشرة بهندسة agentic security السابقة.

BI engineering principle — A governed semantic layer is more important than a clever text-to-SQL prompt.

Keywords: *Metrics Contract • Lineage • Freshness • Counterfactual • Continuous Intelligence*

35 مسارات **AGI** من منظور هندسي

Scaling, paradigm shifts, recursive improvement and multi-agent collectives

بحث «From AGI to ASI» Google DeepMind يعرض أربعة مسارات عامة: توسيع AGI، تحولات paradigm، recursive improvement، وlarge-scale multi-agent collectives. [R12] لا تعني هذه المسارات أن ASI تحقق؛ هي إطار لتحليل كيف قد تتزايد القدرة. بالنسبة للمهندس اليوم، أكثرها ملموساً هو recursive R&D: النماذج تكتب code، تشغل evals، تحلل failures، وتقترح تحسينات ضمن loop تحقق.

لكن AGI تتطلب أكثر من نجاح benchmark. التحديات تشمل continual learning دون catastrophic forgetting أو causal/world modeling، robust out-of-distribution adaptation، goal stability، long-term memory with provenance، self-model/uncertainty calibration، وembodied intelligence. resource-aware planning يضيف perception/action latency وphysical safety.

يمكن بناء «AGI-oriented architecture» الآن: research loops، self-evaluation، model routing، tools، memory، modular cognition. ولكن يجب وصفها كـ architecture hypothesis، لا كإثبات AGI. الفصل العلمي بين engineering aspiration وvalidated capability ضروري.

Keywords: AGI • ASI • Recursive Improvement • World Models • Continual Learning • Multi-Agent Collectives

36 ما لم يُحل بعد

Failure modes that remain frontier research problems

أكبر الفجوات ليست fluency. أولاً، long-horizon reliability: احتمال خطأ صغير في كل خطوة يتراكم. ثانياً، robust planning تحت عدم اليقين وتغير العالم. ثالثاً، memory write correctness ومنع poisoning. رابعاً، evaluation للمهام المفتوحة حيث لا يوجد ground truth. خامساً، security عندما يملك الوكيل أذونات حقيقية.

سادساً، hidden/latent reasoning يجعل interpretability غير كاملة. سابعاً، multi-agent coordination قد يضاعف الأخطاء والتكلفة. ثامناً، continual learning الآمن في الإنتاج لا يزال صعباً؛ غالبية الأنظمة تستخدم external memory لا تحديثاً مستمراً للأوزان. تاسعاً، benchmark contamination وharness sensitivity يحدان من معنى الأرقام. عاشراً، governance/identity standards ما زالت تتطور. [R30][R35][R36][R37]

لهذا يجب مقاومة «AGI by integration»؛ جمع 200 skills و40 agents وذاكرة ضخمة قد يزيد coverage لكنه لا يضمن generalization أو coherence. الكثافة المعرفية ليست بديلاً عن falsifiable evals. كل قدرة جديدة تحتاج task suite وحدود failure واضحة.

- Long-horizon compounding error •
- Continual learning safety •
- Causal/world-model robustness •
- Evaluation of open-ended work •
- Agent identity/delegation •
- Memory poisoning •
- Benchmark/harness sensitivity •

Keywords: *Open Problems • Reliability • Continual Learning • Interpretability • Benchmark Contamination*

37 مرجع معماري إنتاجي 2026

A model-agnostic, protocol-native, verification-first blueprint

البنية المقترحة تتكون من API/UX → Intent & Risk Classifier → Task Planner → Policy/Identity → Orchestrator → Model Router → Tool/Agent Gateways → Sandboxed Executors → Evidence/State Stores → Verifiers → Commit/Action. كل transition يحمل trace context. MCP Gateway يربط tools/resources؛ A2A Gateway يربط agentic services؛ event bus للمهام الطويلة؛ state store مستقل عن framework.

يعمل planner بـprivileges صفر تقريباً. authorization service يصدر capability token قصير العمر لكل action بعد مراجعة task/risk/trajectory. executor لا يقرأ secrets إلا ما يلزم. verifier يستخدم deterministic checks أولاً ثم model critic. commit service هو الوحيد القادر على تثبيت side effect عالية الحساسية، وربما بعد human approval.

لـ R&D، أضف eval factory وenvironments + verifiers مثال على بناء simulation environment. Turing RL Environments للتدريب/التقييم. [R45] للمؤسسة، أنشئ digital twin أو sandboxed staging ليتعلم agent workflow دون لمس production.

الطبقة	مسؤوليتها	**Invariant**
Planner	تفكيك الهدف	لا side effects
Policy/Identity	صلاحيات وdelegation	least privilege
Orchestrator	state/graph/retry	bounded execution
Model Router	اختيار model/effort	budget + capability
Gateway/Executor	tools/A2A/MCP	sandbox + scoped token
Verifier/Commit	اختبار ثم تثبيت	no commit without evidence

Keywords: *Reference Architecture • Policy Service • MCP Gateway • A2A Gateway • Verifier • Commit Service*

38 مصفوفة اختيار هندسية وخارطة تنفيذ

Pragmatic stack selection for advanced agentic systems

لا يوجد stack واحد لكل مشروع. لمسار software engineering طويل: model قوي في terminal + LangGraph/Microsoft/OpenAI harness + repo index + sandbox + tests. لمسار BI: semantic layer + data agent + query verifier + bounded actions. لمسار research: multimodal/long-context model + retrieval + code/simulation tools + experiment ledger + independent critic. لمسار high-security: deterministic workflow أكبر، أصغر في القرار، approvals أكثر.

خارطة التنفيذ من أربع مراحل: (1) Instrument: trace كل شيء وابنِ eval set من المهام الحقيقية. (2) Bound: حدد schemas/policies/side-effect tiers/checkpoints. (3) Optimize: routing, cache, context, parallelism فقط عندما يثبت eval أن التخصص أو التوازي يرفع النتائج. (4) Expand بناءً على cost-per-success: أضف agents/tools

الهدف المعماري ليس «أقصى استقلالية»، بل أقصى verified autonomy ضمن budget ومجال ثقة محدد. عندما ينخفض اليقين أو ترتفع العواقب، النظام يصعد إلى verifier/human، لا يستمر لأنه ما زال يملك tokens.

الحالة	Stack أولي موصى به	ملاحظة
Long-horizon coding	GPT/Claude/Gemini/Kimi + graph harness + sandbox	tests/checkpoints إلزامية
Enterprise BI	Semantic layer + agent + governed tools	لا raw SQL autonomy
Research agent	Long-context + code/sim + evidence ledger	independent verification
High-consequence ops	Workflow-first + scoped agent decisions	HITL/authorization gates

Keywords: *Stack Selection • Verified Autonomy • Cost per Success • Risk Tier • Implementation Roadmap*

39 سجل مصادر **Google Drive** الداخلية

August-11 *reports merged*; *August*-12 *report intentionally excluded* 22

تم تجميع النسخ المتكررة/المتباينة عنواناً ومحتوى لتقارير "AGI, Agentic AI Frameworks & Business Intelligence Trends Report" بتاريخ 11 أغسطس 2026. لم تُحذف الملفات الأصلية ولم يُعتبر التكرار تصويتاً مستقلاً على صحة الادعاء؛ بل استُخدم لاستخراج themes والتقاط التقنيات المذكورة ثم التحقق الخارجي. التقرير المؤرخ 12 أغسطس ظهر في نفس البحث الزمني لكنه استُبعد من corpus المطلوب حتى تبقى حدود المهمة واضحة.

المعرفات أدناه هي manifest مصدر داخلي D01-D22. تم دمج موضوعاتها في الفصول السابقة: reasoning models، long-horizon execution، evals، security، BI، memory/state، MCP/A2A، frameworks، وsystem architecture. عند تعارض نسختين، رُجّح المصدر الأولي الحالي وليس أحدث صياغة داخلية.

Internal source A	Internal source B
D01: 1l832pPRKVybj3y-thS6KR2kPSxtaa22qFYlb4mS0e7s	D02: 1bwepp5Fl97VjnDAbbrQcOcKUkrVaBGdgwb7UFP11UE4
D03: 1BY62Dj4GDoaoKFl9-Na7t_kEt-jr3sy0n0O5zBDifo4	D04: 1aozITfPLQUzh2cDXx046R45HWiC0TQ5xLYRLVI35FGY
D05: 1Qeprnkq8tD-ZOlsYOfH3L_b5vc6bfQsGdrtXMIp7TAs	D06: 15_6nSWzvFDe3qu1UJQC66dmxngg9hSPoH6fXNHG-NNE
D07: 1JiQro7_9lRx9MHg19GSA_C6jbWoXw99FSMi3Bkxvd4E	D08: 1NOukR62bwR_qtxCxk4W0ZPBrVCK6B84whMllG1KR23Y
D09: 1UdpOhXrduVgAlRJStMs8j8-nF44FovaIOPVPKQBrtn8	D10: 1fAyVOc-Lv7tgP5rkWHIv0_-E7LPOLTPzSx53nes0GyQ
D11: 155ny0Xe4lzKQsCo2Ebo-_6QprLZTOwHLgFcBKd-3CtM	D12: 1UF3smMg_DKK2KxFTum44XYImoQpMrEDdWs-knQ6oai0
D13: 1m1ERjMQMTJ6VcqshBOvrePeeT19mDqLe5g-rfV6elv0	D14: 1FMZmOzdci0-cMa3ZCX3pfg5GawaMPVYGx8-X6kI9_Vc
D15: 1VharSexjUlEi44X5EOqZ6ZiNWdtBwEAKRznMFB3jGcU	D16: 1C1dCCL6Fo_We2erbW_c14KVnOHYx0OrENh70DABYl90
D17: 1W1L-G9y6pzhs2yH2WEgNYU47C2hmC4LcAIfcahp8T5Q	D18: 141_DlAWQccY0PcIttDn8ArCM768gMicAhJktOOUsA8I
D19: 16ufLEBXShD19REiuMBpQTKgl1b8tke9hRWotgkaul9s	D20: 1Kn0NXOthvoKiASh113hk2une3Ty2XwQdfNglc-sTQX0
D21: 1P99w19izqEaYzb41uKrvj3mvE5xOAybPmN0VBHphpTE	D22: 1iJz4Nj99jsYuXikQvfzjwipMn7onWqtywJ_tN8x0u7w

Keywords: *Google Drive Corpus* • *D01-D22* • *Scientific Merge* • *Deduplication*

40 المراجع الأولية والعلمية

Primary sources used to verify and extend the Drive corpus

ترتيب المراجع أدناه مطابق للرموز [Rxx] المستخدمة في المتن. أولوية المصادر: وثائق ومختبرات أصلية، مواصفات بروتوكولات رسمية، أوراق علمية، ثم وثائق منصات البيانات. تاريخ الوصول: 16 أغسطس 2026.

OpenAI. GPT-5.6: Frontier intelligence that scales with your ambition. 2026-07-09. Source [R01]

OpenAI. The next evolution of the Agents SDK. 2026-04-15. Source [R02]

OpenAI. Inside our in-house data agent. 2026-01-29. Source [R03]

OpenAI. Designing AI agents to resist prompt injection. 2026-03-11. Source [R04]

OpenAI. Reasoning models and chain-of-thought controllability. 2026-03-05. Source [R05]

OpenAI. GPT-5.6 System Card. 2026-07-09. Source [R06]

Anthropic. Introducing Claude Opus 4.8. 2026-05-28. Source [R07]

Anthropic. Long-running Claude for scientific computing. 2026-03-23. Source [R08]

Anthropic. Effective harnesses for long-running agents. 2025-11-26. Source [R09]

Anthropic. How we contain Claude across products. 2026-05-25. Source [R10]

Google DeepMind. Gemini 3.1 Pro Model Card. 2026-02-19. Source [R11]

Google DeepMind. From AGI to ASI. 2026-06-12. Source [R12]

Google Developers. Developer's Guide to AI Agent Protocols. 2026-03-18. Source [R13]

Google Developers. ADK integrations ecosystem. 2026-02-27. Source [R14]

Moonshot AI. Kimi K3 - Open Frontier Intelligence. 2026. Source [R15]

DeepSeek. Transparency Center - DeepSeek-V4. 2026-04-24. Source [R16]

Microsoft. Microsoft Agent Framework Overview. 2026. Source [R17]

LangChain. LangGraph documentation. 2026. Source [R18]

CrewAI. CrewAI documentation. 2026. Source [R19]

Mastra. Mastra AI Agent Framework. 2026. Source [R20]

Pydantic. Pydantic AI / Pydantic Graph documentation. 2026. Source [R21]

LlamaIndex. Agents as tools / multi-agent documentation. 2026. Source [R22]

Stanford NLP. DSPy - Programming, not prompting, foundation models. 2026. Source [R23]

Hugging Face. smolagents documentation. 2026. Source [R24]

AG2. AG2 Open-Source AgentOS. 2026. Source [R25]

Model Context Protocol. The 2026-07-28 Specification. 2026-07-28. Source [R26]

A2A Project. Agent2Agent Protocol. 2026. Source [R27]

Turing / arXiv. Deep Interaction: CoT editing framework. 2026-07-15. Source [R28]

arXiv. Thinking Checklist Reward. 2026-07-22. Source [R29]

arXiv. Not All LLM Reasoning Is Visible in Chain-of-Thought. 2026-07-24. Source [R30]

ACL 2026. SeLaR: Selective Latent Reasoning. 2026. Source [R31]

arXiv. SWE-Explore. 2026-06. Source [R32]

arXiv. DeepSWE. 2026-07. Source [R33]

arXiv. Architectural Implications of Agentic AI Workflows. 2026-08-05. Source [R34]

arXiv. Securing Agentic AI: From Per-Action Checks to Trajectory Assurance. 2026-08-03. Source [R35]

NIST NCCoE. Software and AI Agent Identity and Authorization - Concept Paper. 2026-02-05. Source [R36]

NIST. AI Agent Standards Initiative. 2026-02-17. Source [R37]

European Commission. AI Act Service Desk - FAQ / enforcement timeline. 2026. Source [R38]

Google Cloud. Looker updates for the agentic BI era. 2026-04-22. Source [R39]

Google Cloud. Looker Agentic Workflows for monitoring and insights. 2026-07-29. Source [R40]

Microsoft. Power BI developer: Agentic capabilities. 2026. Source [R41]

Databricks. AI/BI 2026 release notes. 2026. Source [R42]

Snowflake. Cortex Agents documentation. 2026. Source [R43]

Snowflake. Cortex Agents access control. 2026. Source [R44]

Turing. RL Environments. 2026. Source [R45]

— معجم مصطلحات وخلاصة تنفيذية

Arabic definitions with canonical English keywords

English keyword	التعريف العربي المختصر
Agentic AI	أنظمة تستقبل هدفاً وتخطط وتستعمل أدوات وتحدّث حالة وتنفذ أفعالاً ضمن حدود.
Harness	طبقة runtime حول النموذج: tools, state, context, sandbox, policies, retries, tracing.
Reasoning Effort	ميزانية test-time compute المخصصة للبحث والتحقق قبل الإخراج.
MCP	Model Context Protocol لربط الوكلاء بالأدوات والموارد والقوالب.
A2A	Agent2Agent protocol لتبادل المهام والقدرات بين تطبيقات وكيلية مستقلة.
Durable Execution	حفظ الحالة/checkpoints بحيث يمكن استئناف المهمة بعد الانقطاع.
Trajectory Assurance	تحقق أمني على تسلسل الأفعال والهوية والمصدر والأثر التراكمي.
Semantic Layer	عقود مقاييس وعلاقات وصلاحيات تمنح BI معنى أعمالياً محكوماً.
Verified Autonomy	استقلالية لا تُحسب بعدد الخطوات بل بنسبة المهام الناجحة بعد تحقق ضمن سياسة/ميزانية.
AGI-oriented Architecture	بنية بحثية تجمع reasoning, memory, tools, learning/evals؛ لا تعني وحدها تحقق AGI.

الخلاصة: حالة الفن في 2026 هي «نظام استدلال وتنفيذ قابل للقياس» لا مجرد LLM. أقوى البنى تجعل النموذج جزءاً قابلاً للاستبدال داخل runtime يملك state صريحة، verifiers، evals، sandbox، identity، tool protocols، checkpoints، وmodel routing. الفرق بين prototype وproduction هو القدرة على الاستئناف، تفسير الأثر، منع الصلاحيات الزائدة، وقياس success-per-budget. أما AGI فتبقى هدفاً بحثياً؛ أفضل مسار هندسي نحوها هو بناء وحدات قابلة للتكذيب والقياس بدلاً من إعلان العمومية من كثرة الوكلاء أو حجم الـ knowledge base.

> قاعدة أخيرة — إذا لم يمكن إعادة تشغيل المهمة، وقياس نجاحها، ومعرفة الأدلة التي استخدمتها، وإيقافها قبل side effect خطرة، فهي ليست بنية وكيلية إنتاجية مهما كان النموذج قوياً.

Keywords: *Verified Autonomy • Agentic OS • Reasoning Runtime • Evidence • Safety • AGI-Oriented Engineering*